第 9 章 概率图模型

9.1 基本思想

概率图模型（Probabilistic Graphical Models）以图的方式（有向图或无向图）来表示模型中变量之间的依赖关系及条件独立关系，从而直观地描述复杂概率分布的结构。借助这种图结构，可以确定联合概率分布的分解形式，进而利用这种分解进行统计推断和参数学习。此处的推断（Inference）是指：在给定模型结构和参数的前提下，根据可观测变量的取值，计算目标变量的概率分布（如边缘概率或条件概率）或最有可能的取值。

为了说明如何利用图结构来表示概率分布，我们假设需要对三个随机变量 x_1、x_2 和 x_3 的联合概率分布 $p(x_1, x_2, x_3)$ 进行建模。根据概率的链式法则，该联合分布可以分解为：

$$p(x_1, x_2, x_3) = p(x_3|x_1, x_2)p(x_2|x_1)p(x_1) \tag{9.1}$$

上述分解形式可用图9.1(a) 所示的有向图表示。联合分布分解式右侧的每一项对应图中的一个节点（Node）。对于任意一个节点，其条件概率中的条件变量由指向该节点的父节点（Parents）所组成。以变量 x_3 为例，图9.1(a) 中表示该变量的节点有两个父节点（即指向该节点 x_3 的两条有向边的箭尾分别连接 x_1 和 x_2），因此该节点对应的条件概率为 $p(x_3|x_1, x_2)$。变量 x_1 没有父节点，因此其对应的概率分布不包含条件变量，即为边缘分布 $p(x_1)$。

虽然公式9.1对联合概率分布 $p(x_1, x_2, x_3)$ 的分解始终成立，但联合分布中的各个随机变量本身是对称的（见公式左侧），而分解后的表达式则引入了变量之间的次序关系（如公式右侧），即编号较大的变量以编号较小的变量作为条件。需要注意的是：在实际建模时，通常需要将这些随机变量对应到具体应用领域中的对象，并根据领域知识、数据特征或建模假设等确定变量之间的依赖关系。

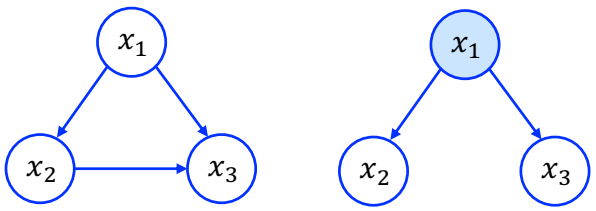


图 9.1: 联合概率分布的有向图表示。图中节点表示随机变量，而节点之间的有向边表示变量之间的条件依赖关系。(a) 联合分布按概率链式法则分解为 $p(x_1, x_2, x_3) = p(x_3|x_1, x_2)p(x_2|x_1)p(x_1)$。(b) 假设给定 x_1 之后（带背景色），变量 x_2 与 x_3 条件独立，则可去除图 (a) 中的有向边 $x_2 \to x_3$，联合分布相应地简化为 $p(x_1, x_2, x_3) = p(x_3|x_1)p(x_2|x_1)p(x_1)$，从而减少建模所需的参数数量。

假设在给定 x_1 的条件下，变量 x_2 和 x_3 条件独立，记为：

$$x_2 \perp x_3 \mid x_1 \tag{9.2}$$

根据这一条件独立关系，在给定 x_1 之后，进一步获知 x_2 的取值不会改变 x_3 的条件分布。因此，公式9.1右侧的 $p(x_3|x_1,x_2)$ 可简化为 $p(x_3|x_1)$，其联合概率分布相应地分解为：

$$p(x_1,x_2,x_3)=p(x_3|x_1)p(x_2|x_1)p(x_1) \tag{9.3}$$

这一分解有效减少了描述联合概率分布所需的参数数量，从而简化了建模过程。例如，假设 x_1、x_2 和 x_3 均为具有 K 个可能取值的离散随机变量。对于 $p(x_3|x_1,x_2)$，条件变量 x_1 和 x_2 共有 K^2 种取值组合。对于每一种组合，都需要确定 x_3 的 K 个可能取值对应的条件概率。由于这些概率之和为 1，只需确定其中的 $K-1$ 个概率，剩余一个概率便随之确定。因此，描述 $p(x_3|x_1,x_2)$ 共需要 $K^2(K-1)$ 个自由参数。相较之下，对于 $p(x_3|x_1)$，条件变量仅有 x_1，其共有 K 种可能的取值，因此只需要 $K(K-1)$ 个自由参数。公式9.3所给出的联合概率分布分解可用图9.1(b)所示的有向图表示。上述过程体现了概率图模型的核心思想之一，即利用变量之间的条件独立关系，将复杂的联合概率分布分解为若干局部条件分布的乘积，进而利用这种分解实现更高效的概率推断与参数学习。

在进一步讨论如何通过调整图结构简化概率建模之前，我们先以一个简单的线性回归模型为例，介绍概率图模型中常用的术语和表示方法。对于输入特征向量 $\boldsymbol{x}$，线性回归模型采用如下函数预测其对应的目标值：

$$f(\boldsymbol{x},\boldsymbol{w})=\boldsymbol{x}^\top\boldsymbol{w} \tag{9.4}$$

其中 $\boldsymbol{w}$ 为模型的权重参数向量。给定包含 N 个样本的训练数据集 $\mathcal{D}=\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，通常可以采用最小二乘法估计模型参数 $\boldsymbol{w}$。参数一旦确定之后，对于给定的输入向量 $\boldsymbol{x}$，模型的预测值由公式9.4计算得出，即模型的预测值依赖于输入向量 $\boldsymbol{x}$ 和权重参数 $\boldsymbol{w}$。相应的概率图表示如图9.2(a) 所示，其中 $\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N\}$ 视为给定的输入，并非模型中的随机变量，因此在图中用实心圆点表示。对于模型中的随机变量，我们进一步区分已观测变量和未观测变量：已观测变量的取值已知，用带有背景色的圆圈表示；而未观测变量的取值未知，仍用空心圆圈表示。

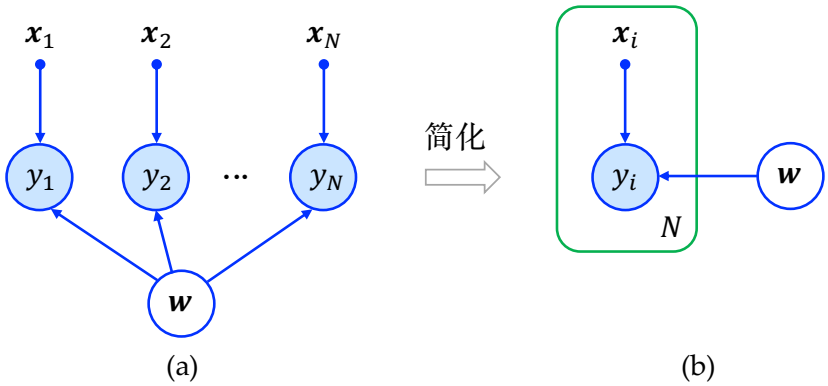


图 9.2: 线性回归模型的图表示及其简化形式。(a) 包含 N 个样本的图表示。其中实心圆点表示给定的输入 $\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N\}$，带背景色的圆圈表示已观测的目标变量 $\{y_1,\ldots,y_N\}$，空心圆圈表示未观测的权重向量 $\boldsymbol{w}$。每个 y_i 的条件分布以对应的输入向量 $\boldsymbol{x}_i$ 和所有样本共享的权重参数 $\boldsymbol{w}$ 为条件。(b) 利用矩形框简化重复结构之后的表示。其中矩形框右下方的 N 表示重复的次数。

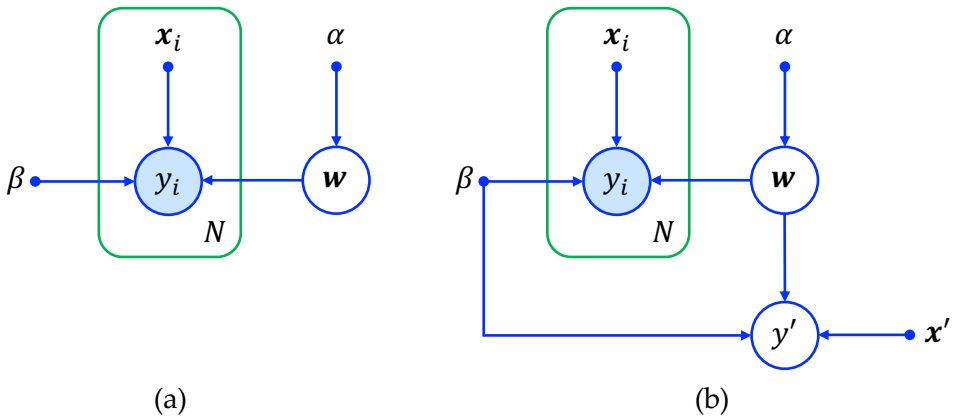
根据图9.2(a) 所示的结构，并且按照上述方式，可将联合概率分布写成图中各节点条件概率或边缘分布的乘积形式，则线性模型中目标变量 $\mathbf{Y}$ 与权重向量 $\boldsymbol{w}$ 的联合概率分布可分解为：

$$p(\mathbf{Y}, \boldsymbol{w}|\mathbf{X}) = p(\boldsymbol{w}) \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i, \boldsymbol{w}) \tag{9.5}$$

其中 $\mathbf{Y} = (y_1, \ldots, y_N)^\top$，且 $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)^\top$。以上分解也体现了模型中所隐含的条件独立假设，即在给定各输入 $\boldsymbol{x}_i$ 和权重 $\boldsymbol{w}$ 之后，各样本的目标变量 $y_1, \ldots, y_N$ 相互条件独立。为避免逐一绘制图9.2(a) 中的 N 组 $\boldsymbol{x}_i \to y_i$ 重复结构，通常会采用如9.2(b) 中的绿色矩形框（Plate）来进行简化表示，并在矩形框的右下角标出结构重复的次数（此处为 N）。

与第1章介绍的多项式拟合类似（多项式拟合关于权重参数是线性的，因此属于线性模型），我们可以引入高斯噪声来刻画观测值与模型预测值之间的随机偏差，并用精度参数 $\beta > 0$ 表示噪声方差的倒数。具体而言，假设在给定输入 $\boldsymbol{x}_i$ 和权重 $\boldsymbol{w}$ 后，目标变量 y_i 服从以 $f(\boldsymbol{x}_i, \boldsymbol{w})$ 为均值、β^{-1} 为方差的高斯分布，即 $p(y_i|\boldsymbol{x}_i, \boldsymbol{w}, \beta) = \mathcal{N}(y_i|f(\boldsymbol{x}_i, \boldsymbol{w}), \beta^{-1})$。此外，我们为权重向量 $\boldsymbol{w}$ 指定零均值的各向同性高斯先验 $p(\boldsymbol{w}|\alpha) = \mathcal{N}(\boldsymbol{w}|\mathbf{0}, \alpha^{-1}\mathbf{I})$，其中 $\alpha > 0$ 为控制先验分布集中程度的精度参数，而 $\mathbf{I}$ 为单位矩阵。相应的概率图表示如图9.3(a) 所示。在给定样本输入 $\mathbf{X}$、精度参数 α 和 β 的条件下，模型的联合概率分布可分解为如下形式：

$$p(\mathbf{Y}, \boldsymbol{w}|\mathbf{X}, \alpha, \beta) = p(\boldsymbol{w}|\alpha) \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i, \boldsymbol{w}, \beta) \tag{9.6}$$



(a) (b)

图 9.3: 贝叶斯线性回归模型训练与预测阶段的概率图表示。(a) 训练阶段：根据输入向量 $\boldsymbol{x}_i$ 及其对应的观测目标 y_i，推断所有样本共享的权重向量 $\boldsymbol{w}$ 的后验分布。其中 α 为权重高斯先验的精度参数，β 为高斯观测噪声的精度参数。(b) 预测阶段：给定新输入 $\boldsymbol{x}'$，结合训练数据得到的权重后验分布，计算待预测目标变量 y' 的后验预测分布。

根据贝叶斯公式，权重向量的后验分布为：

$$p(\boldsymbol{w}|\mathbf{Y}, \mathbf{X}, \alpha, \beta) = \frac{p(\mathbf{Y}, \boldsymbol{w}|\mathbf{X}, \alpha, \beta)}{p(\mathbf{Y}|\mathbf{X}, \alpha, \beta)} \tag{9.7}$$

对于给定的观测数据及超参数 α 和 β，上式中的分母并不依赖于 $\boldsymbol{w}$，仅在后验分布中起归一化

作用。因此，当我们只关心后验分布关于 $\boldsymbol{w}$ 的形式时，可以写为：

$$p(\boldsymbol{w}|\mathbf{Y},\mathbf{X},\alpha,\beta) \propto p(\mathbf{Y},\boldsymbol{w}|\mathbf{X},\alpha,\beta) = p(\boldsymbol{w}|\alpha)\prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i,\boldsymbol{w},\beta) \tag{9.8}$$

上式是我们所熟悉的贝叶斯方法的常见形式，即后验分布正比于先验分布与似然函数的乘积。其中 $p(\boldsymbol{w}|\alpha)$ 为先验分布，连乘项 $\prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i,\boldsymbol{w},\beta)$ 则是在给定输入和观测目标后，关于 $\boldsymbol{w}$ 的似然函数。

估计权重向量 $\boldsymbol{w}$ 的后验分布并非最终目标。我们最终是希望能够基于训练数据集 $\mathcal{D}$，对新输入 $\boldsymbol{x}'$ 对应的目标变量 y' 进行预测。其相应的概率图如图9.3(b) 所示。在给定输入 $\boldsymbol{x}'$、$\mathbf{X}$ 及超参数 α 和 β 的条件下，新目标变量 y'、训练目标 $\mathbf{Y}$ 与权重向量 $\boldsymbol{w}$ 的联合概率分布可分解为：

$$p(y',\mathbf{Y},\boldsymbol{w}|\boldsymbol{x}',\mathbf{X},\alpha,\beta) = p(y'|\boldsymbol{x}',\boldsymbol{w},\beta)p(\boldsymbol{w}|\alpha)\prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i,\boldsymbol{w},\beta) \tag{9.9}$$

根据条件概率的定义，我们有：

$$p(y',\boldsymbol{w}|\mathbf{Y},\boldsymbol{x}',\mathbf{X},\alpha,\beta) = \frac{p(y',\mathbf{Y},\boldsymbol{w}|\boldsymbol{x}',\mathbf{X},\alpha,\beta)}{p(\mathbf{Y}|\boldsymbol{x}',\mathbf{X},\alpha,\beta)} \tag{9.10}$$

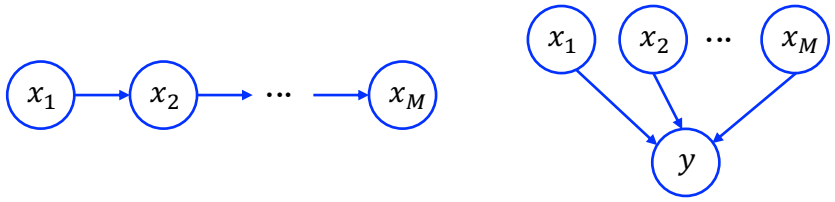
在给定训练数据集后，分母既不依赖于 y'，也不依赖于 $\boldsymbol{w}$。为了得到新目标变量 y' 的后验预测分布，我们对权重向量 $\boldsymbol{w}$ 进行边缘化，即对上式两侧关于 $\boldsymbol{w}$ 积分，得到：

$$p(y'|\mathbf{Y},\boldsymbol{x}',\mathbf{X},\alpha,\beta) \propto \int p(y',\mathbf{Y},\boldsymbol{w}|\boldsymbol{x}',\mathbf{X},\alpha,\beta)\, d\boldsymbol{w} \tag{9.11}$$

其中 $\mathbf{Y}$ 直接取训练数据集中已观测到的目标值（即作为给定条件，而非随机变量处理）。上式中的正比关系是由于省略了与 y' 无关的归一化因子（即公式9.10右侧的分母项）。

接下来，我们以离散随机变量为例，简要讨论如何通过引入条件独立假设来调整图结构，进而简化概率建模。假设模型包含 M 个离散随机变量，并且每个变量均有 K 个可能取值。若不引入任何独立性假设，则联合分布可用全连接的有向无环图表示，其中任意两个不同节点之间都有一条有向边。描述这一联合分布需要 K^M-1 个自由参数。这是因为 M 个变量共有 K^M 种可能的取值组合，每种组合都需要指定一个概率。由于这些概率之和必须为 1，确定其中 K^M-1 个概率后，剩余一个概率便随之确定。因此，当 $K \geq 2$ 时，模型所需的参数数量随变量数量 M 呈指数增长。另一个极端情况是，假设所有随机变量相互独立，对应的概率图中不存在任何边，联合分布可分解为 M 个单变量边缘分布的乘积。每个边缘分布只需 $K-1$ 个自由参数，因此整个模型仅需 $M(K-1)$ 个自由参数。此时，模型的参数数量随变量数量 M 呈线性增长。

除上述两种极端情况之外，还存在多种介于两者之间的图结构。例如，对于图9.4(a) 所示的链式结构，边缘分布 $p(x_1)$ 需要 $K-1$ 个参数，而建模每个条件分布 $p(x_i|x_{i-1})$ 需要 $K(K-1)$ 个参数。由于这样的条件分布共有 $M-1$ 个，模型所需的参数总数为 $K-1+(M-1)K(K-1)$。因此，模型的参数数量随变量数量 M 呈线性增长，并随每个变量可能取值数 K 呈二次增长。若用链式结构表示时序数据，则相应的模型满足一阶马尔科夫性（也称无后效性）。即给定前一时刻的状态后，当前状态的条件分布不再依赖于更早的历史状态。换言之，在给定 x_{i-1} 的条件下，x_i 与更早的历史状态 $(x_1,\ldots,x_{i-2})$ 条件独立。



(a) (b)

图 9.4: 链式结构与参数化示例。设每个变量 x_i 均有 K 个可能取值。(a) 链式结构。除变量 x_1 外，每个节点 x_i 仅以 x_{i-1} 为父节点。因此，给定前一变量 x_{i-1} 后，x_i 与更早的变量 $(x_1, \ldots, x_{i-2})$ 条件独立。模型共有 $(K-1)+(M-1)K(K-1)$ 个自由参数。(b) 条件分布的参数化。若 y 为二值变量，建模条件分布 $p(y|x_1, \ldots, x_M)$ 需要 K^M 个自由参数。若采用 Logistic 回归模型替代条件概率，则仅需 $M+1$ 个参数，其中包括一个截距参数。

我们还可以通过参数共享来进一步简化模型。仍以上述链式结构为例，假设所有条件分布 $p(x_i|x_{i-1})$ 均共享同一组参数，则描述这些条件分布仅需 $K(K-1)$ 个参数。加上边缘分布 $p(x_1)$ 所需的 $K-1$ 个参数，整个模型的参数总数减少为 $K-1+K(K-1) = K^2-1$。因此，在 K 固定时，参数数量不再随链的长度 M 增长。另一种减少模型参数的常用方法是**参数化**（Parameterization），即用参数较少的函数形式替代条件概率。以图9.4(b) 所示的结构为例，假设目标变量 y 取 0 或 1 二值之一。由于每种输入取值组合需要一个自由参数来确定对应的概率值，建模 $p(y|x_1, \ldots, x_M)$ 共计需要 K^M 个参数。若将每个 x_i 作为一个数值特征，则可采用如下形式的 Logistic 回归模型来表示该条件分布：

$$p(y=1|x_1, \ldots, x_M) = \sigma\left(w_0 + \sum_{i=1}^{M} w_i x_i\right) \tag{9.12}$$

其中 $\sigma(\cdot)$ 为 Sigmoid 函数。此时，模型仅包含 M 个权重参数和一个截距参数 w_0，参数总数为 $M+1$。这种参数化方法通过限制条件分布的函数形式，大幅减少了所需的参数数量。

9.2 条件独立与图结构

上一节的讨论表明，利用条件独立（Conditional Independence）假设可以简化模型结构，并有助于降低推断与学习的计算代价。本节将进一步讨论如何从概率图的结构中识别其所蕴含的条件独立假设。通过识别和分析这些条件独立假设，我们可以结合领域知识，考察概率图隐含的条件独立假设是否合理，从而评估概率图所对应的模型是否适合用于目标问题的建模。

考虑 x_1、x_2 和 x_3 三个随机变量的情况。在给定 x_1 的条件下，x_2 与 x_3 的条件独立关系有以下两种等价表述。其一，在给定 x_1 和 x_2 的条件下，x_3 的条件分布与仅给定 x_1 时相同，即：

$$p(x_3|x_2, x_1) = p(x_3|x_1) \tag{9.13}$$

这意味着，在已知 x_1 的取值后，进一步获知 x_2 的取值不会改变 x_3 的条件分布。此时，我们称：x_3 在给定 x_1 的条件下独立于 x_2。其二，在给定 x_1 的条件下，x_2 与 x_3 的条件联合分布可以分

解为各自条件分布的乘积，即：

$$\begin{aligned} p(x_3, x_2|x_1) &= p(x_3|x_2, x_1)p(x_2|x_1) \\ &= p(x_3|x_1)p(x_2|x_1) \end{aligned} \tag{9.14}$$

此时，我们称：给定 x_1 时，x_2 与 x_3 条件独立。这一条件联合分解形式也表明，条件独立关系具有对称性，即在给定 x_1 的条件下，x_3 条件独立于 x_2，同时也意味着 x_2 条件独立于 x_3。

接下来，我们将讨论三种由三个随机变量构成的基本有向图结构：分叉结构、链式结构和碰撞结构（如图9.5所示）。我们将重点分析中间节点（即图中随机变量 x_1）对应的变量在处于已观测或未观测状态时，对于另两个变量之间条件独立关系的影响。

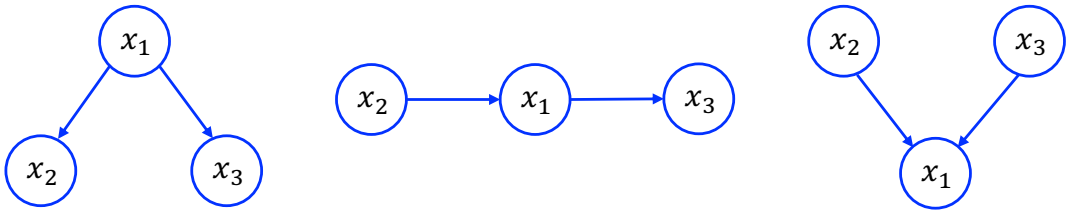


图 9.5: 三个随机变量构成的基本有向图结构。(a) 分叉结构。(b) 链式结构。(c) 碰撞结构。

对于图9.5(a) 所示的分叉结构，联合概率分布可按图结构分解为（即联合分布分解为各节点条件分布或边缘分布的乘积形式）：

$$p(x_1, x_2, x_3) = p(x_3|x_1)p(x_2|x_1)p(x_1) \tag{9.15}$$

当这三个变量均处于未观测（Unobserved）状态时，我们先来考察 x_2 与 x_3 之间的无条件独立关系。为此，我们将上式两侧关于 x_1 积分（若为离散变量，则将积分替换为对其所有可能取值的求和），可得：

$$\int p(x_1, x_2, x_3)\, dx_1 = p(x_2, x_3) = \int p(x_3|x_1)p(x_2|x_1)p(x_1)\, dx_1$$

上式结果一般不能分解为 $p(x_2)p(x_3)$ 的形式。因此，在未给定 x_1 的条件下，分叉结构并不保证 x_2 与 x_3 无条件独立（但也不能断言它们必然不独立）。若 x_2 与 x_3 在无条件下不独立，可记为：

$$x_2 \not\perp x_3 \mid \emptyset \tag{9.16}$$

其中空集 $\emptyset$ 表示不以任何变量的取值作为条件，而 $\not\perp$ 表示不独立。

当 x_1 的取值已观测（Observed）时，我们可以考察给定 x_1 后的条件联合分布，并在概率图中用带底色的圆圈来表示 x_1（如图9.1(b) 所示），则 x_2 和 x_3 的条件联合分布可分解为：

$$\begin{aligned} p(x_2, x_3|x_1) &= \frac{p(x_1, x_2, x_3)}{p(x_1)} \\ &= \frac{p(x_3|x_1)p(x_2|x_1)p(x_1)}{p(x_1)} \quad \text{（利用公式9.15）} \\ &= p(x_3|x_1)p(x_2|x_1) \end{aligned}$$

因此，给定 x_1 后，x_2 与 x_3 的条件联合分布可以分解为各自条件分布的乘积，即两者条件独立，可以记为：

$$x_2 \perp x_3 \mid x_1$$

从图9.5(a) 可以看出，节点 x_2 与 x_3 之间存在一条经过节点 x_1 的通路（Path）。需要注意的是，这里的通路并不要求沿着有向边的箭头方向行进。在这条通路上，指向 x_2 与 x_3 的两条有向边均以箭尾与中间节点 x_1 相连，这种连接方式称为“尾对尾”（Tail-to-tail），对应于分叉结构。当中间节点 x_1 的取值已观测，即将 x_1 作为给定条件时，我们称这条通路被 x_1 所阻断（Blocked）。换言之，在给定 x_1 时，x_2 与 x_3 条件独立。当三个变量均未观测时，这条通路未被阻断，因此不能仅凭分叉结构推出 x_2 与 x_3 无条件独立。

对于图9.5(b) 所示的链式结构，联合概率分布可按图结构分解为：

$$p(x_1, x_2, x_3) = p(x_2)p(x_1|x_2)p(x_3|x_1) \tag{9.17}$$

我们同样先考虑三个变量均未观测的情况。为了考察 x_2 与 x_3 之间是否存在无条件独立关系，将上式两侧关于 x_1 进行积分：

$$p(x_2, x_3) = \int p(x_2)p(x_1|x_2)p(x_3|x_1)\, dx_1 = p(x_2) \int p(x_3|x_1)p(x_1|x_2)\, dx_1 = p(x_2)p(x_3|x_2)$$

上式结果一般不能进一步分解为 $p(x_2)p(x_3)$。因此，在未给定 x_1 的条件下，链式结构也并不能保证 x_2 与 x_3 无条件独立。

当 x_1 的取值已观测时（将图9.5(b) 中表示 x_1 的圆圈加上背景色），我们来考察给定 x_1 后 x_2 与 x_3 的条件联合分布的分解形式：

$$\begin{aligned}
p(x_2, x_3|x_1) &= \frac{p(x_1, x_2, x_3)}{p(x_1)} \\
&= \frac{p(x_2)p(x_1|x_2)p(x_3|x_1)}{p(x_1)} \quad \text{（利用公式9.17）} \\
&= \frac{p(x_1, x_2)p(x_3|x_1)}{p(x_1)} \\
&= \frac{p(x_2|x_1)p(x_1)p(x_3|x_1)}{p(x_1)} \\
&= p(x_2|x_1)p(x_3|x_1)
\end{aligned}$$

因此，在给定 x_1 的条件下，x_2 与 x_3 条件独立，即 $x_3 \perp x_2 \mid x_1$。

同样地，我们观察图9.5(b) 中连接 x_2 与 x_3 之间的通路。在中间节点 x_1 处，一条有向边以箭头与其相连，另一条有向边以箭尾与其相连，因而这种连接方式称为“头对尾”（Head-to-tail），对应于链式结构。当三个变量均未观测时，这条通路未被阻断，x_2 与 x_3 通常相互依赖。但当 x_1 的取值已观测，即将 x_1 作为给定条件时，这条通路被 x_1 阻断。因此，在给定 x_1 的条件下，链式结构中的 x_2 与 x_3 条件独立。

最后，我们考察图9.5(c) 所示的碰撞结构。在这一结构中，两条有向边 $x_2 \to x_1$ 和 $x_3 \to x_1$ 均指向中间节点 x_1，两个箭头在此汇聚，形成“头对头”（Head-to-head）的连接方式，因此称为

碰撞结构。按图9.5(c) 的结构，联合概率分布可分解为：

$$p(x_1, x_2, x_3) = p(x_2)p(x_3)p(x_1|x_2, x_3) \tag{9.18}$$

同样地，我们先考虑三个变量均未观测的情况。对上式两侧同时关于 x_1 进行积分，可得：

$$p(x_2, x_3) = \int p(x_2)p(x_3)p(x_1|x_2, x_3)\, dx_1 = p(x_2)p(x_3) \underbrace{\int p(x_1|x_2, x_3)\, dx_1}_{=1} = p(x_2)p(x_3)$$

因此，与分叉结构和链式结构有所不同，在三个变量均未观测时，碰撞结构能够保证 x_2 与 x_3 无条件独立，即：

$$x_2 \perp x_3 \mid \emptyset \tag{9.19}$$

当 x_1 取值已观测时，我们考察给定 x_1 后 x_2 与 x_3 的条件联合分布：

$$\begin{aligned} p(x_2, x_3|x_1) &= \frac{p(x_1, x_2, x_3)}{p(x_1)} \\ &= \frac{p(x_2)p(x_3)p(x_1|x_2, x_3)}{p(x_1)} \quad \text{（利用公式9.18）} \end{aligned}$$

以上结果一般不能进一步分解为 $p(x_2|x_1)p(x_3|x_1)$ 的形式。因此，给定 x_1 之后，x_2 与 x_3 通常不再独立，此时可记为：

$$x_2 \not\perp x_3 \mid x_1 \tag{9.20}$$

综上所述，分叉结构和链式结构在未给定中间节点 x_1 时，并不保证 x_2 与 x_3 无条件独立；而在给定 x_1 之后，则能够保证两者条件独立。对于碰撞结构，情况正好相反：在未给定 x_1 时，x_2 与 x_3 无条件独立；在给定 x_1 后，原本独立的两个变量可能产生条件依赖。表9.1总结了上述三种基本有向图结构在中间节点已观测和未观测两种情况下所蕴含的独立性关系。

在上述碰撞结构中，当中间节点 x_1 未观测时，随机变量 x_2 与 x_3 无条件独立；而当 x_1 已观测时，两者在给定 x_1 的条件下反而可能不再独立。这一现象似乎有些违反直觉，我们通过一个简单的例子加以说明。假设我们分别抛掷两枚正常的硬币，并根据抛掷结果来控制一盏灯的亮与灭：只要有一枚硬币为正面时，灯就亮；而当两枚硬币均为反面时，灯才灭。两枚硬币的结果分别对应变量 x_2 和 x_3，灯的状态对应变量 x_1，从而构成碰撞结构 $x_2 \to x_1 \leftarrow x_3$。

未观测灯的状态时，知道第一枚硬币的结果，并不会改变第二枚硬币为正面的概率，该概率始终为 $1/2$。然而，当观测到灯亮时，我们便知道两枚硬币中至少有一枚为正面。此时，如果进一步得知第一枚为反面，便可确定第二枚必为正面；如果得知第一枚为正面，则第二枚既可以是正面，也可以是反面，两种情况等概率。因此，在已知灯亮的条件下，第一枚硬币的结果会改变我们对第二枚硬币结果的概率判断，两者在这一条件下不再独立。其中关键在于：观测到共同作用的结果后，两个原本独立的变量同时受到这一结果的约束，因而可能产生条件依赖。

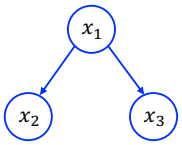
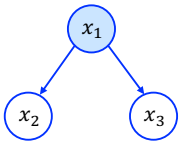
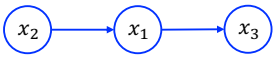
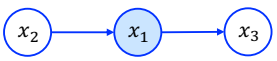
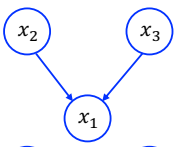
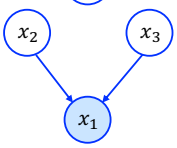
在有向概率图模型中，上述三种基本图结构中的通路阻断规则可以推广到一般的有向无环图，用于判断随机变量集合之间的条件独立关系。下面我们给出**有向分离**（D-separation）性质的一般表述 [56]。考虑一个有向无环图，其中 A、B 和 C 为任意三个互不相交的节点集合，它们的并集不必包含图中的所有节点。我们希望判断该图结构是否蕴含条件独立关系 $A \perp B \mid C$。

为此，我们考察从集合 A 中任意节点到集合 B 中任意节点的所有可能通路。如果某条通路上存在一个满足以下任一条件的中间节点，则称这条通路相对于给定集合 C 被阻断：

- 通路上节点的连接方式是“头对尾”或“尾对尾”，并且该节点属于集合 C；
- 通路上节点的连接方式是“头对头”，并且该节点及其所有后代节点均不属于集合 C。

如果所有这些通路都被阻断，则称集合 A 与 B 关于集合 C 是有向分离的。此时，任何按照该图结构分解的联合概率分布都满足条件独立关系 $A \perp B \mid C$。上述第二条规则之所以同时涉及碰撞节点及其后代节点，是因为即使碰撞节点本身未观测，其后代节点的观测也可能提供有关该碰撞节点的信息，并在通路两侧的变量之间引入条件依赖。因此，只要该碰撞节点或其一后代节点属于集合 C，该碰撞节点就不再阻断这条通路。需要注意的是，后代节点已观测并不意味着碰撞节点本身已被观测或其取值已被确定。

表 9.1: 三种基本有向图结构及其条件独立性

有向图结构	连接方式	中间节点状态	图结构	条件独立性
分叉结构	尾对尾	x_1 未观测		$x_2 \not\perp x_3 \mid \emptyset$
		x_1 已观测		$x_2 \perp x_3 \mid x_1$
链式结构	头对尾	x_1 未观测		$x_2 \not\perp x_3 \mid \emptyset$
		x_1 已观测		$x_2 \perp x_3 \mid x_1$
碰撞结构	头对头	x_1 未观测		$x_2 \perp x_3 \mid \emptyset$
		x_1 已观测		$x_2 \not\perp x_3 \mid x_1$

在有向概率图模型中，有向分离是指：相对于给定的变量集合，连接另外两个变量集合的所有通路均被阻断。因此，对于任何按照该图结构分解的联合概率分布，在已知条件变量（如集合 C）取值的情况下，进一步获知其中一组变量的取值（如集合 A），不会改变另一组变量的条件概率分布（如集合 B）。图9.6展示了在不同观测条件下，判断随机变量 x_1 与 x_5 是否有向分离的例子。在图9.6(a) 中，从 x_1 到 x_5 的通路未被节点 x_4 阻断，这是因为 x_4 在这条通路上呈“尾对尾”连接（即分叉结构），并且 x_4 未被观测。节点 x_2 也未阻断这条通路。尽管 x_2 在

该通路上呈“头对头”连接（即碰撞结构），但其后代节点 x_3 属于条件集合，即 x_3 已被观测。因此，这条通路未被阻断，x_1 与 x_5 并非关于 x_3 有向分离，不能由该图结构推出条件独立关系 $x_1 \perp x_5 \mid x_3$。

在图9.6(b) 中，从 x_1 到 x_5 的通路被节点 x_4 阻断，因为 x_4 在这条通路上呈“尾对尾”连接，且 x_4 已被观测。此外，这条通路也被节点 x_2 所阻断，因为 x_2 在该通路上呈“头对头”连接，且它及其后代节点均不属于条件集合。因此，x_1 与 x_5 关于 x_4 是有向分离的，任何按照该图结构分解的联合概率分布都满足条件独立关系 $x_1 \perp x_5 \mid x_4$。

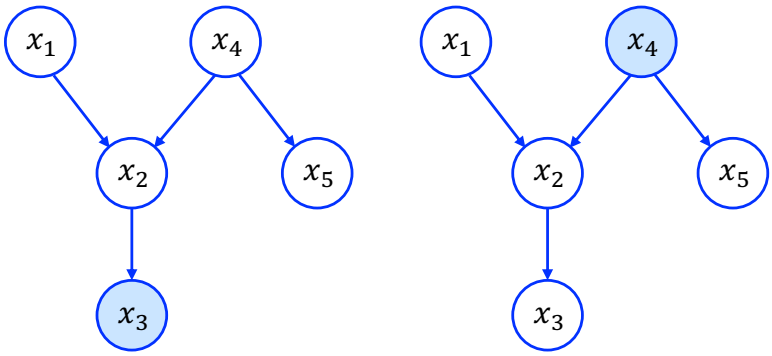


图 9.6: 有向分离（D-separation）示例。(a) 给定 x_3 时，连接 x_1 与 x_5 的通路既未被 x_2 阻断，也未被 x_4 阻断，因此两者关于 x_3 并非有向分离。(b) 给定 x_4 时，连接 x_1 与 x_5 的通路在 x_2 和 x_4 处均被阻断，因而 x_1 与 x_5 是关于 x_4 有向分离的。

在判断有向分离时，图9.3(b) 中以实心圆点表示的超参数 α 和 β 均视为已观测节点。这些超参数节点通常没有父节点。任何以超参数节点为中间节点的通路，都在该节点处呈“尾对尾”连接（即分叉结构），并由于该超参数的取值已观测而被阻断。因此，在判断概率图模型中随机变量之间的有向分离关系时，可以忽略这些超参数节点的影响。

在构建有向概率图模型之后，我们可以利用有向分离准则识别出图模型所隐含的条件独立关系，再结合领域知识判断这些假设是否合理。此外，有向分离准则也为简化概率推断提供了依据。例如，如果 x_2 与 x_3 关于 x_1 有向分离，那么对于任何按照该图结构分解的联合概率分布，都有 $p(x_2|x_3, x_1) = p(x_2|x_1)$。这意味着，在已经给定 x_1 的情况下，进一步获知 x_3 不会改变 x_2 的条件概率分布，因此可以从该分布的条件项中省去 x_3。

马尔可夫毯（Markov Blanket）是上述思想的重要应用。如有向概率图9.7所示，节点 x_i 的马尔可夫毯由其父节点、子节点以及子节点的其他父节点所组成。在给定其余所有变量的取值之后，x_i 的条件分布仅依赖于其马尔可夫毯中变量（图中带背景色的节点）的取值。这一性质可以通过有向分离准则来解释。考察从 x_i 出发、通向其马尔可夫毯之外任意节点的通路。如果通路首先经过 x_i 的某个父节点，那么该父节点在通路上呈“头对尾”或“尾对尾”连接，因而在给定父节点的取值后，这条通路被阻断。如果通路首先经过 x_i 的某个子节点，则需要区分两种情况。若通路沿该子节点的出边继续延伸，则该子节点在通路上呈“头对尾”连接，因而在给定

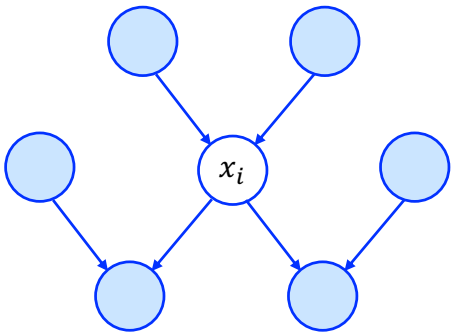


图 9.7: 有向概率图中的马尔可夫毯（Markov Blanket）示例。节点 x_i 的马尔可夫毯由其父节点、子节点以及子节点的其他父节点组成。在给定图中其余所有变量（以带背景色的节点表示）的取值后，x_i 的条件分布仅依赖于其马尔可夫毯中变量的取值。

该子节点的取值后，此通路被阻断。若通路从该子节点转向它的另一个父节点，则该子节点在通路上呈“头对头”连接（即碰撞结构），给定其取值并不会阻断这条通路。不过，该子节点的另一个父节点也属于马尔可夫毯内的随机变量，并且在继续通向毯外节点的通路上呈“头对尾”或“尾对尾”连接，因此，给定这个父节点的取值便能阻断通路。由此可见，给定 x_i 的马尔可夫毯后，连接 x_i 与毯外节点的所有通路均被阻断，因而 x_i 与毯外所有变量条件独立。这意味着，在已知马尔可夫毯中随机变量取值的情况下，毯外变量不再为 x_i 提供额外信息，因此计算 x_i 的条件分布只需利用马尔可夫毯中的局部信息。马尔可夫毯体现了“马尔可夫性质”的核心思想：给定适当的局部信息后，其余信息就不再影响目标变量的条件分布。

9.3 有向概率图模型

我们以简单的有向图为例，已经介绍了联合概率分布的分解、条件独立、有向分离和马尔可夫毯等概念。本节将在此基础上，进一步讨论几种常用的有向概率图模型。有向概率图模型通常称为贝叶斯网络（Bayesian Network）。这类模型利用有向无环图表示随机变量之间的依赖结构，便于描述数据的生成过程，也可用于刻画变量之间的时间依赖关系。朴素贝叶斯模型和隐马尔可夫模型都是其中的典型代表。

贝叶斯网络由一个有向无环图以及各节点对应的局部条件概率分布共同构成，其联合概率分布可分解为所有局部条件分布的乘积。其中每个节点对应的条件分布均以其父节点所对应的变量为条件。因此，对于包含 N 个节点的贝叶斯网络，其联合概率分布可写为：

$$p(\boldsymbol{x}) = \prod_{i=1}^{N} p(x_i|\mathrm{pa}(x_i)) \tag{9.21}$$

其中 $\boldsymbol{x} = (x_1, \ldots, x_N)^\top$，$\mathrm{pa}(x_i)$ 表示节点 x_i 的所有父节点所构成的变量集合。对于没有父节点的根节点，相应的因子就是其边缘分布 $p(x_i)$。上述公式给出了有向概率图模型中联合概率分布的一般分解形式。虽然上式中将每个节点对应于单个随机变量，但也可以用一个节点表示一组随机变量或一个随机向量。

需要注意的是，贝叶斯网络的图结构中不能存在环，即不能从某个节点出发，沿着有向边的箭头方向经过若干条边后又回到起点。满足这一条件的有向图称为有向无环图（Directed Acyclic Graph）。有向无环图中的节点可以按某种顺序进行编号，我们通常习惯让每条有向边都从编号较小的节点指向编号较大的节点，这样的节点排列称为拓扑序。

公式9.21所示的分解形式使得模型的对数似然可以表示为各局部条件分布对应的对数似然之和。当训练数据中各节点变量均已观测，且各局部条件分布的参数互不共享、不存在跨局部分布的参数约束时，则最大化模型的对数似然等价于分别最大化各局部对数似然。因此，可以分别求得各局部条件分布参数的最大似然估计，从而简化模型的参数学习。

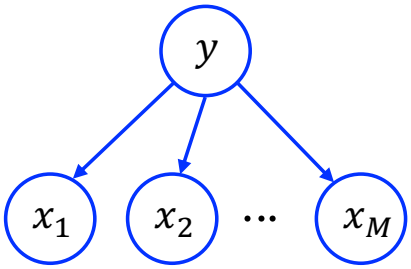


图 9.8: 朴素贝叶斯模型的有向概率图表示。该模型假设，在给定类别标签 y 的条件下，各输入特征 x_j 相互条件独立。

接下来，我们介绍**朴素贝叶斯模型**（Naive Bayes）。这是一种经典的分类模型，它通过引入条件独立假设来简化模型表示与参数估计，其对应的有向概率图如图9.8所示。该模型的目标是根据包含 M 个特征的输入向量 $\boldsymbol{x} = (x_1, \ldots, x_M)^\top$，预测对应的类别标签 y。一般而言，即使给定类别标签，不同输入特征之间仍可能存在依赖关系。若不对条件分布 $p(\boldsymbol{x}|y)$ 施加额外限制，即使每个特征均为二值变量，特征向量也有 2^M 种可能的取值组合，因而每个类别对应的条件分布需要 $2^M - 1$ 个自由参数。为了减少参数数量，朴素贝叶斯模型假设：在给定类别标签 y 的条件下，各特征相互条件独立（即其条件联合分布可以分解为各特征条件分布的乘积）：

$$x_j \perp x_k \mid y \tag{9.22}$$

其中 $j \neq k$。这一假设意味着，类别标签 y 与输入特征向量 $\boldsymbol{x}$ 的联合概率分布可以分解为：

$$p(\boldsymbol{x}, y) = p(y) \prod_{j=1}^{M} p(x_j|y) \tag{9.23}$$

给定包含 N 个样本的训练数据集 $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$，模型的对数似然可以写成以下形式：

$$\sum_{i=1}^{N} \ln p(y_i) + \sum_{i=1}^{N} \sum_{j=1}^{M} \ln p(x_{ij}|y_i) \tag{9.24}$$

其中 x_{ij} 表示第 i 个样本的第 j 维特征。对于离散特征，类别先验分布和各特征条件分布的最大似然估计均可通过简单的频数统计得到：

$$\hat{p}(y = c) = \frac{N_c}{N}, \quad p(x_j = v|y = c) = \frac{N_{cjv}}{N_c} \tag{9.25}$$

其中 N_c 表示训练数据集中类别为 c 的样本数，N_{cjv} 表示这些样本中第 j 个特征取值为 v 的样本数。在估计 $p(x_j|y)$ 时，只需统计该特征的各个取值与类别标签共同出现的次数，无需统计不同特征之间的各种取值组合。如此，高维类条件分布的估计便转化为多个一维条件分布的估计，从而简化模型的参数学习。需要注意的是，各特征在给定类别标签后的条件独立性是建模时预先引入的假设，并不是实际学习后所得到的结论。即使真实数据不满足上述条件独立假设，模型仍会按这一分解形式进行估计，因此所得到的条件分布可能与真实分布存在偏差。然而，以适当的模型偏差换取更少的参数和更简单的参数估计过程，有时可能是一种有益的折中与权衡。

9.4 无向概率图模型

在前一节中，我们介绍了有向概率图模型，其联合概率分布可以分解为图中各节点对应的局部条件概率分布的乘积。此外，有向无环图还蕴含一组条件独立关系，任何按照该有向图结构分解的概率分布都需要满足这些条件独立关系。我们将在本节中讨论另一大类概率图模型，即无向图模型（Undirected Graphical Model）。与有向图模型类似，无向图模型也从两个方面描述概率分布：一是规定联合概率分布的分解形式，二是刻画变量之间的条件独立关系。

有向概率图模型通常称为贝叶斯网络，而无向概率图模型则称为**马尔可夫随机场**（Markov Random Field）或马尔可夫网络（Markov Network）。无向图由一组节点以及连接节点的边组成，每个节点对应一个随机变量或一组随机变量，每条边连接一对节点。与有向图不同，无向图中的边不带箭头，用于表示变量之间不区分方向的依赖关系。无向图模型适合表达具有空间邻接关系的变量之间的相互作用，以及多个变量之间的相互约束。典型的无向概率图模型包括伊辛模型（第6.6节）、玻尔兹曼机（第7.2节）、条件随机场（第13.2节）等。

对于有向图，我们通常采用有向分离准则来判断图结构是否蕴含某个条件独立关系，其基本思路是检查连接两个节点的所有通路是否都被给定的条件集合所阻断。其中碰撞结构的阻断规则与分叉结构和链式结构有所不同：不仅需要考察碰撞节点本身是否属于条件集合，还需要考虑其后代节点是否属于条件集合（如图9.6(a) 所示）。

相较之下，无向图中的边没有方向，节点之间也不存在父子关系，因此无需区分表9.1所示的各种结构。可以预见，无向图中判断变量之间的条件独立性关系会较有向图更为直观和简单。首先，如果两个节点之间不存在任何通路，即这两个节点分别属于不连通的两个子图，则这两个节点必然相互独立。其次，若连接两个节点的每条通路上都至少存在一个已观测的中间节点，则在给定这些已观测节点所对应变量的取值后，这两个节点条件独立。对于第二种情况，可以想象将这些已观测节点及与其相连的边从图中删除。此时，两个节点将分属两个不连通的子图。因此，在给定这些已观测变量的条件下，这两个节点所表示的随机变量条件独立。

我们可以将上述在无向图中判断条件独立关系的方法推广到由多个随机变量组成的集合。如图9.9所示，假设集合 $C = \{x_4, x_5\}$ 中的随机变量均已观测，且从集合 $A = \{x_1, x_2, x_3\}$ 中任一节点到集合 $B = \{x_6, x_7, x_8\}$ 中任一节点的所有通路都经过集合 C 中的节点，因此这些通路均被集合 C 中的节点所阻断。此时，我们称集合 C 将集合 A 与 B 分离。在给定集合 C 中变量

的取值后，A 与 B 这两组随机变量条件独立，记为：

$$A \perp B \mid C \tag{9.26}$$

如果将集合 C 中变量及与其相连的边从图中删除，集合 A 和 B 之间显然将再无通路。

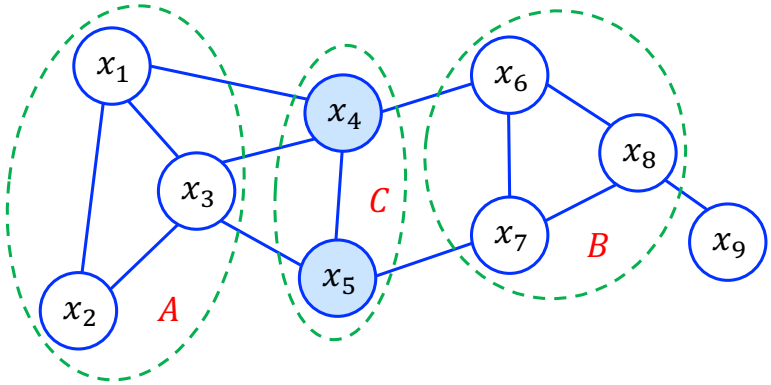


图 9.9: 无向图的示例。其中集合 $C = \{x_4, x_5\}$ 中的随机变量均已观测，从集合 $A = \{x_1, x_2, x_3\}$ 中任意节点到集合 $B = \{x_6, x_7, x_8\}$ 中任意节点的每一条通路都至少经过集合 C 中的一个节点。因此，给定集合 C 中变量的取值之后，集合 A 与 B 这两组随机变量条件独立，记为 $A \perp B \mid C$。

接下来，我们讨论如何根据一个无向图，对图中节点表示的随机变量的联合概率分布进行分解。我们从一个简单的例子出发，考虑由三个变量组成的无向链式结构 $x_1 — x_2 — x_3$，其中变量 x_1 与 x_2 之间、x_2 与 x_3 之间分别有一条无向边相连。这个简单的无向图表达了如下条件独立关系：

$$x_1 \perp x_3 \mid x_2 \tag{9.27}$$

因此，这三个随机变量的联合概率分布可以分解为：

$$\begin{aligned} p(x_1, x_2, x_3) &= p(x_2)p(x_1, x_3|x_2) \\ &= p(x_2)p(x_1|x_2)p(x_3|x_2) \end{aligned}$$

将上述乘积中的因子重新定义如下：

$$\psi_{12}(x_1, x_2) = p(x_2)p(x_1|x_2), \quad \psi_{23}(x_2, x_3) = p(x_3|x_2)$$

其中 $\psi_{12}(x_1, x_2)$ 和 $\psi_{23}(x_2, x_3)$ 通常称为势函数（Potential Function）。于是我们有：

$$p(x_1, x_2, x_3) = \psi_{12}(x_1, x_2)\psi_{23}(x_2, x_3) \tag{9.28}$$

从上式可以看出，这个无向图对应的联合概率分布可以表示为局部势函数的乘积。图中所蕴含的条件独立关系使得原本涉及三个变量的联合分布能够分解为两个分别只涉及相邻变量的因子。

对于更一般的无向图，Hammersley-Clifford 定理 [16] 指出：一个严格为正的联合概率分布满足该图所表达的条件独立关系，当且仅当它可以表示为图中各极大团所对应的势函数的乘积，并经归一化后得到：

$$p(\boldsymbol{x}) = \frac{1}{\mathcal{Z}} \prod_{C} \psi_C(\boldsymbol{x}_C) \tag{9.29}$$

其中乘积遍历图中的所有极大团 C，$\boldsymbol{x}_C$ 表示由极大团 C 中的变量组成的向量，而 $\psi_C(\boldsymbol{x}_C)$ 为相应的势函数。$\mathcal{Z}$ 被称为**配分函数**（Partition Function），它是保证联合概率分布归一化的常数。对于离散变量，配分函数定义为：

$$\mathcal{Z} = \sum_{\boldsymbol{x}} \prod_{C} \psi_C(\boldsymbol{x}_C) \tag{9.30}$$

其中求和遍历所有变量可能的取值组合，从而保证 $\sum_{\boldsymbol{x}} p(\boldsymbol{x}) = 1$。对于连续变量，则将上述求和操作替换为相应的积分。以离散变量为例，上述定理中的“严格为正”是指：对于各变量取值空间中的每一种取值组合 $\boldsymbol{x}$，均有 $p(\boldsymbol{x}) > 0$。

在无向图中，如果一个节点集合中的任意两个不同节点之间都有一条边直接相连，则称该节点集合为一个团（Clique）。换言之，团中的节点两两相连。如果一个团不能通过加入图中的其他节点而形成更大的团，则称其为**极大团**（Maximal Clique）。也就是说，对于团外的任意一个节点，团内至少有一个节点与它之间不存在直接相连的边。如图9.9所示，节点 x_1 和 x_2 构成一个团，但并不是极大团。这是因为 x_3 与 x_1 和 x_2 之间均有直接相连的边，因此加入 x_3 之后，集合 $\{x_1, x_2, x_3\}$ 仍然构成一个团，且比原来的团更大。然而，如果再加入节点 x_4，所得集合就不再是团，因为 x_4 与 x_2 之间没有边直接相连。综上所述，集合 $A = \{x_1, x_2, x_3\}$ 构成该图中的一个极大团。图9.9中的无向图共有 7 个极大团。因此，满足该图所表达的条件独立关系的联合概率分布可以分解为：

$$\begin{aligned} p(\boldsymbol{x}) = \frac{1}{\mathcal{Z}} & \psi_{123}(x_1, x_2, x_3)\psi_{134}(x_1, x_3, x_4)\psi_{345}(x_3, x_4, x_5) \\ & \psi_{46}(x_4, x_6)\psi_{57}(x_5, x_7)\psi_{678}(x_6, x_7, x_8)\psi_{89}(x_8, x_9) \end{aligned}$$

此处可能会产生一个疑问：为什么有向图模型（即贝叶斯网络）对应的联合概率分布分解式不需要额外的归一化因子，而无向图模型（即马尔可夫网络）的分解式却需要引入配分函数 $\mathcal{Z}$ 进行归一化？简单而言，贝叶斯网络分解式中的每一个因子都是条件分布或边缘分布，这些因子各自满足归一化条件，并按照有向无环图的结构依次相乘，从而保证所得联合概率分布也满足归一化条件。无向图模型分解式中的势函数只要求非负，并不要求其本身是概率分布，其乘积也不一定满足归一化条件，因此需要通过配分函数 $\mathcal{Z}$ 进行整体归一化。由于公式9.28中的两个势函数是根据联合概率分布的分解式构造的，它们的乘积恰好等于原联合概率分布，因此已经满足归一化条件。此时，可以认为配分函数为 $\mathcal{Z} = 1$。

在有向图模型中，条件分布的归一化是指：在给定条件变量的取值后，对相应变量的所有可能取值求和的结果等于 1，即（以离散变量为例）：

$$\sum_{x_i} p(x_i|\text{pa}(x_i)) = 1 \tag{9.31}$$

考虑由三个变量构成的简单有向链式结构 $x_1 \to x_2 \to x_3$，其联合分布可以分解为：

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

我们按照从末端节点到起始节点的顺序（即按编号较大到编号较小的次序），依次对 x_3、x_2

和 x_1 求和：

$$\begin{aligned}
&\sum_{x_1,x_2,x_3} p(x_1)p(x_2|x_1)p(x_3|x_2) \\
=&\sum_{x_1,x_2} p(x_1)p(x_2|x_1)\underbrace{\sum_{x_3} p(x_3|x_2)}_{=1} \\
=&\sum_{x_1} p(x_1)\underbrace{\sum_{x_2} p(x_2|x_1)}_{=1} \\
=&\sum_{x_1} p(x_1) = 1
\end{aligned}$$

这种逐步消去的过程表明，有向无环图可以利用各局部条件分布的归一化性质，保证联合分布的归一化。任何非空的有向无环图都存在没有子节点的节点（如以上例子中的 x_3），该节点所表示的变量只出现在自身所在的条件分布中，因此可以先对它求和，使该因子变为 1。移除该节点后，对剩余图重复这一过程，最终即可得到归一化结果。

对于无向图，考虑类似的链式连接 $x_1 - x_2 - x_3$，其联合分布通常可以分解为如下形式：

$$p(x_1, x_2, x_3) = \frac{1}{\mathcal{Z}}\psi_{12}(x_1, x_2)\psi_{23}(x_2, x_3)$$

对上式先关于 x_3 的部分求和，可得：

$$\sum_{x_3} \psi_{23}(x_2, x_3)$$

由于势函数不必是条件概率分布，求和后的结果一般仍是依赖于 x_2 的函数，并不一定等于 1。因此，不能像有向无环图模型那样，利用局部条件分布的归一化性质，通过逐步求和将各因子化为 1，而需要借助配分函数 $\mathcal{Z}$ 保证联合概率分布的整体归一化。需要注意的是，即使每个势函数对其所涉及变量的所有可能取值求和都等于 1，也不能保证这些势函数的乘积已经归一化。这是因为不同势函数可能包含共同的变量，使得对乘积的求和不能拆分为对各势函数分别求和后再相乘。

由于公式9.29中的势函数 $\psi_C(\boldsymbol{x}_C)$ 要求严格为正，因此通常可以将其表示为如下指数形式：

$$\psi_C(\boldsymbol{x}_C) = \exp(-E(\boldsymbol{x}_C)) \tag{9.32}$$

其中 $E(\boldsymbol{x}_C)$ 称为极大团 C 对应的**能量函数**（Energy Function）。由于联合概率分布由各势函数的乘积经归一化得到，而指数函数相乘相当于指数部分相加，因此整个系统的总能量可以表示为各极大团对应能量的总和。

与有向图模型中联合概率分布的各因子不同，无向图模型中的势函数通常没有直接的概率解释。除了要求严格正性外，势函数本身不必满足概率归一化条件（将通过引入配分函数 $\mathcal{Z}$ 对其乘积进行整体归一化）。因此，选择势函数时具有更大的灵活性。与此同时，这也带来了一个问题：在具体应用中，应当依据什么原则选择合适的势函数？一种直观的理解方式是将势函数看作对局部变量不同取值配置的偏好程度的刻画。某种局部配置对应的势函数值越大，就表示

该势函数越偏好这一配置；相应地，该配置对应的能量值也越低。从整体来看，具有较高概率的全局配置，是那些能够较好地兼顾各个团势函数所表达偏好的配置。由于不同团的偏好有时可能相互冲突，一个高概率的全局配置不一定能同时使每个团的势函数都达到最大值，但其对应的总能量是相对较低的。

我们已经在第7.2节介绍受限玻尔兹曼机时见过类似的能量函数。当时，我们通过优化受限玻尔兹曼机的参数，使得模型生成实际观测数据（即输入样本）的概率最大化。从能量模型的角度来看，这相当于将实际观测到的数据映射为系统能量较低的状态。



(a) (b)

图 9.10: 基于马尔可夫随机场的图像去噪示例。(a) 含有随机白色噪点的图像。(b) 去噪后的图像。此例子改编自文献 [10]。

我们通过一个简单的二值图像去噪的例子来说明无向概率图模型的应用 [8, 9, 27]。图9.10(a)是一张含有随机白色噪点的二值图像，其中每个像素非黑即白，可以采用 $+1$ 和 -1 编码这两种颜色。原始图像以纯黑色为背景，包含白色的“永和九年”四个汉字。我们的目标是尽可能恢复出原始的无噪声图像。

设观测到的含噪声图像由一组二值变量 $y_i \in \{-1, +1\}$ 表示，其中 i 表示图中像素的编号。相应地，我们用 $x_i \in \{-1, +1\}$ 表示未知的无噪声图像中第 i 个像素的取值。假设受噪声影响的像素仅占较小比例，则 x_i 与 y_i 通常具有相同的取值。此外，在无噪声图像中，相邻像素 x_i 与 x_j 通常也倾向于具有相同的取值。这些先验知识可以通过图9.11所示的马尔可夫随机场模型来表达。该无向概率图中的极大团分为两类，且每个极大团均由两个变量组成。第一类极大团为 $\{x_i, y_i\}$，其对应的能量函数用于刻画第 i 个未知无噪像素与已知含噪像素之间的一致性。对于这类团，我们选择一个简单的能量函数 $-\eta x_i y_i$，其中 $\eta \geq 0$ 为常数（超参数），用于控制该一致性约束在总能量中的权重。当 $\eta > 0$ 时，若 x_i 与 y_i 同号，则该能量项为 $-\eta$；若二者异号，则该能量项为 $+\eta$。因此，这一能量项通过赋予同号设置更低的能量和更大的势函数值，鼓励 x_i 与 y_i 取值一致。另一类极大团由相邻像素对 $\{x_i, x_j\}$ 构成。同样地，我们倾向于让相邻像素取相同的值，因此也选择类似的能量函数 $-\beta x_i x_j$，其中 $\beta \geq 0$ 为常数，用于控制相邻像素一致性约束的权重。当 $\beta > 0$ 时，取值相同时的能量低于取值不同时的能量。

如果希望模型倾向于让 x_i 取某一特定值，还可以为每个像素加入能量项 αx_i。当 $\alpha > 0$ 时，$x_i = -1$ 对应的能量更低，因此该项偏好 x_i 取值 -1；当 $\alpha < 0$ 时，该项则偏好 $x_i = +1$；而当 $\alpha = 0$ 时，该项不对 x_i 的两种取值施加偏好。至此，该模型完整的能量函数可以表示为：

$$E(\boldsymbol{x}, \boldsymbol{y}) = \alpha \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j - \eta \sum_i x_i y_i \tag{9.33}$$

以上能量函数所定义的关于 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 的联合概率分布为：

$$p(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{\mathcal{Z}} \exp\left(-E(\boldsymbol{x}, \boldsymbol{y})\right) \tag{9.34}$$

其中配分函数 $\mathcal{Z}$ 保证联合概率分布归一化。

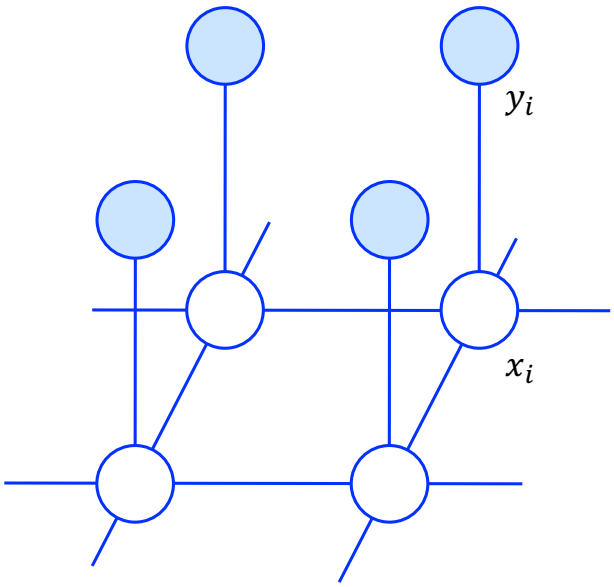


图 9.11: 用于图像去噪的马尔可夫随机场的无向图表示。其中 $x_i \in \{-1, +1\}$ 为二值随机变量，表示未知无噪声图像中第 i 个像素的取值，而 $y_i \in \{-1, +1\}$ 表示已观测的含噪图像中对应像素的取值。带背景色的圆圈表示已观测变量，而不带背景色的圆圈则表示未观测随机变量。无向概率图模型比较适合描述类似变量之间的空间依赖关系。

为了求得使公式9.33所示能量函数最小的 $\boldsymbol{x}$，我们首先将 $\boldsymbol{y}$ 中的各个变量固定为含噪图像中实际观测到的像素值。在给定 $\boldsymbol{y}$ 的条件下，最小化能量函数的值等价于最大化公式9.34所示的联合概率，也等价于最大化条件概率 $p(\boldsymbol{x}|\boldsymbol{y})$。因此，这一过程实际上是在求无噪声图像 $\boldsymbol{x}$ 的最大后验估计。

我们可以采用一种简单的迭代条件模态（Iterated Conditional Modes）算法 [45] 来进行求解。其基本思想是：首先将各变量初始化为 $x_i = y_i$。随后，每次选取一个变量 x_j，在保持其他所有变量取值不变的情况下，分别计算 $x_j = +1$ 和 $x_j = -1$ 两种状态对应的总能量，并将 x_j 设置为能量较低的状态。若两种状态对应的能量相等，则保持 x_j 的当前取值不变。由于在给定 $\boldsymbol{y}$ 时，能量越低，条件概率 $p(\boldsymbol{x}|\boldsymbol{y})$ 越高，因此上述更新要么保持当前设置及其概率不变，要么可降低能量并增大对应的概率。此外，每次更新只会改变包含 x_j 的能量项，因此只需进行局部计算，无需重新计算整个图像的总能量。接下来，对其他节点执行同样的更新，并不断重复这一过程，直到满足适当的停止条件。变量 x_j 可以按固定顺序更新，也可以随机选择，但应保证所有变量都有机会被反复访问。如果在一轮更新中，每个变量都至少被访问一次，且所有变量的取值均未发生变化，则说明单独改变任何一个变量的取值都无法进一步降低能量。此时，算法已达到局部最优值，即条件概率 $p(\boldsymbol{x}|\boldsymbol{y})$ 的一个局部最大值，但该迭代算法不保证取得全局最大值。

采用上述迭代条件模态算法，并且将公式9.33中的超参数分别设为 $\alpha=0$、$\beta=1$ 和 $\eta=2$。其中 $\alpha=0$ 表示 x_i 取 -1 和 $+1$ 的先验概率相同。以含噪图像作为初始值，运行算法直至收敛，即可得到类似图9.10(b) 所示的去噪后的图像。

9.5 精确与近似推断

前面我们介绍了如何利用概率图模型描述随机变量之间的依赖关系，并将复杂的联合概率分布分解为若干局部概率分布（有向图模型），或若干势函数的乘积经归一化后的形式（无向图模型）。然而，借助图结构建立概率模型通常并不是最终目的。我们往往还希望在给定模型及其部分变量的观测值之后，计算其他变量的概率分布或最可能的取值。这类根据给定模型和观测信息，计算未观测变量的概率分布或最可能取值的过程，统称为**推断**（Inference）。

例如，在前述的图像去噪问题中，给定含噪图像 $\boldsymbol{y}$ 之后，计算第 i 个像素取值为 $+1$ 的概率 $p(x_i=+1|\boldsymbol{y})$ 就是一个推断问题。计算单个变量或一组变量的边缘概率分布的问题，通常称为**边缘推断**（Marginal Inference）。当给定观测信息时，所计算的就是相应变量的后验边缘分布。推断还包括寻找最可能的变量取值组合。例如，求取使后验概率 $p(\boldsymbol{x}|\boldsymbol{y})$ 最大的无噪声图像 $\boldsymbol{x}$ 属于**最大后验**（Maximum A Posteriori, MAP）推断。当优化目标包含模型中的所有未观测变量时，这一任务通常称为**最可能解释**（Most Probable Explanation, MPE）推断。若只关心其中一部分未观测变量，则通常需要先对其余未观测变量进行边缘化，再寻找目标变量后验概率最大的取值组合。尽管这些推断问题的具体目标有所不同，但它们的共同点都是：利用模型所定义的联合概率分布以及已有的观测信息，对未观测变量的概率分布或可能取值进行计算和确定。

从计算上看，求取边缘分布或条件分布，通常需要对目标变量之外的未观测变量进行求和或积分。假设模型中的变量分为以下三组：目标变量 $\boldsymbol{x}$、已观测变量 $\boldsymbol{y}$ 以及其余未观测变量 $\boldsymbol{z}$。对于离散变量，在 $p(\boldsymbol{y})>0$ 时，我们有：

$$p(\boldsymbol{x}|\boldsymbol{y})=\frac{\sum_{\boldsymbol{z}}p(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})}{\sum_{\boldsymbol{x}}\sum_{\boldsymbol{z}}p(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})} \tag{9.35}$$

对于连续型随机变量，则将相应的求和替换为积分。上述公式虽然看似简洁，但直接穷举变量的所有可能取值进行求和计算可能十分困难。例如，若 $\boldsymbol{z}=(z_1,\ldots,z_M)^{\top}$ 包含 M 个二值变量，则直接求和需要遍历 2^M 种可能的取值组合。因此，即使联合分布能够表示为简单的局部因子的乘积，推断所需要的计算量仍可能很大。利用图结构及其所表达的分解形式来简化计算过程也是研究概率图模型的另一个重要目的。

根据最终得到的结果是否与理论上的真实概率完全一致，推断方法通常可以分为**精确推断**（Exact Inference）和**近似推断**（Approximate Inference）。精确推断是指在有限计算过程中得到理论上准确的边缘概率、条件概率或最优取值，这里不考虑计算机运算中的数值舍入误差。需要注意的是，“精确”是相对于给定模型而言的，并不意味着模型本身与真实的数据分布完全一致。对于可能取值数量有限的离散变量，若模型具有链式或树形的结构，通常可以将全局计算组织为一系列局部计算，从而高效地得到精确结果（详见本节后续的和积算法）。

当精确推断难以完成，或其计算时间和存储开销超出可接受范围时，我们通常采用计算上可行的方法来逼近精确的推断结果，也就是近似推断。例如，前面所介绍的迭代条件模态算法通过局部更新寻找后验概率较高的变量取值。在能量相等时保持当前变量取值不变的规则下，对于有限离散状态空间，每次发生状态改变都会使能量严格降低，因此算法经过有限次更新之后必然收敛。然而，迭代条件模态算法通常只能保证收敛到相对于单变量更新而言的局部最优解，而不能保证得到后验概率的全局最大值。因此，该算法可以看作一种近似求解最大后验（MAP）估计的方法。

近似推断中常用的方法主要包括**采样方法**和**变分推断**。采样方法通过直接或间接地从目标分布中生成样本，或者从适当的辅助分布中抽取样本并进行相应的加权或校正，利用样本的统计特性来估计所关心的概率或期望等，从而避免对所有可能的变量取值组合进行穷举而带来的计算上的不可行性。变分推断则是在一族便于处理的分布（如高斯分布）中，通过优化衡量分布差异的目标函数，寻找尽可能接近目标分布的近似分布，并利用该近似分布完成所需的计算。近似推断将在第12章中详细介绍，本节主要讨论精确推断方法。

推断与参数学习密切相关，但二者关注的问题有所不同。参数学习主要根据训练数据确定模型参数，而本节讨论的推断是在给定模型结构及其参数的条件下，计算目标变量的概率分布或最可能的取值。当模型包含隐变量时，学习过程往往需要先根据当前参数进行推断，计算隐变量的后验分布或相关期望，再利用这些结果更新模型参数。因此，推断有时也是学习算法的重要组成部分。

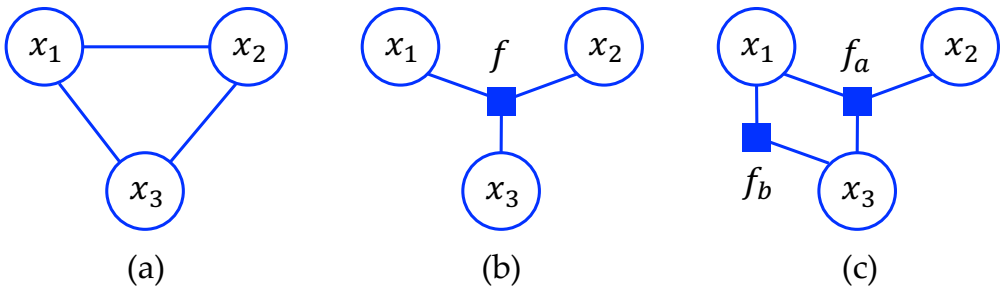


图 9.12: 无向图与因子图的对应关系示例。(a) 包含一个极大团 $\{x_1, x_2, x_3\}$ 及其对应势函数 $\psi(x_1, x_2, x_3)$ 的无向图。(b) 包含一个因子 $f(x_1, x_2, x_3) = \psi(x_1, x_2, x_3)$ 的因子图来表示与图 (a) 相同的联合概率分布。(c) 表示同一联合概率分布的另一种因子图，其中势函数被进一步分解为 $\psi(x_1, x_2, x_3) = f_a(x_1, x_2, x_3) f_b(x_1, x_3)$。

在进一步讨论概率图模型的推断方法之前，我们先将有向图模型和无向图模型统一表示为**因子图**（Factor Graph），以便在统一的框架下介绍适用于这两类模型的推断方法。相较于有向图或无向图，因子图能够更清晰和显式地表示联合概率分布分解中的各项因子，以及每个因子所涉及的变量（如图9.12(b) 所示）。在因子图中，除了以圆圈表示的变量节点之外，还引入了以小方块表示的因子节点。根据前面的讨论，有向概率图模型的联合概率分布可以分解为各节点对应的条件分布或边缘分布的乘积。在相应的因子图中，为每个条件分布或边缘分布设置一个因子节点，并用无向边将其与该分布涉及的所有变量节点相连。对于无向概率图模型，其联合

概率分布可以表示为若干势函数的乘积经归一化后的形式。在相应的因子图中，为每个势函数设置一个因子节点，并用无向边将其与该势函数涉及的所有变量节点相连。配分函数作为整体归一化常数，通常不在因子图中单独表示。

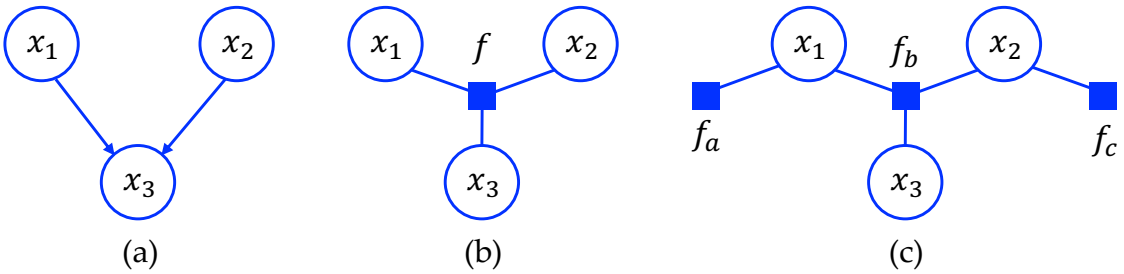


图 9.13: 有向图模型与因子图的对应关系示例。(a) 有向图所表示联合概率分布可以分解为 $p(x_1, x_2, x_3) = p(x_1)p(x_2)p(x_3|x_1, x_2)$。(b) 表示同一联合概率分布的因子图，其中引入的因子为 $f(x_1, x_2, x_3) = p(x_1)p(x_2)p(x_3|x_1, x_2)$。(c) 同一联合概率分布的另一因子图表示，其中三个因子分别为：$f_a(x_1) = p(x_1)$、$f_c(x_2) = p(x_2)$ 和 $f_b(x_1, x_2, x_3) = p(x_3|x_1, x_2)$。

将无向图或有向图模型转换为相应因子图的具体过程如下。对于通过无向图表示的联合概率分布，首先为原图中的各个节点创建相应的变量节点（用圆圈表示），然后为各个极大团创建对应的因子节点（用小方块表示），并令相应因子等于该极大团的势函数。最后，将每个因子节点与该因子所涉及的所有变量节点用无向边相连。需要注意的是，通过进一步分解势函数或合并因子，同一联合概率分布可以得到不同的因子图表示。图9.12展示了同一无向图模型对应两种不同因子图的例子。类似地，将有向图模型转换为因子图时，首先为原图中的各个节点创建相应的变量节点，然后为各个条件概率分布或边缘分布创建对应的因子节点。最后，将每个因子节点与对应分布所涉及的所有变量节点用无向边相连。同样地，通过合并因子或在可能的情况下进一步分解因子，同一有向图模型也可以得到不同的因子图表示（如图9.13所示）。上述例子说明，同一有向图模型或无向图模型所表示的联合概率分布，可以具有不同的因子分解形式。因此，因子图通过引入因子节点显式地表示各个分解因子及其所涉及的变量，能够比有向图或无向图更具体地表达联合概率分布所采用的分解形式。

上述讨论表明，有向图和无向图都可以转换为相应的因子图，且因子图对于联合概率分布的分解形式表达得更为清晰和具体。因此，本节后续的讨论都将基于因子图展开。我们将依次介绍如何利用因子图计算变量的边缘分布、给定观测信息后的条件分布，以及最可能的变量取值组合。为了便于理解，我们从图9.14 (a) 所示的例子入手，然后从中引出**和积算法**（The Sum-product Algorithm）和**最大和算法**（The Max-sum Algorithm）。其中和积算法可用于计算单个或部分随机变量的边缘概率。在给定观测信息之后，也可用于计算相应的后验边缘分布。最大和算法则用于寻找让全局联合概率最大的变量状态组合。在给定观测信息之后，其目标是寻找使后验概率最大的未观测变量的取值组合。

接下来，我们先讨论树形因子图上的推断过程，再讨论如何将相应方法推广到含环因子图。所谓树形结构，是指图中任意两个不同节点之间仅存在一条简单路径。因此，删除树形因子图

中的任意一条边，都会将其分成两个互不连通的子图。如图9.14 (a) 所示因子图对应的联合概率分布可分解为：

$$p(\boldsymbol{x}) = p(x_1, x_2, x_3, x_4, x_5) = f_a(x_1) f_b(x_1, x_2, x_3) f_c(x_2) f_d(x_4) f_e(x_3, x_4, x_5) \tag{9.36}$$

为简化表述，我们先假设各因子的乘积已经归一化，且图中所有变量均为离散变量。若要计算变量 x_3 的边缘分布，就需要对除 x_3 之外的所有变量求和：

$$p(x_3) = \sum_{\boldsymbol{x} \setminus x_3} p(\boldsymbol{x}) = \sum_{x_1} \sum_{x_2} \sum_{x_4} \sum_{x_5} f_a(x_1) f_b(x_1, x_2, x_3) f_c(x_2) f_d(x_4) f_e(x_3, x_4, x_5) \tag{9.37}$$

其中 $\boldsymbol{x} \setminus x_3$ 表示除 x_3 之外的所有变量 $\{x_1, x_2, x_4, x_5\}$，求和遍历这些变量的所有联合取值组合。若各因子的乘积尚未归一化，则可以先按上述过程计算未归一化的边缘函数，再对其进行归一化后得到 $p(x_3)$。

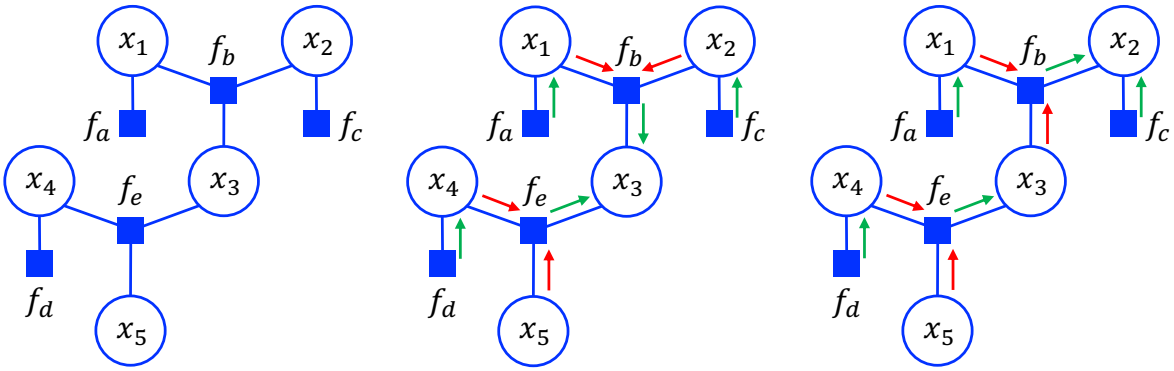


图 9.14: 树形因子图上的和积算法示例。(a) 由变量节点（圆圈）和因子节点（小方块）构成的树形因子图。(b) 以 x_3 为目标节点，消息从各叶节点逐步向 x_3 汇集，用于计算其边缘分布 $p(x_3)$。(c) 以 x_2 为目标节点的消息传递过程，用于计算边缘分布 $p(x_2)$。红色箭头表示变量节点向因子节点传递消息，绿色箭头表示因子节点向变量节点传递消息。

根据公式9.37，利用乘法对加法的分配律，并将与求和变量无关的因子移至相应求和符号之外，可得：

$$\begin{aligned} p(x_3) &= \sum_{x_1} \sum_{x_2} \sum_{x_4} \sum_{x_5} f_a(x_1) f_b(x_1, x_2, x_3) f_c(x_2) f_d(x_4) f_e(x_3, x_4, x_5) \\ &= \underbrace{\left(\sum_{x_1} \sum_{x_2} f_a(x_1) f_b(x_1, x_2, x_3) f_c(x_2) \right)}_{m_{f_b \to x_3}(x_3)} \underbrace{\left(\sum_{x_4} \sum_{x_5} f_d(x_4) f_e(x_3, x_4, x_5) \right)}_{m_{f_e \to x_3}(x_3)} \end{aligned} \tag{9.38}$$

从公式9.38可以看出，对于每个给定 x_3 的取值，两个括号内的求和分别涉及互不重叠的变量集合 $\{x_1, x_2\}$ 与 $\{x_4, x_5\}$。因此，可以分别计算，再将结果相乘。这就将原本涉及四个变量的联合求和，转化为两个分别涉及两个变量的局部求和。上述分解也可以从图结构的角度加以理解：将变量节点 x_3 及其相连的边从树形因子图中删除后，原图分成两个互不连通的子图。其中一个子图原先通过因子 f_b 与 x_3 相连，另一个子图原先通过因子 f_e 与 x_3 相连。在计算时，我们仍

然保留这两个因子对 x_3 的依赖，并分别对各子图中的变量求和，得到两个仅依赖于 x_3 的函数。我们将这两个函数分别记为 $m_{f_b\to x_3}(x_3)$ 和 $m_{f_e\to x_3}(x_3)$。对于变量节点 x_3 而言，它们可以看作分别由相邻因子节点 f_b 和 f_e 传递过来的消息（Message）。每条消息汇总了相应子图中的因子在各个 x_3 取值下的计算结果，而两条消息的乘积就给出了 x_3 的边缘分布。将这一过程推广到一般的树形因子图，某个变量 x_i 的边缘分布等于其所有相邻因子节点传入消息的乘积，即：

$$p(x_i)=\prod_{f_s\in\mathcal{S}(x_i)} m_{f_s\to x_i}(x_i) \tag{9.39}$$

其中 $\mathcal{S}(x_i)$ 表示与变量节点 x_i 相邻的因子节点集合。需要注意的是，从因子节点传递到变量节点的消息并非单个数值，而是关于该变量取值的函数。对于变量的每一种可能取值，消息都有一个对应的函数值。因此，可以将消息理解为相应子图对该变量不同取值的相对支持程度。将传入同一变量节点的所有消息相乘并归一化后，就得到该节点的信念（Belief），即对该变量边缘概率分布的估计。

我们进一步对公式9.38中的第一个乘积项进行分解（第二个乘积项也可进行类似分解）：

$$\begin{aligned} m_{f_b\to x_3}(x_3) &= \sum_{x_1}\sum_{x_2} f_a(x_1)f_b(x_1,x_2,x_3)f_c(x_2) \\ &= \sum_{x_1}\sum_{x_2} f_b(x_1,x_2,x_3)\underbrace{f_a(x_1)}_{m_{x_1\to f_b}(x_1)}\underbrace{f_c(x_2)}_{m_{x_2\to f_b}(x_2)} \qquad (9.40)\\ &= \sum_{x_1}\sum_{x_2} f_b(x_1,x_2,x_3)\prod_{x_k\in\mathcal{S}(f_b)\setminus x_3} m_{x_k\to f_b}(x_k) \qquad (9.41)\end{aligned}$$

其中 $\mathcal{S}(f_b)$ 表示与因子节点 f_b 相邻的变量节点集合，而 $\mathcal{S}(f_b)\setminus x_3$ 则表示从该集合中去除目标变量节点 x_3 后得到的集合。计算从 f_b 发送给 x_3 的消息时，只会使用其他相邻变量节点传入 f_b 的消息，而并不包含来自 x_3 的消息。这是因为当前计算的是由因子节点 f_b 传递给目标变量节点 x_3 的消息（它只汇总 f_b 一侧子图的信息），然后将在目标变量节点 x_3 处采用公式9.39来整合与汇总来自各个方向的消息。虽然因子图中的边是无向的，但沿边传递消息是有方向的。对于树形因子图，在计算某个目标变量的边缘分布时，可以将该变量节点视为树形因子图的根节点，使所需的消息从叶节点出发，逐步向根节点汇集。本例希望计算 x_3 的边缘分布，因此以 x_3 为根节点，其相应的消息传递方向如图9.14(b) 所示，其中红色箭头表示变量节点向因子节点传递消息，而绿色箭头表示因子节点向变量节点传递消息。

公式9.41给出了因子节点向变量节点传递消息的计算过程：首先，将除接收节点之外的所有相邻变量节点传入的消息相乘；然后，乘以该因子节点对应的因子函数；最后，对除接收节点所对应变量之外的所有相邻变量进行边缘化。在本例中，就是先将 $m_{x_1\to f_b}(x_1)$ 和 $m_{x_2\to f_b}(x_2)$ 相乘（即计算公式9.41中的连乘项），再乘以因子函数 $f_b(x_1,x_2,x_3)$，最后对 x_1 和 x_2 的所有可能取值求和，从而得到仅依赖于 x_3 的消息 $m_{f_b\to x_3}(x_3)$。需要注意的是，在树形因子图的消息传递过程中，因子节点只有在收到除接收节点之外的所有相邻变量节点传入的消息之后，才能向接收节点发送消息。

我们已经介绍了从因子节点向变量节点传递消息的计算方法。另一类从变量节点向因子节

点传递的消息的计算方式与公式9.39类似，但在连乘时需要排除来自接收因子节点的消息，其具体计算公式如下：

$$m_{x_i \to f_j}(x_i) = \prod_{f_s \in \mathcal{S}(x_i) \setminus f_j} m_{f_s \to x_i}(x_i) \tag{9.42}$$

$\mathcal{S}(x_i)$ 表示与变量节点 x_i 相邻的因子节点集合，$\mathcal{S}(x_i) \setminus f_j$ 表示从该集合中去除接收因子节点 f_j 后得到的集合。也就是说，变量节点 x_i 将除 f_j 之外的所有相邻因子节点传入的消息相乘，并将结果作为发送给 f_j 的消息。相应地，从因子节点向变量节点传递消息的一般公式为：

$$m_{f_j \to x_i}(x_i) = \sum_{\boldsymbol{x} \setminus x_i} f_j(\boldsymbol{x}) \prod_{x_k \in \mathcal{S}(f_j) \setminus x_i} m_{x_k \to f_j}(x_k) \tag{9.43}$$

其中 $\mathcal{S}(f_j)$ 表示与因子节点 f_j 相邻的变量节点集合。$\boldsymbol{x}$ 表示集合 $\mathcal{S}(f_j)$ 中随机变量组成的向量。也就是说，因子节点 f_j 将除 x_i 之外的所有相邻变量节点传入的消息相乘，再乘以自身对应的因子函数，最后对除 x_i 之外的所有相邻变量求和，从而得到发送给 x_i 的消息。

至此，我们已经介绍了和积算法所需的预备知识和相关定义，下面完整给出利用和积算法计算树形因子图中任一随机变量边缘分布的过程。首先，将待计算边缘分布的目标随机变量节点作为根节点，然后从叶节点开始，逐步向根节点传递消息。叶节点初始消息的计算方法如图9.15所示。对于非叶节点，只有在收到除接收节点之外的所有相邻节点传入的消息之后，才能向接收节点发送消息。根据发送节点和接收节点的类型，分别采用公式9.42计算变量节点向因子节点传递的消息，或采用公式9.43计算因子节点向变量节点传递的消息。当根节点收到所有相邻因子节点传入的消息之后，按照公式9.39将这些消息相乘，并在需要时进行归一化，即可得到目标变量的精确边缘分布。如图9.14(b) 所示，利用和积算法计算变量 x_3 边缘分布的具体步骤如下：

$$m_{f_a \to x_1}(x_1) = f_a(x_1)$$

$$m_{f_c \to x_2}(x_2) = f_c(x_2)$$

$$m_{x_1 \to f_b}(x_1) = m_{f_a \to x_1}(x_1) = f_a(x_1)$$

$$m_{x_2 \to f_b}(x_2) = m_{f_c \to x_2}(x_2) = f_c(x_2)$$

$$m_{f_b \to x_3}(x_3) = \sum_{x_1} \sum_{x_2} f_b(x_1, x_2, x_3) m_{x_1 \to f_b}(x_1) m_{x_2 \to f_b}(x_2) = \sum_{x_1} \sum_{x_2} f_b(x_1, x_2, x_3) f_a(x_1) f_c(x_2)$$

$$m_{f_d \to x_4}(x_4) = f_d(x_4)$$

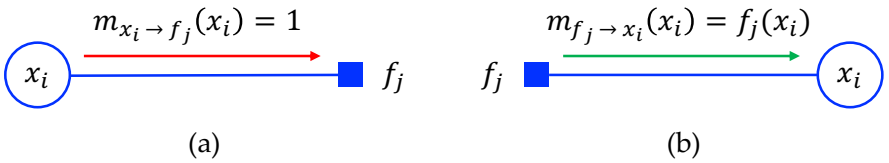
$$m_{x_5 \to f_e}(x_5) = 1$$

$$m_{x_4 \to f_e}(x_4) = m_{f_d \to x_4}(x_4) = f_d(x_4)$$

$$m_{f_e \to x_3}(x_3) = \sum_{x_4} \sum_{x_5} f_e(x_3, x_4, x_5) m_{f_d \to x_4}(x_4) = \sum_{x_4} \sum_{x_5} f_e(x_3, x_4, x_5) f_d(x_4)$$

$$\begin{aligned} p(x_3) &= m_{f_b \to x_3}(x_3) m_{f_e \to x_3}(x_3) \\ &= \left(\sum_{x_1} \sum_{x_2} f_b(x_1, x_2, x_3) f_a(x_1) f_c(x_2) \right) \left(\sum_{x_4} \sum_{x_5} f_e(x_3, x_4, x_5) f_d(x_4) \right) \end{aligned}$$

上述计算过程最终得到的 $p(x_3)$ 与公式9.38的结果一致，验证了本例中利用消息传递方法计算变量 x_3 边缘分布的正确性。



(a) (b)

图 9.15: 和积算法中叶节点的初始消息计算。(a) 叶节点为变量节点时，向其唯一相邻因子节点发送恒为 1 的消息。(b) 叶节点为因子节点时，向其唯一相邻变量节点发送该因子函数作为消息。

对于相同的因子图，若改为计算变量 x_2 的边缘分布，则以 x_2 为根节点，相应的消息传递过程如图9.14(c) 所示。利用和积算法进行计算的具体步骤如下：

$$
\begin{aligned}
m_{f_d \to x_4}(x_4) &= f_d(x_4) \\
m_{x_4 \to f_e}(x_4) &= m_{f_d \to x_4}(x_4) = f_d(x_4) \\
m_{x_5 \to f_e}(x_5) &= 1 \\
m_{f_e \to x_3}(x_3) &= \sum_{x_4}\sum_{x_5} f_e(x_3, x_4, x_5) m_{x_4 \to f_e}(x_4) m_{x_5 \to f_e}(x_5) = \sum_{x_4}\sum_{x_5} f_e(x_3, x_4, x_5) f_d(x_4) \\
m_{x_3 \to f_b}(x_3) &= m_{f_e \to x_3}(x_3) = \sum_{x_4}\sum_{x_5} f_e(x_3, x_4, x_5) f_d(x_4) \\
m_{f_a \to x_1}(x_1) &= f_a(x_1) \\
m_{x_1 \to f_b}(x_1) &= m_{f_a \to x_1}(x_1) = f_a(x_1) \\
m_{f_b \to x_2}(x_2) &= \sum_{x_1}\sum_{x_3} f_b(x_1, x_2, x_3) m_{x_1 \to f_b}(x_1) m_{x_3 \to f_b}(x_3) \\
&= \sum_{x_1}\sum_{x_3} \left(f_b(x_1, x_2, x_3) f_a(x_1) \sum_{x_4}\sum_{x_5} f_e(x_3, x_4, x_5) f_d(x_4) \right) \\
m_{f_c \to x_2}(x_2) &= f_c(x_2) \\
p(x_2) &= m_{f_c \to x_2}(x_2) m_{f_b \to x_2}(x_2) \\
&= f_c(x_2) \sum_{x_1}\sum_{x_3} \left(f_b(x_1, x_2, x_3) f_a(x_1) \sum_{x_4}\sum_{x_5} f_e(x_3, x_4, x_5) f_d(x_4) \right) \\
&= \sum_{x_1}\sum_{x_3}\sum_{x_4}\sum_{x_5} f_a(x_1) f_b(x_1, x_2, x_3) f_c(x_2) f_d(x_4) f_e(x_3, x_4, x_5)
\end{aligned}
$$

上述计算最终得到的结果与直接对联合分布进行边缘化所得到的 $p(x_2)$ 一致。

现在假设我们希望计算树形因子图中每个变量节点的边缘分布。最直接的方法是依次将每个变量节点设为根节点，并分别重新运行一次前述算法。然而，这样做会重复计算许多相同的消息，从而导致计算效率较低。我们通过保存并复用已经计算的消息，可以显著减少重复计算。具体而言，我们可以任意选择一个变量节点作为根节点。与前述过程一样，首先从叶节点开始，

将消息逐步向根节点传递。当这一过程完成后，根节点已经收到所有相邻因子节点传入的消息，因此可以进一步向这些因子节点发送消息。计算发往某个相邻节点的消息时，仍然需要排除该接收节点传入的消息（按公式9.42）。随后，根节点的相邻因子节点结合刚收到的消息与此前保存的其他消息，继续沿远离根节点的方向（即按之前的反方向）传递消息（按公式9.43）。如此逐层向外传递，直至到达所有叶节点。经过上述由叶节点向根节点、再由根节点向叶节点的两轮消息传递后，每条边上都已有消息沿两个方向各传递一次，每个节点也都收到了所有相邻节点传入的消息。此时，对任一变量节点，将其所有相邻因子节点传入的消息相乘，并在需要时进行归一化，就可以得到该变量的精确边缘分布。上述过程的核心思想是：在树形因子图上，只需沿每条边的两个方向各传递一次消息，就能够计算所有变量节点的边缘分布，而不必针对每个变量重复执行整套推断所需的计算过程。

设因子图中共有 E 条边，计算单个变量的边缘分布需要经过 E 条消息的计算与传播，则计算所有变量的边缘分布仅需要传递 $2E$ 条消息（每条边沿两个方向各传递一次）。因此，从消息数量来看，计算所有变量的边缘分布仅需将消息计算次数增加一倍。相比之下，若图中有 N 个变量节点，并针对每个变量分别完整运行一次前述算法，则共需计算 NE 条消息。需要注意的是，因子图同时包含变量节点和因子节点。若树形因子图中有 N 个变量节点和 M 个因子节点，则其边数为 $E = N + M - 1$。当边数 E 与变量节点数 N 成正比时，分别为每个变量运行算法所需的消息数量随 N 呈二次增长，而复用消息的方法所需的消息数量仅随 N 呈线性增长。

我们仍沿用图9.14(b) 所示的例子，并在开始时选择 x_3 作为根节点。当 x_3 收到所有相邻因子节点传入的消息后，就可以沿与此前相反的方向向各相邻因子节点发送消息（按公式9.42）。这些因子节点收到来自 x_3 的消息后，结合此前保存的其他传入消息，向远离根节点方向的相邻变量节点发送消息（按公式9.43）。这一过程逐步向外推进，直至到达所有叶节点。例如，当从变量节点 x_3 出发的消息经过传播后到达变量 x_2 时，我们就可按公式9.39来计算 x_2 的边缘分布（如图9.14(c) 所示）。需要注意的是，最终得到的边缘分布与根节点的选择无关。这里引入根节点只是为了更清楚地说明消息传递的顺序，并不意味着该节点在所有节点中具有特殊性。

在此前的讨论中，我们暂时不考虑归一化问题。下面将简要说明如何在必要时进行归一化。如果因子图由有向图模型转换得到，那么其联合概率分布本身已经归一化。因此，通过树形因子图上的和积算法得到的各个边缘分布也自然满足归一化条件。但若因子图由无向图模型转换而得，且计算时仅使用势函数的乘积，则通常还需要通过配分函数 $\mathcal{Z}$ 进行归一化。设和积算法得到的变量 x_i 的未归一化边缘函数为 $q(x_i)$，则归一化后的边缘分布为：

$$p(x_i) = \frac{1}{\mathcal{Z}} q(x_i) \tag{9.44}$$

计算这样的归一化常数 $\mathcal{Z}$ 其实十分简单，只需对任一变量 x_i 的所有可能取值求和，即：

$$\mathcal{Z} = \sum_{x_i} q(x_i) \tag{9.45}$$

因此，完成消息传递后，只需对单个变量所有可能取值求和，就可以确定归一化常数，无需再对所有变量的联合取值组合进行求和。

我们已经介绍了如何利用和积算法计算变量的边缘分布，接下来将讨论如何在部分变量已观测的情况下计算其他变量的条件分布。为此，我们将观测信息引入消息传递过程，并对最终结果进行归一化，从而得到给定观测信息后的后验边缘分布。下面仍然以离散变量构成的树形因子图为例进行说明。

计算后验边缘分布只需对前面介绍的和积算法作一个简单的改动：在已观测变量向相邻因子节点发送消息时，只保留与观测值对应的消息分量，将其余取值对应的消息分量全部设为 0。除此之外，消息的计算和传递仍然遵循前面介绍的规则。例如，假设 x_2 是一个二值变量，可能取值为 0 或 1。若按照原来的规则计算得到的消息在这两个取值处分别为 s_0 和 s_1。则当观测到 $x_2 = 1$ 时，实际发送的消息为：

$$m_{x_2 \to f_b}(0) = 0, \quad m_{x_2 \to f_b}(1) = s_1 \tag{9.46}$$

也就是说，原来的消息 (s_0, s_1) 变成了 $(0, s_1)$。需要注意的是，这里保留的是消息在观测值处的数值 s_1，而不是将该数值替换为 x_2 观测值 1。

这一简单改动之所以能够引入观测信息，是因为因子节点收到这条消息后，仍然按照原来的规则，将各条传入消息与因子函数相乘，再对相应变量求和。此时，所有包含 $x_2 = 0$ 的项都会因乘以 0 而消失，只有 $x_2 = 1$ 对应的项被保留下来。因此，对于 x_2 的求和实际上就变成了仅保留 $x_2 = 1$ 时的计算结果。观测信息由此被纳入后续消息，并继续传递到图中的其他节点。更一般地，若 x_i 的观测值为 $\hat{x}_i$，则在计算 x_i 向任一相邻因子节点发送的消息时，先按照原来的规则计算消息，再将其中所有 $x_i \neq \hat{x}_i$ 对应的分量设为 0。如果有多个已观测变量，就分别对这些变量执行同样的处理。

引入观测信息后，目标变量仍然通过汇总相邻因子节点传入的消息来计算分布。例如，若已知 $x_2 = \hat{x}_2$，希望计算 x_3 的条件分布，则将传入 x_3 的两条消息相乘，得到：

$$q(x_3) = m_{f_b \to x_3}(x_3) m_{f_e \to x_3}(x_3) \tag{9.47}$$

通过消息 $m_{f_b \to x_3}(x_3)$，$q(x_3)$ 已经包含了 $x_2 = \hat{x}_2$ 这一观测信息，但通常尚未归一化。在该观测事件的概率大于 0 时，我们对 $q(x_3)$ 进行归一化，即可得到所求的后验边缘分布：

$$p(x_3 | x_2 = \hat{x}_2) = \frac{q(x_3)}{\sum_{x_3} q(x_3)} \tag{9.48}$$

如果需要计算所有未观测变量的后验边缘分布，也可以沿用前述双向消息传递过程，并分别对各变量汇总消息后的结果进行归一化。

之前我们提到，消息可以理解为相应子图对某个变量不同取值的相对支持程度。而将传入同一变量节点的所有消息相乘并归一化后，就得到该变量的信念（Belief），用于表示模型对该变量不同取值的概率判断。通过节点之间的消息传递来形成和更新这些信念的过程，通常称为**信念传播**（Belief Propagation）。前面介绍的和积算法，就是用于计算边缘分布的信念传播算法在因子图上的具体形式。其中，消息是节点之间传递的信息，而信念是变量节点汇总传入消息后得到的结果。

此前讨论中，我们以树形因子图为前提。在树形结构中，任意两个节点之间只有一条简单

路径。因此，可以从叶节点开始安排消息的计算与传递顺序，并且只需沿每条边的两个方向各传递一次消息，就能够得到所有变量的精确边缘分布。但若因子图中存在环，情况则有所不同。一方面，图中可能没有叶节点，无法直接按照前述方式启动和完成消息传递；另一方面，一个节点传出的信息可能沿着环经过其他节点后，再次返回该节点。因此，从不同方向传入的消息可能包含重复的信息，不能再将它们理解为来自互不重叠的子图。即使每条边上的消息都已经双向传递过一次，也不能保证已经得到精确的边缘分布。

对于含环的因子图，可以在和积算法的基础上进行扩展，从而得到一种近似推断方法。其基本思想是：先为所有消息设置初始值，然后反复使用和积算法的局部消息计算规则更新消息，直至消息基本稳定或达到预设停止条件，以尝试得到近似的概率估计。这种方法称为**环路信念传播**（Loopy Belief Propagation）。

下面仍以离散变量构成的因子图为例，说明其基本过程。首先，为每条边上的两个方向分别设置初始消息。一种简单的做法是将每条消息的所有分量都设为 1，表示初始消息对变量的不同取值不施加额外偏好。如果某个变量已经观测到，则按照前面介绍的方法，将该变量发出的消息中与观测值不一致的分量设为 0。这样，即使图中没有叶节点，各节点也都有可供使用的初始消息，不必相互等待。随后，反复对消息进行更新。变量节点向因子节点发送消息时，仍采用公式9.42。因子节点向变量节点发送消息时，也仍采用公式9.43。计算发往某个节点的消息时，仍需排除该接收节点传入的消息。对于已观测变量，每次更新时都要继续保留观测值对应的分量，并将其余分量设为 0。

一种简单的更新方式是每轮都根据上一轮的消息计算所有新消息，待计算完成后，再统一替换旧消息。为减少数值过大或过小的问题，并便于比较不同轮次的消息，通常还会对每条消息进行归一化，使其各分量之和为 1。每完成一轮更新，就检查消息是否已经基本稳定。例如，可以比较相邻两轮中所有归一化消息的对应分量。如果这些分量的最大绝对变化量小于预先设定的阈值，就认为消息达到了所要求的收敛精度，并停止迭代计算过程。此外，通常还会设置最大的迭代次数，以避免算法因无法收敛而持续运行。如果达到这一预设的迭代上限时，消息仍未稳定，则也应停止计算。停止迭代之后，将传入每个未观测变量节点的消息相乘并归一化，就能够得到该变量的信念，用于近似其边缘分布。如果消息中已经引入观测信息，则得到的是相应后验边缘分布的近似。

需要强调和注意的是，环路信念传播是一种近似推断方法，既不保证收敛，也不保证结果准确。消息可能在不同数值之间持续振荡，无法达到稳定状态。即使消息已经收敛，得到的信念也可能与模型的真实边缘分布存在较大偏差，甚至给出较差的概率估计。因此，消息收敛只说明继续更新时变化已经很小，并不意味着已经得到正确或足够准确的推断结果。我们将在第12章进一步介绍其他常用的近似推断方法。

我们已经讨论了如何基于因子图计算变量的边缘分布，以及在给定部分变量取值时计算其他变量的后验边缘分布。接下来，我们讨论在给定模型及其参数的情况下，如何寻找使联合概率最大的变量取值组合。对于单个离散变量，寻找其最可能取值的方法比较直接：先计算该变

量的边缘分布，再比较各个取值对应的概率，选择概率最大的取值，即：

$$x_i^* = \arg\max_{x_i} p(x_i) \tag{9.49}$$

在树形因子图中，这一边缘分布可以通过前面介绍的和积算法精确计算。

表 9.2: 两个二值离散随机变量的联合概率分布示例。联合分布达到最大值时对应的变量取值与两个边缘分布各自达到最大值时对应的变量取值并不相同。表中用方框标出联合分布最大概率，用下划线标出两个边缘分布各自的最大概率。

$p(x_1, x_2)$	$x_2 = 0$	$x_2 = 1$	$p(x_1)$
$x_1 = 0$	0.3	0.0	0.3
$x_1 = 1$	0.3	$\boxed{0.4}$	$\underline{0.7}$
$p(x_2)$	$\underline{0.6}$	0.4	1.0

然而，分别选择各个随机变量在其边缘分布中最大概率的取值，再将这些取值组合起来，并不一定能使联合概率达到最大。以表9.2所示的两个二值离散随机变量的联合概率分布为例，当 $x_1 = 1$ 且 $x_2 = 1$ 时，联合概率 $p(x_1, x_2)$ 达到最大值 0.4（用方框标出）。我们对 x_1 的两种可能取值求和，可以得到 x_2 的边缘分布：$p(x_2 = 0) = 0.6$ 和 $p(x_2 = 1) = 0.4$。因此，x_2 在其边缘分布中最可能的取值为 0。同样地，变量 x_1 的边缘分布为 $p(x_1 = 0) = 0.3$ 和 $p(x_1 = 1) = 0.7$，因此，x_1 在其边缘分布中最可能的取值为 1。如果分别根据这两个边缘分布选择各自最可能的取值，就会得到 $x_1 = 1$ 和 $x_2 = 0$ 的组合。但这一取值组合对应的联合概率只有 0.3，小于联合概率的最大值 0.4。两种选择方式之所以会得到不同结果的原因是边缘概率与联合概率所衡量的对象不同。某个变量在给定取值处的边缘概率是将其他变量所有可能取值对应的联合概率相加得到的，而联合概率衡量的是某一个具体取值组合出现的可能性。因此，某个变量取值对应的边缘概率较大，并不意味着包含这一取值的某个具体组合也具有较大的联合概率。寻找使联合概率最大的变量取值组合，需要将各变量的取值作为一个整体加以考虑，其本质是一个组合优化问题。

我们可以在前面所介绍的和积算法的基础上进行扩展，从而得到一种用于寻找使联合概率达到最大值的变量取值组合的算法。下面仍以离散变量构成的树形因子图为例进行说明。对于变量节点向因子节点发送的消息 $m_{x_i \to f_j}(x_i)$，其计算形式保持不变，即将除接收因子节点 f_j 之外的所有相邻因子节点传入的消息相乘。计算结果仍然是关于 x_i 取值的函数。而对于因子节点向变量节点发送的消息 $m_{f_j \to x_i}(x_i)$，则需要作以下两处调整：

- 其一，将求和操作替换为取最大值。在和积算法中，我们将因子函数与其他相邻变量节点传入的消息相乘，再对除 x_i 之外的所有相邻变量求和，从而汇总这些变量所有可能取值组合的贡献。现在，我们希望寻找联合概率最大的取值组合，因此需要对 x_i 的每一种可能取值，比较其他相邻变量的所有取值组合对于 x_i 可能取值的贡献，并保留使上述乘积最大的结果。这个最大值就是消息 $m_{f_j \to x_i}(x_i)$ 在该 x_i 取值处的数值。此处，求和得到的是所有组合的总贡献，而取最大值得到的是最优组合对应的结果。

- 其二，在计算最大值的同时，还要记录使其达到最大值的变量取值组合。最大值只告诉我们最优结果有多大，并没有告诉我们哪些取值产生了这一结果。需要注意的是，此时我们尚不能确定 x_i 最终应取哪个值，因而需要为 x_i 的每一种可能取值分别保存相应的最优组合，记为 $\phi_{f_j \to x_i}(x_i)$。这里记录的是除 x_i 之外、与 f_j 相邻的所有变量的一组取值。如果存在多个并列最优的组合，保存其中任意一组即可。例如，如果 x_i 是二值变量，就需要分别记录：当 $x_i = 0$ 时，其他相邻变量的哪一组取值使上述乘积最大，以及当 $x_i = 1$ 时，哪组取值使上述乘积最大。等到 x_i 的最终取值确定后，就可以通过 $\phi_{f_j \to x_i}(x_i)$ 查出与之对应的最优组合。

与和积算法类似，我们可以任意选择一个变量节点作为根节点，并从叶节点开始，逐步向根节点传递消息。当根节点收到所有相邻因子节点传入的消息后，将这些消息相乘，就可以得到关于根节点取值的函数。对于根节点的每一种取值，该函数已经汇总了各个分支在这一取值下能够达到的最优结果。因此，选择使该函数最大的根节点取值，就能确定一个全局最优组合中根节点变量应取的值。随后，沿着与此前消息传递相反的方向进行回溯。首先，根据根节点已经确定的取值，查找各相邻因子节点保存的 $\phi_{f_j \to x_i}(x_i)$，确定这些因子节点所连接的其他变量的取值。再以这些已经确定的取值为依据，继续查找下一层因子节点保存的记录。如此逐步向外推进，直至到达所有叶节点，就能得到一组使联合概率达到最大值的变量取值组合。

在实践中，大量较小的概率值连续相乘可能导致数值下溢。因此，在寻找使联合概率最大的变量取值组合时，通常在对数域中进行计算，从而将连乘转换为相加。由于对数函数是严格单调递增的，这一转换不会改变使联合概率达到最大值的变量取值组合。至此，从因子节点向变量节点传递消息的计算公式变为：

$$m_{f_j \to x_i}(x_i) = \max_{\boldsymbol{x} \setminus x_i} \left(\ln f_j(\boldsymbol{x}) + \sum_{x_k \in \mathcal{S}(f_j) \setminus x_i} m_{x_k \to f_j}(x_k) \right) \tag{9.50}$$

其中 $\mathcal{S}(f_j)$ 表示与因子节点 f_j 相邻的变量节点集合，$\mathcal{S}(f_j) \setminus x_i$ 表示从该集合中去除接收变量节点 x_i 后得到的集合，而 $\boldsymbol{x}$ 则表示集合 $\mathcal{S}(f_j)$ 中随机变量组成的向量。公式中的最大化操作只对与 f_j 相邻且不同于 x_i 的变量进行，即对于 x_i 的每一种可能取值，将因子函数的对数与其他相邻变量节点传入的对数消息相加，再从这些相邻变量的所有取值组合中选取最大值。与和积算法相比，上述计算将求和操作替换为取最大值，并通过取对数将乘法转换为加法，因此称为最大和算法（The Max-sum Algorithm）。如果仅将求和操作替换为取最大值，但仍采用乘法运算（不取对数），则称为最大积算法（The Max-product Algorithm）。

相应地，最大和算法中从变量节点向因子节点传递消息的计算也需要由连乘改为求和：

$$m_{x_i \to f_j}(x_i) = \sum_{f_s \in \mathcal{S}(x_i) \setminus f_j} m_{f_s \to x_i}(x_i) \tag{9.51}$$

其中 $\mathcal{S}(x_i)$ 表示与变量节点 x_i 相邻的因子节点集合，$\mathcal{S}(x_i) \setminus f_j$ 表示从该集合中去除接收因子节点 f_j 后得到的集合。也就是说，将除接收因子节点 f_j 之外的所有相邻因子节点传入的对数消息相加。如果某个因子值为 0，则约定其对数为 $-\infty$，表示该因子对应的变量取值组合不可

能出现。如图9.15所示的两类叶节点，相应的初始消息也需要调整为：

$$m_{x_i \to f_j}(x_i) = \ln(1) = 0 \tag{9.52}$$

$$m_{f_j \to x_i}(x_i) = \ln f_j(x_i) \tag{9.53}$$

当根节点 x_i 收到所有相邻因子节点传入的消息后，类似和积算法中的公式9.39相应修改为：

$$\ln p^* = \max_{x_i} \left(\sum_{f_s \in \mathcal{S}(x_i)} m_{f_s \to x_i}(x_i) \right) \tag{9.54}$$

其中 $\mathcal{S}(x_i)$ 表示与根节点变量 x_i 相邻的因子节点集合，而 p^* 则表示最大联合概率。由于消息是在对数域中计算的，上式得到的是最大联合概率的对数。相应地，使联合概率达到最大时，其根节点 x_i 的取值可由下式确定：

$$x_i^* = \arg\max_{x_i} \left(\sum_{f_s \in \mathcal{S}(x_i)} m_{f_s \to x_i}(x_i) \right) \tag{9.55}$$

为了便于理解最大和算法，下面以由三个二值离散随机变量构成的链式因子图为例，具体说明上述计算过程。设这三个随机变量分别为 x_1、x_2 和 x_3，且均只能取 0 或 1 两个值。其联合概率分布可分解为：

$$p(x_1, x_2, x_3) = f_a(x_1) f_b(x_1, x_2) f_c(x_2, x_3)$$

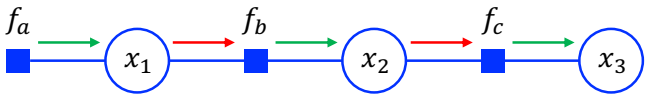


图 9.16: 由三个随机变量构成的链式因子图及其消息传递方向。其中因子节点 f_a 仅与变量节点 x_1 相连，f_b 与 x_1 和 x_2 相连，f_c 与 x_2 和 x_3 相连。红色箭头表示变量节点向因子节点传递消息的方向，绿色箭头表示因子节点向变量节点传递消息的方向。

如图9.16所示，因子节点 f_a 仅与变量节点 x_1 相连，f_b 与 x_1 和 x_2 相连，而 f_c 则与 x_2 和 x_3 相连，从而构成一个链式因子图。链式因子图是树形因子图的一种特殊情形。为了便于理解，我们将三个因子函数分别定义为：

$$f_a(x_1) = p(x_1), \quad f_b(x_1, x_2) = p(x_2|x_1), \quad f_c(x_2, x_3) = p(x_3|x_2)$$

上述因子分解对应于有向链 $x_1 \to x_2 \to x_3$。由于各局部分布均满足相应的归一化条件，三个因子的乘积也自然满足联合概率分布的归一化条件。各因子函数的具体取值如表9.3所示。

表 9.3: 链式因子图示例中的三个因子函数及其取值。

$f_a(x_1)$	$p(x_1)$
$x_1 = 0$	0.6
$x_1 = 1$	0.4

$f_b(x_1, x_2)$	$x_2 = 0$	$x_2 = 1$
$x_1 = 0$	0.2	0.8
$x_1 = 1$	0.7	0.3

$f_c(x_2, x_3)$	$x_3 = 0$	$x_3 = 1$
$x_2 = 0$	0.2	0.8
$x_2 = 1$	0.6	0.4

我们以 x_3 为根节点，从叶节点 f_a 开始，按照最大和算法的规则向根节点传递消息。由于

变量节点 x_1 除因子节点 f_b 之外，仅与 f_a 相邻，因此，x_1 向 f_b 发送的消息就是其从 f_a 接收到的消息，即：

$$m_{x_1\to f_b}(0) = m_{f_a\to x_1}(0) = \ln(0.6)$$
$$m_{x_1\to f_b}(1) = m_{f_a\to x_1}(1) = \ln(0.4)$$

接下来，我们计算因子节点 f_b 向变量节点 x_2 发送的消息，并为 x_2 的每一种可能取值记录相应的最优 x_1 取值。根据公式9.50，我们有：

$$m_{f_b\to x_2}(x_2) = \max_{x_1}\left(\ln f_b(x_1,x_2) + m_{x_1\to f_b}(x_1)\right)$$

将因子函数及传入消息的具体数值代入（下划线表示比较后得到的最大值），可得：

$$m_{f_b\to x_2}(0) = \max_{x_1}\begin{bmatrix}\ln f_b(0,0)+m_{x_1\to f_b}(0)\\ \ln f_b(1,0)+m_{x_1\to f_b}(1)\end{bmatrix} = \max_{x_1}\begin{bmatrix}\ln(0.2)+\ln(0.6)\\ \underline{\ln(0.7)+\ln(0.4)}\end{bmatrix} = \ln(0.28)$$
$$\phi_{f_b\to x_2}(0) = \arg\max_{x_1}\left(\ln f_b(x_1,0) + m_{x_1\to f_b}(x_1)\right) = 1$$
$$m_{f_b\to x_2}(1) = \max_{x_1}\begin{bmatrix}\ln f_b(0,1)+m_{x_1\to f_b}(0)\\ \ln f_b(1,1)+m_{x_1\to f_b}(1)\end{bmatrix} = \max_{x_1}\begin{bmatrix}\underline{\ln(0.8)+\ln(0.6)}\\ \ln(0.3)+\ln(0.4)\end{bmatrix} = \ln(0.48)$$
$$\phi_{f_b\to x_2}(1) = \arg\max_{x_1}\left(\ln f_b(x_1,1) + m_{x_1\to f_b}(x_1)\right) = 0$$

上述计算结果表明：当 $x_2=0$ 时，相应的最优取值为 $x_1=1$；当 $x_2=1$ 时，相应的最优取值为 $x_1=0$。在该链式因子图中，变量节点 x_2 仅与因子节点 f_b 和 f_c 相连。因此，根据公式9.51，计算 x_2 向 f_c 发送的消息时，在排除接收节点 f_c 后，只剩下从 f_b 传入的消息。此时，只需将该传入消息直接转发，无需额外计算，即：

$$m_{x_2\to f_c}(x_2) = m_{f_b\to x_2}(x_2)$$

接下来，我们计算因子节点 f_c 向根节点 x_3 发送的消息。再次利用公式9.50，可得：

$$m_{f_c\to x_3}(0) = \max_{x_2}\begin{bmatrix}\ln f_c(0,0)+m_{x_2\to f_c}(0)\\ \ln f_c(1,0)+m_{x_2\to f_c}(1)\end{bmatrix} = \max_{x_2}\begin{bmatrix}\ln(0.2)+\ln(0.28)\\ \underline{\ln(0.6)+\ln(0.48)}\end{bmatrix} = \ln(0.288)$$
$$\phi_{f_c\to x_3}(0) = \arg\max_{x_2}\left(\ln f_c(x_2,0) + m_{x_2\to f_c}(x_2)\right) = 1$$
$$m_{f_c\to x_3}(1) = \max_{x_2}\begin{bmatrix}\ln f_c(0,1)+m_{x_2\to f_c}(0)\\ \ln f_c(1,1)+m_{x_2\to f_c}(1)\end{bmatrix} = \max_{x_2}\begin{bmatrix}\underline{\ln(0.8)+\ln(0.28)}\\ \ln(0.4)+\ln(0.48)\end{bmatrix} = \ln(0.224)$$
$$\phi_{f_c\to x_3}(1) = \arg\max_{x_2}\left(\ln f_c(x_2,1) + m_{x_2\to f_c}(x_2)\right) = 0$$

根据上述计算结果，比较 x_3 取 0 和 1 时对应的消息值。由于 $\ln(0.288) > \ln(0.224)$，可以确定根节点的最优取值为 $x_3^*=0$，对应的最大联合概率为 0.288。确定根节点的取值后，我们沿消息传递的相反方向进行回溯（如图9.17所示）。根据之前保存的最优取值记录，依次确定：

$$x_2^* = \phi_{f_c\to x_3}(0) = 1, \quad x_1^* = \phi_{f_b\to x_2}(1) = 0$$

最终得到使联合概率达到最大值 0.288 的变量取值组合为 $(x_1^*, x_2^*, x_3^*) = (0,1,0)$。值得注意的是，

我们将在第13章介绍的维特比算法（Viterbi Algorithm）可以看作最大和算法在隐马尔可夫模型链式结构上的应用，该算法用于在给定观测序列的条件下寻找最可能的隐状态序列。

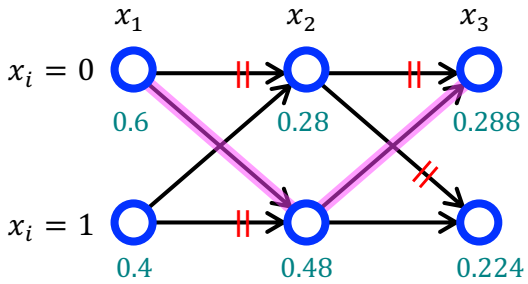


图 9.17: 最大和算法在链式结构中的计算与回溯过程。图中每列对应一个变量，而上下两行分别表示取值 0 和 1。绿色数值表示固定当前变量取值时，从 x_1 到该变量的最优取值组合所对应的联合概率。红色标记表示在取最大值时舍弃的候选连接。当确定 $x_3^* = 0$ 后，依据保存的最优取值记录向左回溯，依次得到 $x_2^* = 1$ 和 $x_1^* = 0$。粉色高亮路径表示最终的最优取值组合 $(0, 1, 0)$，相应的联合概率为 0.288。

9.6 模型参数学习

模型的参数估计可以从两个层面来理解：一是依据什么统计原则确定估计目标，相应的原则包括最大似然估计、最大后验估计和贝叶斯估计等；二是采用什么计算方法求解相应的估计问题，相关的参数优化算法包括梯度下降法、牛顿法等。参数学习与估计是贯穿机器学习的重要内容，之前的多个章节已介绍了相关的统计原则和计算方法，本节将不再赘述。本节旨在从一个简单的例子出发，重点讨论推断与学习之间的关系，进而说明在什么情况下可以采用精确推断，以及在什么情况下则需要采用近似推断。

假设我们有两枚硬币 A 和 B，希望通过重复抛掷实验估计它们各自正面朝上的概率。待估计的参数记为 μ_A 和 μ_B，分别表示硬币 A 和 B 正面朝上的概率，相应的反面朝上概率则分别为 $1 - \mu_A$ 和 $1 - \mu_B$。在每次实验中，我们会随机选择一枚硬币，连续抛掷 10 次，并记录正面朝上的次数。我们先假设，对同一枚硬币的各次抛掷相互独立，且其正面朝上的概率保持不变。若知道每次实验选中的是哪枚硬币，就可以通过简单的频数统计，分别得到两个参数的最大似然估计：对于每枚硬币，只需将其在所有实验中正面朝上的总次数，除以该硬币的总抛掷次数。

但如果并不知道每次实验选中的是哪枚硬币，就需要在模型中引入隐变量 z_i 来表示第 i 次实验选中的是硬币 A 还是硬币 B。同时，我们用 x_i 表示第 i 次实验的观测结果，即连续 10 次抛掷中正面朝上的次数。此外，还需要引入参数 π，表示每次实验选中硬币 A 的概率。相应地，选中硬币 B 的概率为 $1 - \pi$。在这种情况下，我们仍然可以采用最大似然估计，但无法像此前那样，直接通过分别统计两枚硬币正面出现的次数和总抛掷次数来估计 μ_A 和 μ_B，因为我们不知道每次实验的结果应该计入哪枚硬币。

假设我们一共进行了 N 次相互独立的实验，当前需要估计的参数集合记为 $\boldsymbol{\theta} = \{\mu_A, \mu_B, \pi\}$。

如果能够根据当前参数和观测结果 x_i 计算出第 i 次实验选中硬币 A 的后验概率，即：

$$\alpha_i = p(z_i = A|x_i;\boldsymbol{\theta}) \tag{9.56}$$

那么，就可以将第 i 次实验的结果分别以权重 α_i 和 $1-\alpha_i$ 计入硬币 A 和硬币 B 的统计结果，并通过加权频数更新模型参数：

$$\mu_A^{\text{new}} = \frac{\sum_i \alpha_i x_i}{10\sum_i \alpha_i} \tag{9.57}$$

$$\mu_B^{\text{new}} = \frac{\sum_i (1-\alpha_i) x_i}{10\sum_i (1-\alpha_i)} \tag{9.58}$$

$$\pi^{\text{new}} = \frac{1}{N}\sum_i \alpha_i \tag{9.59}$$

其中前两个公式分别用每枚硬币加权后的正面总次数除以加权后的总抛掷次数，第三个公式则将选中硬币 A 的概率更新为各次实验对应后验概率的平均值。

需要注意的是，α_i 依赖于当前的参数估计，因此上述公式只给出一次参数更新。实际计算时，可以从一组初始参数出发，例如 $\mu_A = 0.7$、$\mu_B = 0.3$ 和 $\pi = 0.5$，交替进行隐变量 z_i 的后验推断与模型参数 $\boldsymbol{\theta}$ 的更新，直至满足预设的停止条件。本例说明，在含有隐变量的模型中，参数学习需要借助推断来完成：先利用当前参数计算隐变量的后验分布，再利用这一后验分布更新参数。更新后的参数又成为下一轮推断的依据，从而形成推断与学习交替进行的迭代过程。

在本例中，隐变量 z_i 取值为 A 的后验概率可由如下的贝叶斯公式计算：

$$\alpha_i = \frac{\pi p(x_i|z_i = A;\boldsymbol{\theta})}{\pi p(x_i|z_i = A;\boldsymbol{\theta}) + (1-\pi)p(x_i|z_i = B;\boldsymbol{\theta})} \tag{9.60}$$

其中给定所选硬币后，10 次独立抛掷中正面朝上的次数服从二项分布，因此我们有：

$$p(x_i|z_i = A;\boldsymbol{\theta}) = \begin{pmatrix} 10 \\ x_i \end{pmatrix} \mu_A^{x_i}(1-\mu_A)^{10-x_i}$$

$$p(x_i|z_i = B;\boldsymbol{\theta}) = \begin{pmatrix} 10 \\ x_i \end{pmatrix} \mu_B^{x_i}(1-\mu_B)^{10-x_i}$$

因此，本例中隐变量的后验概率可以在合理的计算代价下精确求得，而无需采用近似推断方法。

我们进一步修改实验方式：不再要求每组实验连续使用同一枚硬币，而是在第一次抛掷时随机选择一枚硬币。此后，每次抛掷前都有一定的概率沿用前一次使用的硬币，也有一定的概率改用另一枚硬币。我们还假设当前选择哪枚硬币只取决于上一次所使用的硬币。设共进行 M 次抛掷，我们用隐变量 z_i 表示第 i 次抛掷所使用的硬币，用 x_i 表示该次抛掷的观测结果。此时，隐变量之间的依赖关系形成链式结构 $z_1 \to z_2 \to \cdots \to z_M$，并且每个 z_i 都指向相应的观测变量 x_i，即 $z_i \to x_i$。模型中共有 M 个隐变量，如果直接枚举所有隐变量的取值组合，就需要考虑 2^M 种可能。然而，我们可以利用模型的链式结构，将全局计算分解为一系列局部计算。如果需要计算给定观测序列后各隐变量的后验边缘分布，可以采用和积算法。如果需要寻找后验概率最大的整条隐状态序列，则可以采用最大和或最大积算法，并通过回溯确定各隐变量的取值。因此，虽然上述修改引入了隐变量之间的依赖关系，但由于每个隐变量只有两种可能取值，

且隐变量之间呈链式结构，上述精确推断的计算量随序列长度 M 呈线性增长。因此，本例能够以较小的计算代价完成精确推断，仍然无需引入近似推断方法。

我们进一步修改实验方式，将抛掷实验改回最初的分批进行，每批包含若干次抛掷，每次抛掷仍可选择不同的硬币。但我们假设同一批内相邻的两次抛掷倾向于使用同一枚硬币，同时还假设相邻批次中处于相同位置的两次抛掷也倾向于使用同一枚硬币。如果将每批实验排成一行，隐变量之间就形成了二维网格结构，即每个隐变量 z_i 既与批内相邻位置的隐变量相连，也与相邻批次对应位置的隐变量相连。用无向边表示上述依赖关系，并将每个 z_i 与其对应的观测变量 x_i 相连，就得到与图9.11所示图像去噪模型类似的结构。假设共进行了 M 次抛掷，隐变量的所有可能取值组合仍然有 2^M 种，然而此时图中包含大量的环。有环本身并不意味着无法进行精确推断，但在这种二维网格结构上精确计算后验边缘分布时，往往需要同时处理多个隐变量的联合取值。当批次数和每批抛掷次数都较大时，所需的计算时间和存储空间可能超出可接受范围。此时，通常需要采用近似推断方法来估计隐变量的后验概率。因此，选择精确推断还是近似推断，不能仅依据随机变量的数量及其依赖结构来决定。虽然这些因素会显著影响推断的计算代价，但在实际应用中，还需要考虑具体任务对推断精度的要求，综合权衡计算代价与提高推断精度所带来的实际收益。

主要符号表

数学符号

$\boldsymbol{x}$	向量
y	标量
D	向量维度
x_j	向量的第 j 个分量（向量分量为标量）
f	输入到输出的某种映射
$\mathbf{X}^\top$	矩阵转置
$\mathbf{X}^{-1}$	逆矩阵
$\|\mathbf{X}\|$	矩阵的行列式
$\mathrm{tr}(\mathbf{X})$	矩阵的迹（矩阵所有特征值之和，也等于主对角线元素的总和）
$\ln$	以自然常数 e 为底的对数
e^x 或 $\exp(x)$	自然常数 e 的指数函数
$f'(x)$	一阶导数
$f''(x)$	二阶导数
$\nabla f(\boldsymbol{x})$	一阶梯度（梯度向量）
$\nabla^2 f(\boldsymbol{x})$	二阶梯度（海森矩阵）
$\frac{\partial f(x)}{\partial x}$	一阶偏导
$\frac{\partial^2 f(x)}{\partial x^2}$	二阶偏导

机器学习

$\mathcal{D}$	数据集
N	样本数量
$\boldsymbol{x}_i$	第 i 个训练样本
y_i	第 i 个训练样本对应的输出
$\kappa(\boldsymbol{x}, \boldsymbol{x}')$	核函数
$\mathcal{L}(\cdot)$	损失函数
$\mathcal{J}(\cdot)$	目标函数或代价函数

概率统计

$\mathbb{E}[x]$	均值或期望
$\text{Var}[x]$	方差
$\text{Cov}[x, z]$	随机变量 x 和 z 之间的协方差
$\Gamma(z)$	伽玛函数
$\mathcal{N}(x\|\mu, \sigma^2)$	高斯分布，其中 μ 为均值，σ 为标准差
$\mathcal{N}(x\|\mu, \lambda^{-1})$	高斯分布，其中 μ 为均值，λ 为精度
$\text{Bern}(x\|\mu)$	伯努利分布
$\text{Bin}(k\|\mu, N)$	二项分布
$\text{Beta}(\mu\|\alpha, \beta)$	贝塔分布
$\text{Poisson}(k\|\lambda)$	泊松分布
$\text{Exp}(\lambda\|\beta)$	指数分布
$\text{Gamma}(\lambda\|\alpha, \beta)$	伽玛分布，其中 α 为形状参数，β 为逆尺度参数
$\text{Mult}(\boldsymbol{m}\|\boldsymbol{\mu}, N)$	多项分布
$\text{Dir}(\boldsymbol{\mu}\|\boldsymbol{\alpha})$	狄利克雷分布
$\mathcal{N}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Sigma})$	多维高斯分布，其中 $\boldsymbol{\mu}$ 为均值向量，$\boldsymbol{\Sigma}$ 为协方差矩阵
$\mathcal{N}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Lambda})$	多维高斯分布，其中 $\boldsymbol{\mu}$ 为均值向量，$\boldsymbol{\Lambda}$ 为精度矩阵
$\mathcal{W}(\boldsymbol{\Lambda}\|\mathbf{W}, \eta)$	威沙特分布，其中 $\mathbf{W}$ 为尺度矩阵，η 为自由度
$\text{NormalGamma}(\mu, \lambda\|\upsilon, \nu, \alpha, \beta)$	高斯-伽玛分布
$\text{NormalWishart}(\boldsymbol{\mu}, \boldsymbol{\Lambda}\|\boldsymbol{\upsilon}, \nu, \mathbf{W}, \eta)$	高斯-威沙特分布
$\text{St}(x\|\mu, \tau, \eta)$	学生氏分布，其中 μ 为均值，τ 为精度，η 为自由度
$\text{St}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Lambda}, \eta)$	多维学生氏分布，其中 $\boldsymbol{\mu}$ 为均值向量，$\boldsymbol{\Lambda}$ 为精度矩阵
$\text{Laplace}(x\|\mu, \tau)$	拉普拉斯分布，其中 μ 为位置参数，τ 为尺度参数
$\chi^2(x\|\eta)$	卡方分布，其中 η 为自由度
$\text{U}(x\|a, b)$	连续型均匀分布，其中 a 为下界、b 为上界 $(a < b)$
$\text{H}[x]$	信息熵，其中 x 为随机变量
$\text{H}[y\|x]$	条件熵，其中 x 和 y 为随机变量
$\text{H}[p, q]$	分布 p 和 q 之间的交叉熵
$D_{\text{KL}}(p\|\|q)$	分布 p 和 q 之间的 KL 散度

参考文献

[1] Daniel J. Amit, Hanoch Gutfreund, and Haim Sompolinsky. “Storing infinite numbers of patterns in a spin-glass model of neural networks”. In: *Physical Review Letters* 55.14 (1985), pp. 1530–1533.

[2] Mikel Artetxe et al. “On the role of bidirectionality in language model pre-training”. In: *Findings of the Conference on Empirical Methods in Natural Language Processing* (2022), pp. 3973–3985.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. “Layer normalization”. In: *arXiv: 1607.06450* (2016).

[4] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166.

[5] Jame O. Berger. *Statistical decision theory and Bayesian analysis (second edition)*. Springer, 1980.

[6] James Bergstra and Yoshua Bengio. “Random search for hyper-parameter optimization”. In: *Journal of Machine Learning Research* 13.2 (2012).

[7] Jeremy Bernstein and Laker Newhouse. “Old optimizer, new norm: An anthology”. In: *Proceedings of the 16th Annual Workshop on Optimization for Machine Learning* (2024), pp. 1–19.

[8] J Besag. “On spatio-temporal models and Markov fields”. In: *Transactions of the 7th Prague Conference on Information Theory, Statistical Decision Functions and Random Processes* (1974), pp. 47–75.

[9] Julian Besag. “On the statistical analysis of dirty pictures”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 48.3 (1986), pp. 259–279.

[10] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[11] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. “A training algorithm for optimal margin classifiers”. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. 1992, pp. 144–152.

[12] George E.P. Box and Norman Richard Draper. *Empirical model-building and response surfaces*. John Wiley & Sons, 1987.

[13] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[14] Tom Brown et al. “Language models are few-shot learners”. In: *Proceedings of the Conference on Neural Information Processing Systems* 33 (2020), pp. 1877–1901.

[15] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder–decoder for statistical machine translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (2014), pp. 1724–1734.

[16] Peter Clifford. “Markov random fields in statistics”. In: *Disorder in Physical Systems: A Volume in Honour of John M. Hammersley* (1990), pp. 19–32.

[17] Thomas H. Cormen et al. *Introduction to algorithms (third edition)*. The MIT Press, 2022.

[18] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley, 1991.

[19] George Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[20] Morris H. DeGroot and Mark J. Schevish. *Probability and statistics (forth edition)*. China Machine Press, 2012.

[21] Jacob Devlin et al. “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2019), pp. 4171–4186.

[22] Alexey Dosovitskiy et al. “An image is worth 16×16 words: Transformers for image recognition at scale”. In: *Proceedings of the International Conference on Learning Representations* (2021).

[23] John Duchi, Elad Hazan, and Yoram Singer. “Adaptive subgradient methods for online learning and stochastic optimization”. In: *Journal of Machine Learning Research* 12.7 (2011).

[24] Walter D. Fisher. “On grouping for maximum homogeneity”. In: *Journal of the American Statistical Association* 53.284 (1958), pp. 789–798.

[25] Roger Fletcher. *Practical methods of optimization (second edition)*. Wiley, 1987.

[26] Yoav Freund and Robert E. Schapire. “A decision-theoretic generalization of on-line learning and an application to boosting”. In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139.

[27] Stuart Geman and Donald Geman. “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), pp. 721–741.

[28] Felix A. Gers, Nicol N. Schraudolph, and Jürgen Schmidhuber. “Learning precise timing with LSTM recurrent networks”. In: *Journal of Machine Learning Research* 3 (2002), pp. 115–143.

[29] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 249–256.

[30] Ian J Goodfellow et al. “Generative adversarial nets”. In: *Proceedings of the Conference on Neural Information Processing Systems* 27 (2014).

[31] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction (second edition)*. Springer, 2009.

[32] Kaiming He and Jian Sun. “Convolutional neural networks at constrained time cost”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 5353–5360.

[33] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.

[34] Kaiming He et al. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 1026–1034.

[35] Nicholas J. Higham. *Functions of matrices: Theory and computation*. SIAM, 2008.

[36] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network”. In: *arXiv: 1503.02531* (2015).

[37] Geoffrey E. Hinton. “Training products of experts by minimizing contrastive divergence”. In: *Neural Computation* 14.8 (2002), pp. 1771–1800.

[38] John J. Hopfield. “Neural networks and physical systems with emergent collective computational abilities”. In: *Proceedings of the National Academy of Sciences* 79.8 (1982), pp. 2554–2558.

[39] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *Proceedings of the International Conference on Machine Learning* (2015), pp. 448–456.

[40] Arieh Iserles. *A first course in the numerical analysis of differential equations*. 44. Cambridge university press, 2009.

[41] Max Jaderberg et al. “Population based training of neural networks”. In: *arXiv: 1711.09846* (2017).

[42] Kevin Jamieson and Ameet Talwalkar. “Non-stochastic best arm identification and hyperparameter optimization”. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics* (2016), pp. 240–248.

[43] Keller Jordan et al. *Muon: An optimizer for hidden layers in neural networks*. 2024. URL: https://kellerjordan.github.io/posts/muon/.

[44] Diederik P. Kingma and Jimmy L. Ba. “Adam: A method for stochastic optimization”. In: *Proceedings of the International Conference on Learning Representations* (2015).

[45] Josef Kittler and Janos Föglein. “Contextual classification of multispectral pixel data”. In: *Image and vision computing* 2.1 (1984), pp. 13–29.

[46] Solomon Kullback and Richard A. Leibler. “On information and sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[47] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (2002), pp. 2278–2324.

[48] Mike Lewis et al. “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), pp. 7871–7880.

[49] Yaron Lipman et al. “Flow matching guide and code”. In: *arXiv: 2412.06264* (2024).

[50] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv: 1711.05101* (2017).

[51] Calvin Luo. “Understanding diffusion models: A unified perspective”. In: *arXiv: 2208.11970* (2022).

[52] Marvin Minsky and Seymour Papert. *Perceptrons: An introduction to computational geometry*. The MIT Press, 1969.

[53] Kevin P. Murphy. *Machine learning: A probabilistic perspective*. The MIT Press, 2012.

[54] Brady Neal et al. “A modern take on the bias-variance tradeoff in neural networks”. In: *arXiv: 1810.08591* (2018).

[55] Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Proceedings of the Conference on Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[56] Judea Pearl. *Probabilistic reasoning in intelligent systems*. Morgan kaufmann, 2014.

[57] Matthew E. Peters et al. “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2018), pp. 2227–2237.

[58] John C. Platt. “Fast training of support vector machines using sequential minimal optimization”. In: *Advances in Kernel Methods* (1999), pp. 185–208.

[59] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *Proceedings of the International Conference on Learning Representations* (2016).

[60] Alec Radford et al. “Improving language understanding by generative pre-training”. In: *OpenAI’s Technical Report* (2018).

[61] Colin Raffel et al. “Exploring the limits of transfer learning with a unified text-to-text transformer”. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67.

[62] Scott Reed et al. “Generative adversarial text to image synthesis”. In: *Proceedings of the International Conference on Machine Learning* (2016), pp. 1060–1069.

[63] Edmund T. Rolls. *Cerebral cortex: principles of operation*. Oxford University Press, 2016.

[64] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional networks for biomedical image segmentation”. In: *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention* (2015), pp. 234–241.

[65] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. “Learning representations by back-propagating errors”. In: *Nature* 323.6088 (1986), pp. 533–536.

[66] Leonard J. Savage. *The subjective basis of statistical practice*. University of Michigan, 1961.

[67] Andrew M Saxe, James L McClelland, and Surya Ganguli. “Exact solutions to the nonlinear dynamics of learning in deep linear neural networks”. In: *Proceedings of the International Conference on Learning Representations* (2014).

[68] Robert E. Schapire. “The strength of weak learnability”. In: *Machine learning* 5.2 (1990), pp. 197–227.

[69] Rico Sennrich, Barry Haddow, and Alexandra Birch. “Neural machine translation of rare words with subword units”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (2016), pp. 1715–1725.

[70] Claude Elwood Shannon. “A mathematical theory of communication”. In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[71] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. “Practical Bayesian optimization of machine learning algorithms”. In: *Proceedings of the Conference on Neural Information Processing Systems* 25 (2012).

[72] Yang Song et al. “Score-based generative modeling through stochastic differential equations”. In: *Proceedings of the International Conference on Learning Representations* (2021).

[73] Gilbert Strang. *Introduction to linear algebra*. Wellesley-Cambridge Press, 2016.

[74] Jianlin Su et al. “Roformer: Enhanced transformer with rotary position embedding”. In: *Neurocomputing* 568 (2024), p. 127063.

[75] Tijmen Tieleman and Geoffrey E. Hinton. “Lecture 6.5-RMSprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural Networks for Machine Learning* 4.2 (2012), pp. 26–26.

[76] Antonio Torralba, Phillip Isola, and William Freeman. *Foundations of computer vision*. The MIT Press, 2024.

[77] Ashish Vaswani et al. “Attention is all you need”. In: *Proceedings of the Conference on Neural Information Processing Systems* 30 (2017).

[78] Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. “Deep learning for Chinese word segmentation and POS tagging”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (2013), pp. 647–657.

[79] Ji Zhu et al. “Multi-class AdaBoost”. In: *Statistics and its Interface* 2.3 (2009), pp. 349–360.

[80] Hui Zou and Trevor Hastie. “Regularization and variable selection via the elastic net”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 67.2 (2005), pp. 301–320.