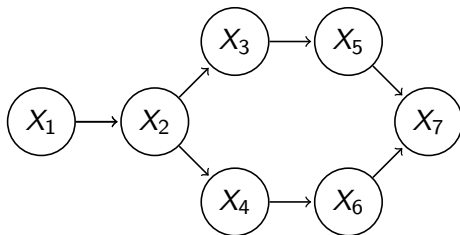


## Specifying a graphical model



This graphical model specifies a joint distribution

$$\begin{aligned} p(X_1, X_2, X_3, X_4, X_5, X_6, X_7) &= \prod_i p(X_i | X_{\text{pa}(i)}) \\ &= p(X_1) p(X_2 | X_1) p(X_3 | X_2) p(X_4 | X_2) \\ &\quad p(X_5 | X_3) p(X_6 | X_4) p(X_7 | X_5, X_6) \end{aligned}$$

# Determining conditional independencies from graphs

In the topological order of nodes of a DAG, parent nodes precede child nodes. **There can be many topological orders.**

Given an order  $O$ , let  $\mathbf{pnp}_O(i)$  denote a set of nodes that precede node  $i$  in a topological order but are not its parents.

We can show that

$$X_i \perp X_{\mathbf{pnp}_O(i)} | X_{\mathbf{pa}(i)}$$

These are basic conditional independence relationships.

# Verifying a conditional independence

We want to show

$$p(X_3|X_2, X_1) = \frac{p(X_3, X_2, X_1)}{p(X_2, X_1)} = p(X_3|X_1)$$

Marginals are

$$\begin{aligned} p(X_3, X_2, X_1) &= p(X_1)p(X_2|X_1)p(X_3|X_1) \\ p(X_2, X_1) &= p(X_1)p(X_2|X_1) \end{aligned}$$

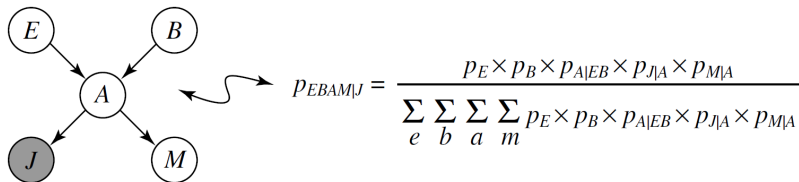
plugging them in we get

$$p(X_3|X_2, X_1) = \frac{p(X_3, X_2, X_1)}{p(X_2, X_1)} = \frac{p(X_1)p(X_2|X_1)p(X_3|X_1)}{p(X_1)p(X_2|X_1)} = p(X_3|X_1)$$

and this confirms  $X_3 \perp X_2|X_1$

# Representing evidence in Bayesian networks

- ▶ When we condition on some of the variables, the result is a conditional density that can have different independence properties.
- ▶ When we condition on a variable, we shade the corresponding node in the Bayes net.



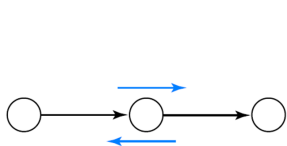
# Encoding conditional independence via d-separation

- ▶ Bayesian networks encode the independence properties of a density.
- ▶ We can determine if a conditional independence  $X \perp Y | Z$  holds by appealing to a graph separation criterion called *d-separation* (*direction-dependent separation*).
- ▶  $X$  and  $Y$  are d-separated if there is no active path between them.
- ▶ The formal definition of active paths is somewhat involved. The *Bayes Ball Algorithm* gives a nice graphical definition.

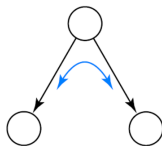
# Bayes Ball Algorithm

- ▶ You want to check if  $X \perp Y | \mathcal{Z}$ . Imagine passing a "ball" from a node to a node, if the ball can make it from X to Y they are dependent. Shade the nodes in  $\mathcal{Z}$  and apply following rules:

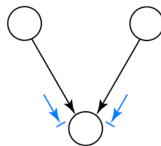
**An Undirected path P is active if a Bayes ball travelling along it never encounters the "stop" symbol  $\rightarrow \perp$**



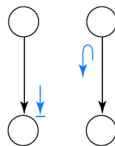
**P Contains a Chain**



**P Contains a fork**

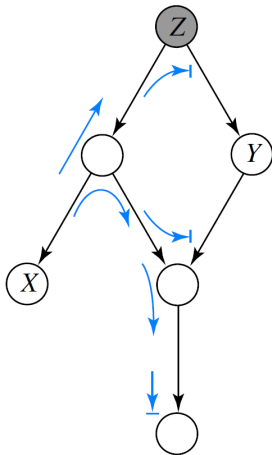


**P Contains a V-structure**



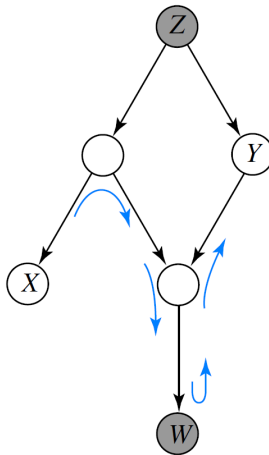
**Boundary Conditions**

## A double-header: two games of Bayes Ball



no active paths

$$X \perp\!\!\!\perp Y \mid Z$$



one active path

$$X \not\perp\!\!\!\perp Y \mid \{W, Z\}$$

# Learning Bayesian Networks

Learning a Bayesian Network requires us to determine

- ▶ network's structure
- ▶ network's parameters

You will implement both of these in your HW2.



# Learning a BayesNet

Probability of a state of all  $n$  variables  $\mathbf{x}$

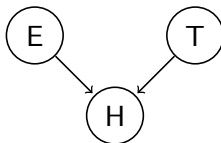
$$p(X_1 = x_1, \dots, X_n = x_n) = \prod_{j=1}^n p(X_j = x_j | X_{\mathbf{pa}(j)} = x_{\mathbf{pa}(j)}, \theta_j)$$

Our goal is to learn parameters  $\Theta = (\theta_1, \dots, \theta_n)$  from multiple samples of the state of the Bayes net.

## Example

Data:

Sample \ Variable	Earthquake	Truck	House moved
1	1	0	1
2	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
S	0	1	0



Conditional distributions and their parameters

- ▶  $p(E = 1 | \theta_E) = \theta_E$ , where  $\theta_E \in [0, 1]$
- ▶  $p(T = 1 | \theta_T) = \theta_T$ , where  $\theta_T \in [0, 1]$
- ▶  $p(H = 1 | E = e, T = t, \theta_H) = \theta_{H,e,t}$ , where  $\theta_{H,e,t} \in [0, 1]$

# Learning a BayesNet – likelihood

Data  $S$  instances of Bayes net's state;  $x_{i,j}$  state of variable  $X_j$  in  $i^{\text{th}}$  sample.

$$L(\Theta) = \underbrace{\prod_{i=1}^S}_{\text{samples}} \underbrace{\prod_{j=1}^n}_{\text{variables}} p(X_j = x_{i,j} | X_{\text{pa}(j)} = x_{i,\text{pa}(j)}, \theta_j)$$

# Likelihood Example

Data:

Sample \ Variable	Earthquake	Truck	House moved
1	1	0	1
2	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$
S	0	1	0

$$\begin{aligned} L(\Theta) = & p(E = 1|\theta_E)p(T = 0|\theta_T)p(H = 1|E = 1, T = 0, \theta_H) \\ & p(E = 0|\theta_E)p(T = 0|\theta_T)p(H = 0|E = 0, T = 0, \theta_H) \\ & \vdots \\ & p(E = 0|\theta_E)p(T = 1|\theta_T)p(H = 0|E = 0, T = 1, \theta_H) \end{aligned}$$

## Likelihood Example – continued

$$\begin{aligned} L(\Theta) &= p(E = 1|\theta_E)p(T = 0|\theta_T)p(H = 1|E = 1, T = 0, \theta_H) \\ &\times p(E = 0|\theta_E)p(T = 0|\theta_T)p(H = 0|E = 0, T = 0, \theta_H) \\ &\times \vdots \\ &\times p(E = 0|\theta_E)p(T = 1|\theta_T)p(H = 0|E = 0, T = 1, \theta_H) \end{aligned}$$

In terms of parameters

$$\begin{aligned} L(\Theta) &= \theta_E \quad (1 - \theta_T) \quad \theta_{H,1,0} \\ &\times (1 - \theta_E) \quad (1 - \theta_T) \quad (1 - \theta_{H,0,0}) \\ &\times \vdots \\ &\times (1 - \theta_E) \quad \theta_T \quad (1 - \theta_{H,1,0}) \end{aligned}$$

# Learning a BayesNet – log-likelihood

Log-likelihood is given by

$$\text{LL}(\Theta) = \underbrace{\sum_{i=1}^S}_{\text{samples}} \underbrace{\sum_{j=1}^n}_{\text{variables}} \log p(X_j = x_{i,j} | X_{\text{pa}(j)} = x_{i,\text{pa}(j)}, \theta_j)$$

Crucial observation:

$$\text{LL}(\Theta) = \underbrace{\sum_{j=1}^n}_{\text{variables}} \underbrace{\sum_{i=1}^S}_{\text{samples}} \log p(X_j = x_{i,j} | X_{\text{pa}(j)} = x_{i,\text{pa}(j)}, \theta_j)$$

# Learning a BayesNet – maximizing log-likelihood

$$\begin{aligned}\operatorname{argmax}_{\theta_j} \text{LL}(\Theta) &= \operatorname{argmax}_{\theta_j} \sum_{j=1}^n \sum_{i=1}^S \log p(X_j = x_{i,j} | X_{\mathbf{pa}(j)} = x_{i,\mathbf{pa}(j)}, \theta_j) \\ &= \operatorname{argmax}_{\theta_j} \sum_{i=1}^S \log p(X_j = x_{i,j} | X_{\mathbf{pa}(j)} = x_{i,\mathbf{pa}(j)}, \theta_j)\end{aligned}$$

To learn parameters  $\theta_j$  we only need states of node  $j$  and its parents.

Further, we do not need to concern ourselves with the rest of the graph.

# Learning BayesNets – structure learning

If we assume that BayesNet is a tree – each variable has a single parent – we can derive an algorithm to discover the optimal structure.

Given an empirical distribution of the data  $f$  we wish to find optimal distribution with factorization

$$p(X_1, \dots, X_n) = \prod_{j=1}^n p(X_j | X_{\mathbf{pa}(j)})$$

where  $\mathbf{pa}(j)$  has at most one element for all  $j$ .

Under this assumption, we can define edge set

$$\text{Edges} = \{(j, \mathbf{pa}(j)) | j = 1, \dots, n\}$$



# Learning BayesNets – structure learning

Writing out log-likelihood

$$\begin{aligned}\text{LL}(\Theta) &= \sum_{i=1}^S \sum_j \log p(x_{i,j} | x_{i,\text{pa}(j)}) \\&= \sum_{i=1}^S \sum_{j=1}^n \sum_{k=1}^n [(j, k) \in \text{Edges}] \log p(x_{i,j} | x_{i,k}) \\&= \sum_{j=1}^n \sum_{k=1}^n [(j, k) \in \text{Edges}] \left( \sum_{i=1}^S \log p(x_{i,j} | x_{i,k}) \right) \\&= \sum_{(j,k) \in \text{Edges}} \left( \sum_{i=1}^S \log p(x_{i,j} | x_{i,k}) \right)\end{aligned}$$

# Learning BayesNets – structure learning

$$LL(\Theta) = \sum_{(j,k) \in \text{Edges}} \left( \sum_{i=1}^S \log p(x_{i,j} | x_{i,k}) \right)$$

We observed that we can learn optimal  $p(X_j | X_k, \theta_j)$  independently of the graph structure.

$$\theta_j^* = \underset{\theta_j}{\operatorname{argmax}} \sum_{i=1}^S \log p(x_{i,j} | x_{i,k}, \theta_j)$$

# Learning BayesNets – structure learning

$$LL(\Theta) = \sum_{(j,k) \in \text{Edges}} \underbrace{\sum_{i=1}^S \log p(x_{i,j} | x_{i,k}, \theta_j^*)}_{\text{weight of edge (j,k)}}$$

Hence, we wish to find a tree with maximum weight, where each edge's weight is

$$w_{j,k} = \sum_{i=1}^S \log p(X_j = x_{i,j} | X_k = x_{i,k}, \theta_j^*)$$

# Chow-Liu tree learning algorithm

Observe that

$$\sum_{i=1}^S \log p(X_j = x_{i,j} | X_k = x_{i,k}) = I(X_j, X_k) - H(X_j)$$

and since  $H(X_j)$  does not depend on the edges we can rewrite the problem as

$$\operatorname{argmax}_{\text{Edges}} \sum_{(j,k) \in \text{Edges}} I(X_j, X_k) - H(X_j) = \operatorname{argmax}_{\text{Edges}} \sum_{(j,k) \in \text{Edges}} I(X_j, X_k)$$

Note that this is an example of **maximum spanning tree** problem which can be solved efficiently – HW2.

# Chow-Liu tree learning algorithm – details

Mutual information  $I(X_j, X_k)$  is computed on the empirical distribution

$$f_{j,k}(a, b) = \frac{\sum_{i=1}^S [X_j = a, X_k = b]}{S}$$

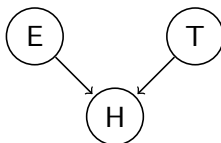
$$f_l(a) = \frac{\sum_{i=1}^S [X_l = a]}{S}$$

$$I(X_j, X_k) = \sum_a \sum_b f_{j,k}(a, b) \log \frac{f_{j,k}(a, b)}{f_j(a)f_k(b)}$$

## Chow-Liu tree – downsides

Single parent assumptions can be restrictive.

The tree models do not permit explaining away.



Tree structured Bayes nets are step above independent models.

On the upside, they are extremely easy to learn.