Graphical Models Review

Anthony Platanios

Graphical Models

Directed

Bayesian Networks

Undirected

Markov Random Fields

Model

- **Conditional independence assumptions**
- Joint probability distribution of variables (parameterized)

Combine Prior Knowledge

- Over **dependencies**
- Over parameter values

Independence

Conditional

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

$$\Leftrightarrow$$

$$\mathbb{P}(X \mid Y, Z) = \mathbb{P}(X \mid Z)$$

Marginal

$$X \perp\!\!\!\perp Y$$

$$\Leftrightarrow$$

$$\mathbb{P}(X \mid Y) = \mathbb{P}(X)$$

How do directed graphical models help?

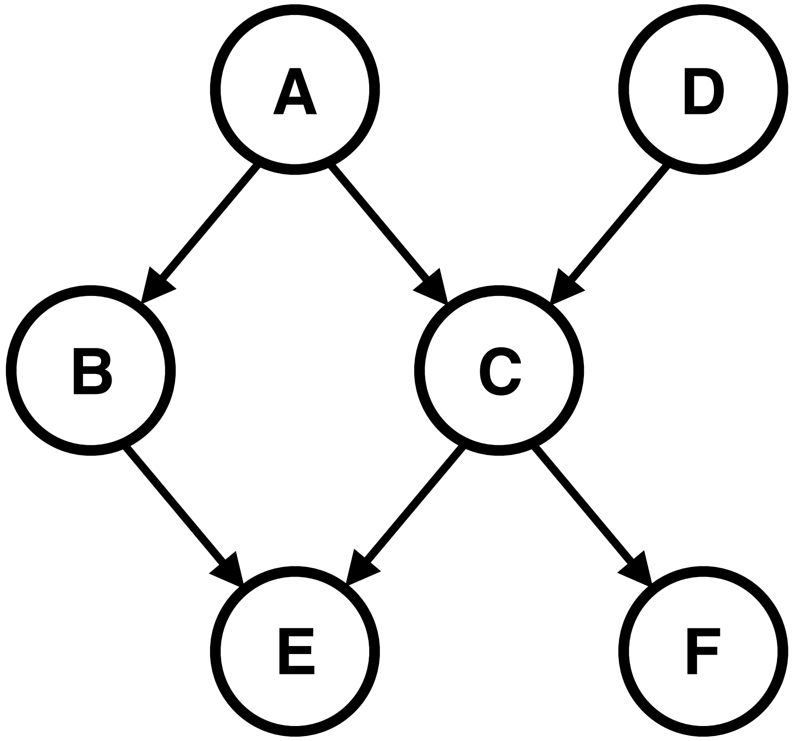
Bayesian Networks - Directed GM

Chain Rule of Probability

$$\begin{aligned}\mathbb{P}(A,B,C,D,E,F) = {} & \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(C \mid A,B) \\ & \mathbb{P}(D \mid A,B,C)\mathbb{P}(E \mid A,B,C,D) \\ & \mathbb{P}(F \mid A,B,C,D,E)\end{aligned}$$

Bayesian Network

$$\begin{aligned}\mathbb{P}(A,B,C,D,E,F) = {} & \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(D) \\ & \mathbb{P}(C \mid A,D) \\ & \mathbb{P}(E \mid B,C) \\ & \mathbb{P}(F \mid C)\end{aligned}$$



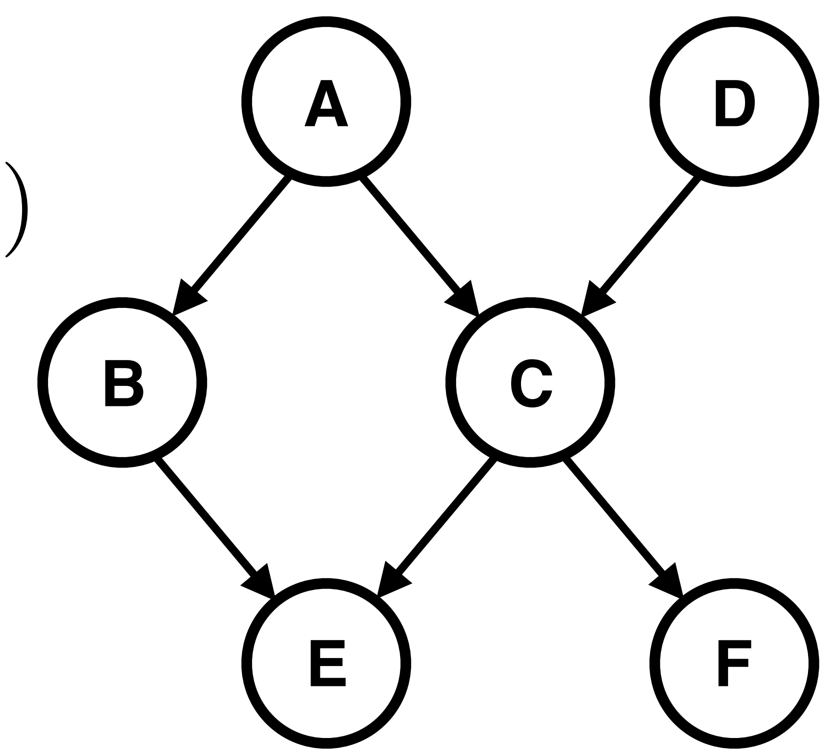
Bayesian Networks - Directed GM

Smaller conditional probability tables (CPTs) → **Less Parameters**

↑

Bayesian Network

$$\mathbb{P}(A,B,C,D,E,F) = \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(D)$$
$$\mathbb{P}(C \mid A, D)$$
$$\mathbb{P}(E \mid B, C)$$
$$\mathbb{P}(F \mid C)$$

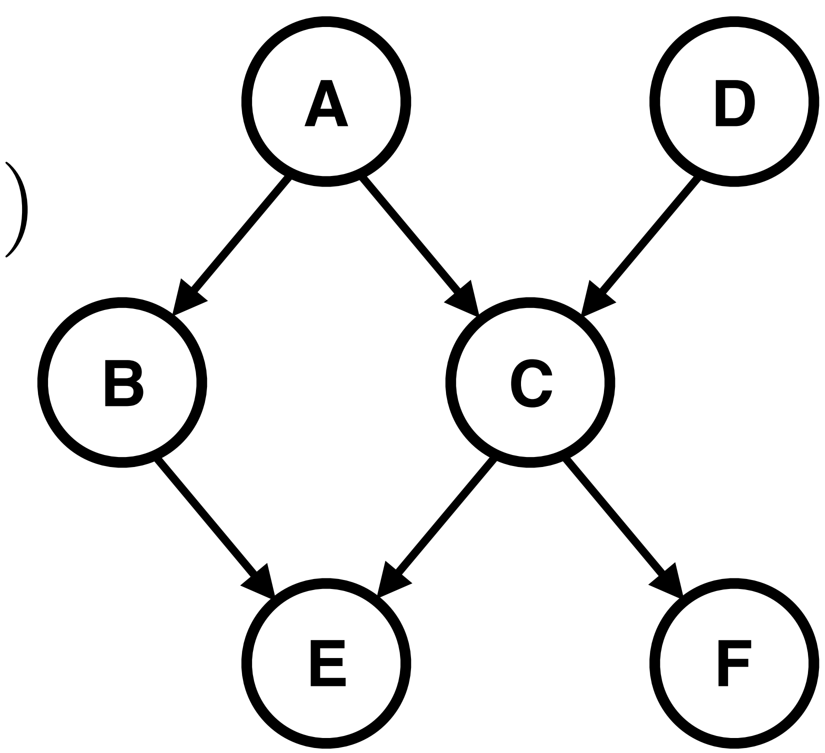


Bayesian Networks - Directed GM

Nodes encode variables

Edges encode **dependencies**

$$\mathbb{P}(A, B, C, D, E, F) = \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(D)\mathbb{P}(C \mid A, D)\mathbb{P}(E \mid B, C)\mathbb{P}(F \mid C)$$



Bayesian Networks - Directed GM

In general:

$$\mathbb{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i \mid \mathrm{Parents}(X_i))$$

But can we determine general conditional independence properties?

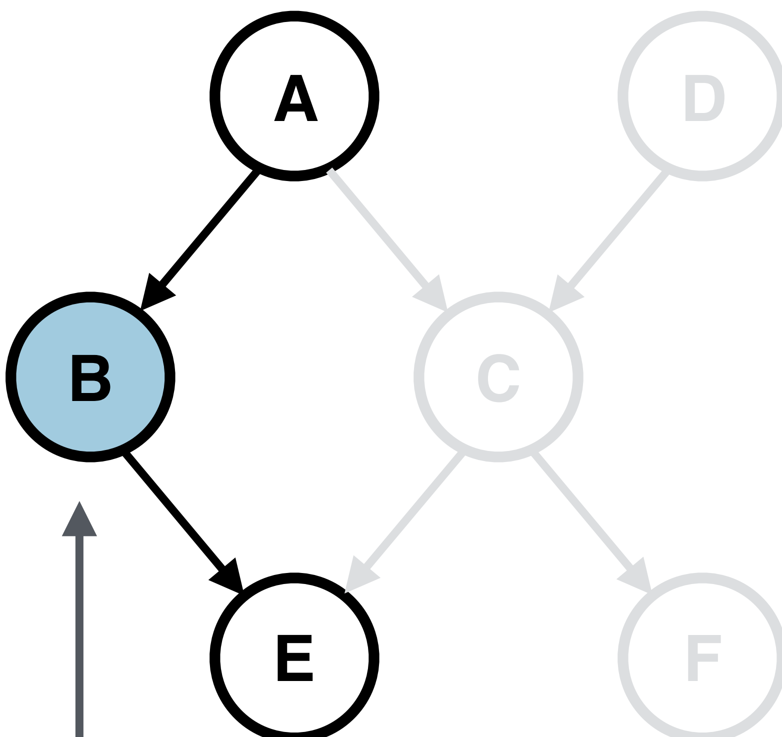
D-Separation

Yes, with the **d-separation criterion!**

In order to see how this is possible, let us first consider **three simple cases**. Then we are going to see a simple way (more like a game) for figuring out independence properties of a graph using this criterion.

D-Separation

Case #1: **Head to Tail**

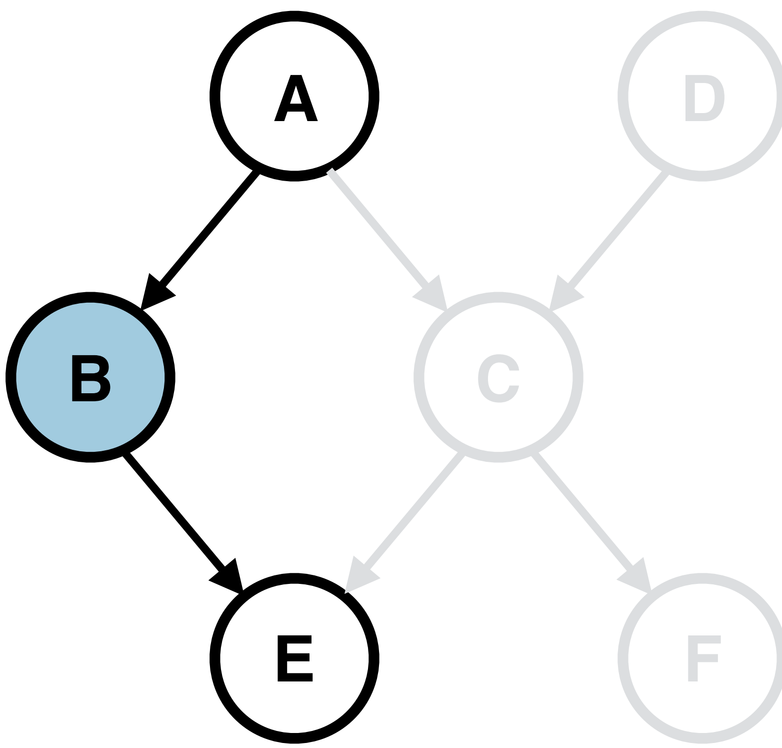


“Heads” and “tails” refer to the connecting edges heads (i.e., arrow pointers) and tails

Shaded nodes are observed and we want to see if observing them induces any independencies

D-Separation

Case #1: **Head to Tail**



$$\mathbb{P}(A, B, E) = \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(E \mid B) \Rightarrow$$

$$\mathbb{P}(A, E \mid B) = \frac{\mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(E \mid B)}{\mathbb{P}(B)}$$

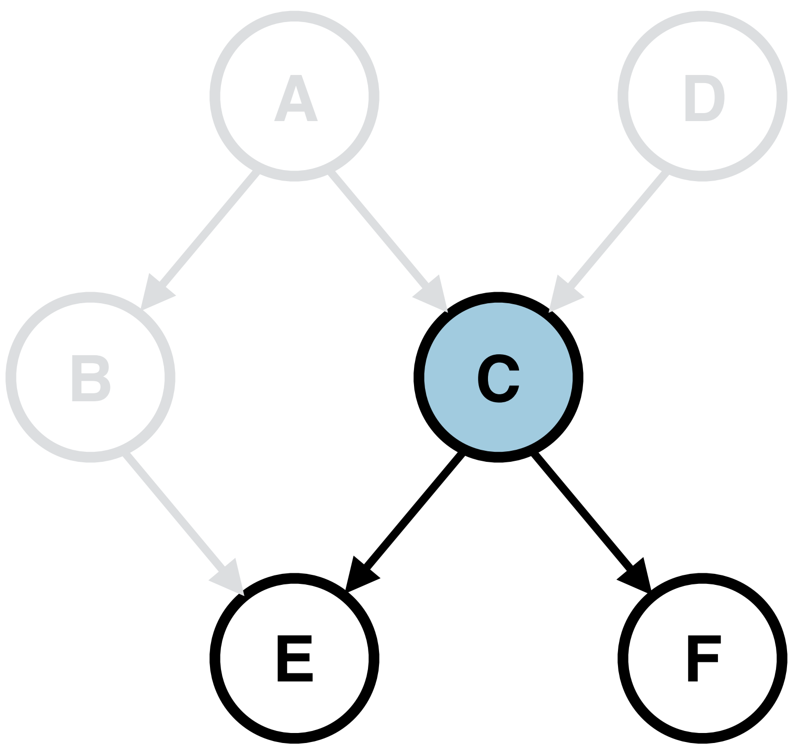
$$= \frac{\mathbb{P}(A, B)}{\mathbb{P}(B)}\mathbb{P}(E \mid A)$$

$$= \mathbb{P}(A \mid B)\mathbb{P}(E \mid B)$$

$$A \perp\!\!\!\perp E \mid B$$

D-Separation

Case #2: **Tail to Tail**



$$\mathbb{P}(C, E, F) = \mathbb{P}(C)\mathbb{P}(E \mid C)\mathbb{P}(F \mid C) \Rightarrow$$

$$\mathbb{P}(E, F \mid C) = \frac{\mathbb{P}(C)\mathbb{P}(E \mid C)\mathbb{P}(F \mid C)}{\mathbb{P}(C)}$$

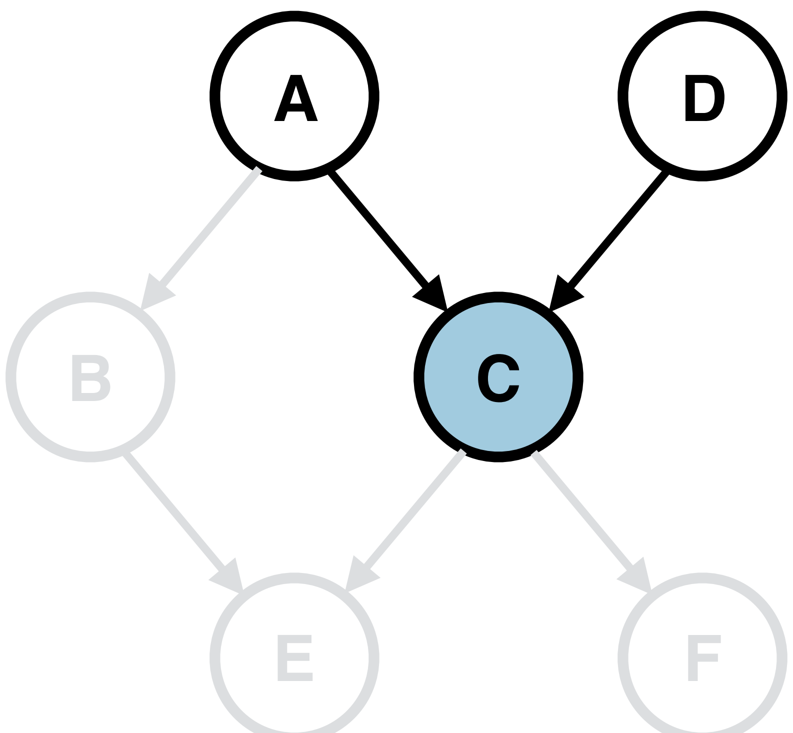
$$= \mathbb{P}(E \mid C)\mathbb{P}(F \mid C)$$

$$\Downarrow$$

$$E \perp\!\!\!\perp F \mid C$$

D-Separation

Case #3: **Head to Head** ⟶ Also known as **colliders**



$$\mathbb{P}(A, C, D) = \mathbb{P}(A)\mathbb{P}(D)\mathbb{P}(C \mid A, D) \Rightarrow$$

$$\mathbb{P}(A, D \mid C) = \frac{\mathbb{P}(A)\mathbb{P}(D)\mathbb{P}(C \mid A, D)}{\mathbb{P}(C)}$$

$$= \mathbb{P}(A, D \mid C)$$

$$A \not\perp\!\!\!\perp D \mid C$$

Explaining Away:

A: Earthquake
B: Break-in
C: Motion alarm

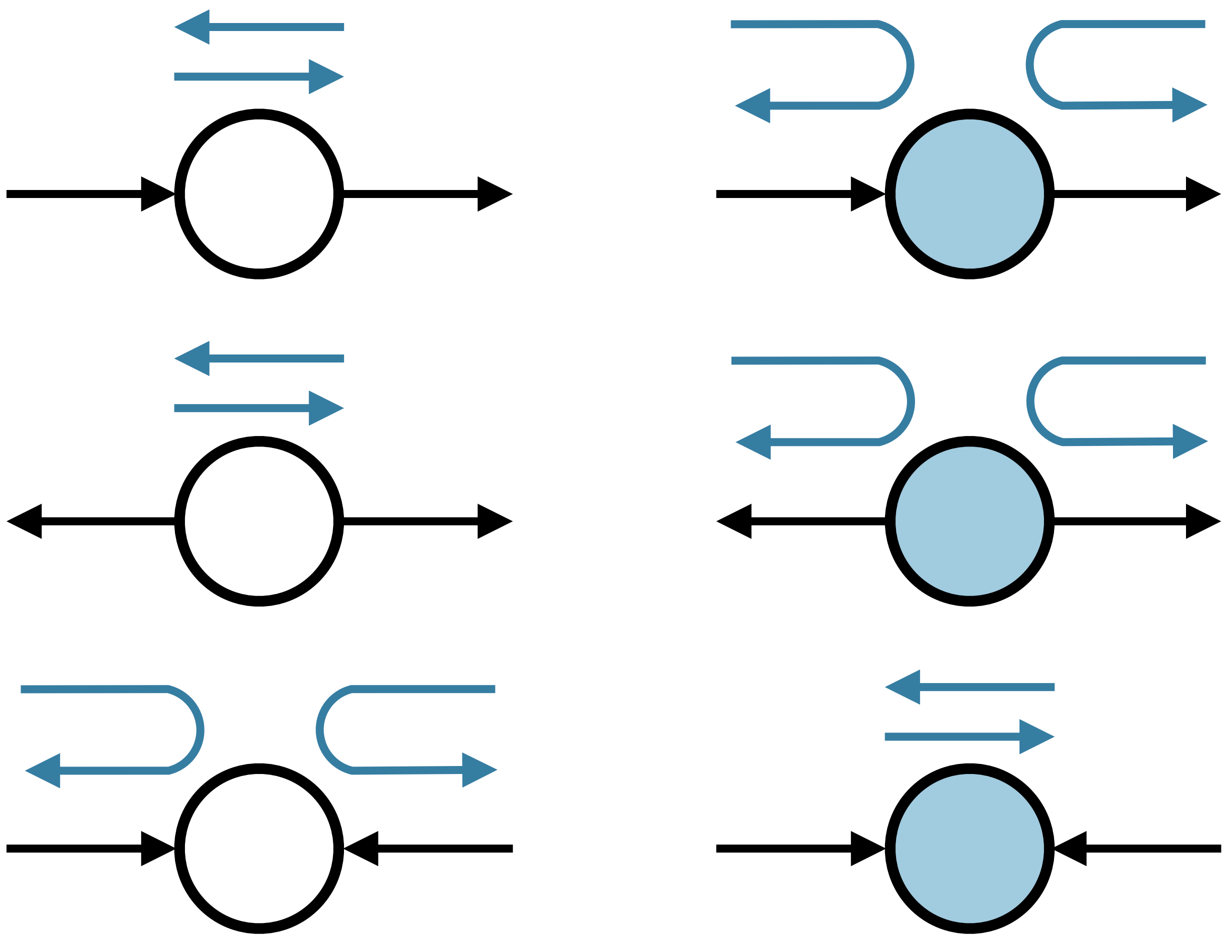
Given that the motion alarm went off, if we learn that a burglar broke in, then we know that it's unlikely an earthquake happened
The burglary event "explains away" the earthquake event

D-Separation through **Bayes Ball**

This might be a little complicated to remember and apply, so let's look at an easier way to work out d-separation

D-Separation through **Bayes Ball**

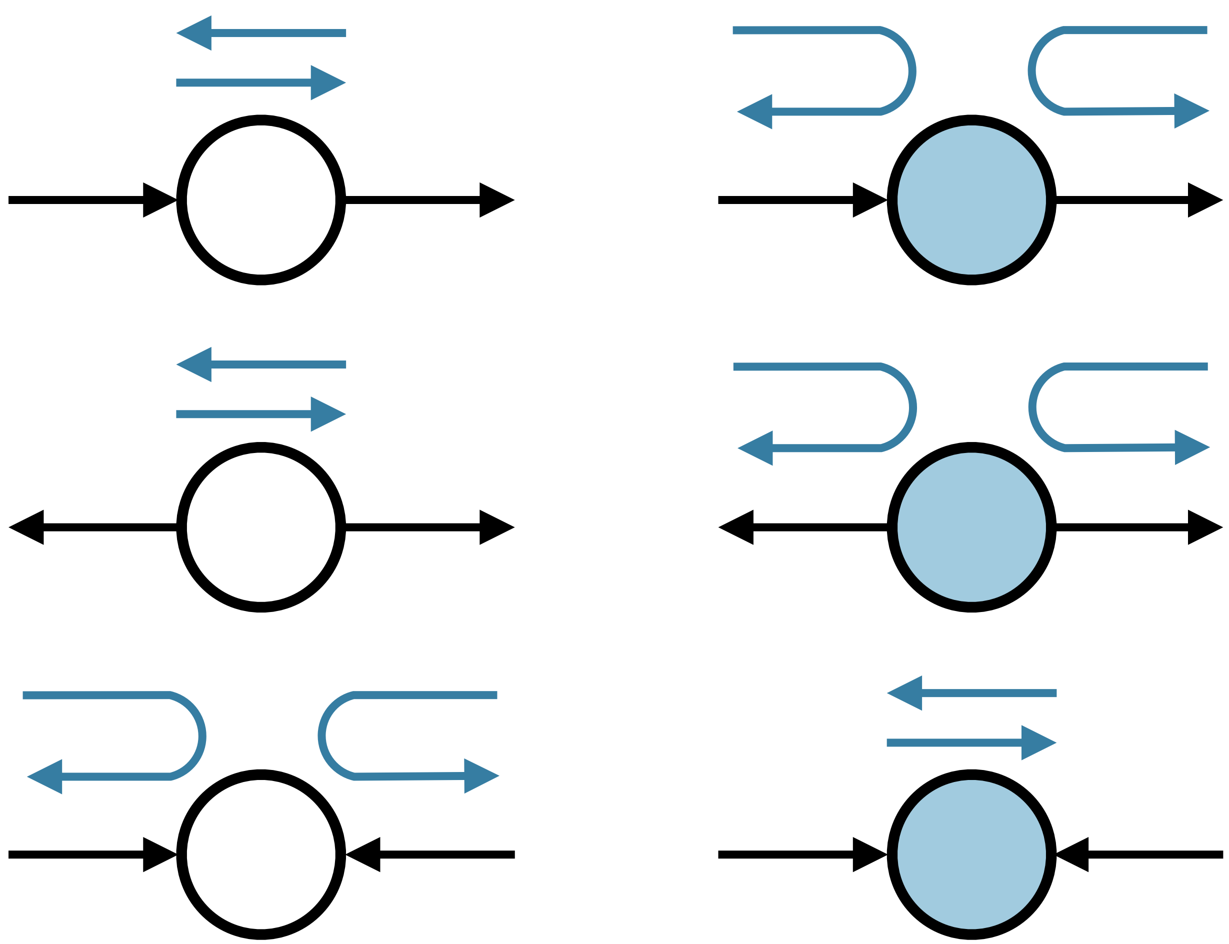
Imagine a ball, **Bayes ball**. This ball is allowed to "travel" on our model, but only in certain **allowed ways**:



Two variables are conditionally independent when Bayes ball cannot travel from one to the other

D-Separation through **Bayes Ball**

Imagine a ball, **Bayes ball**. This ball is allowed to "travel" on our model, but only in certain **allowed ways**:



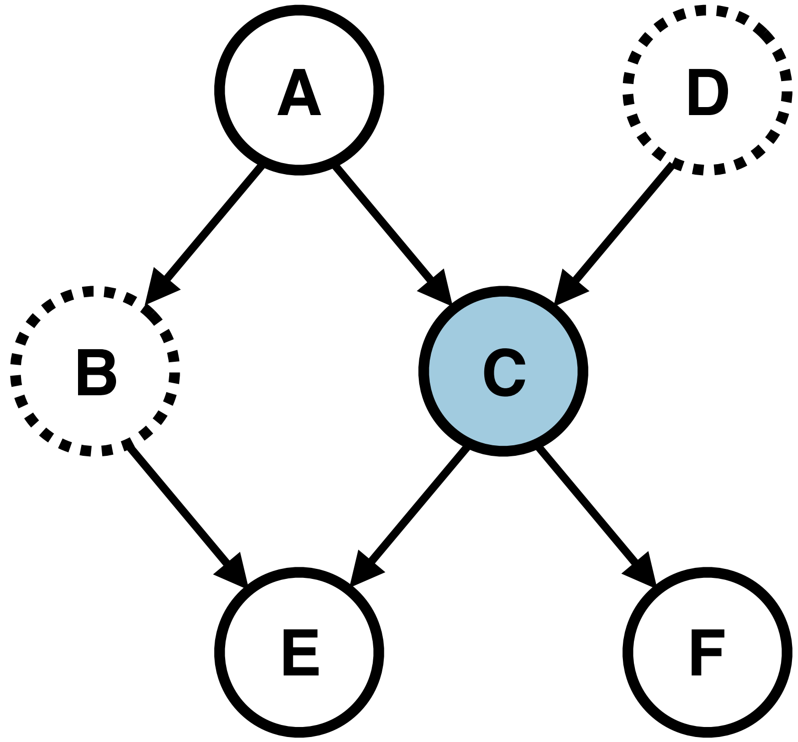
Case #1: **Head to Tail**

Case #2: **Tail to Tail**

Case #3: **Head to Head**

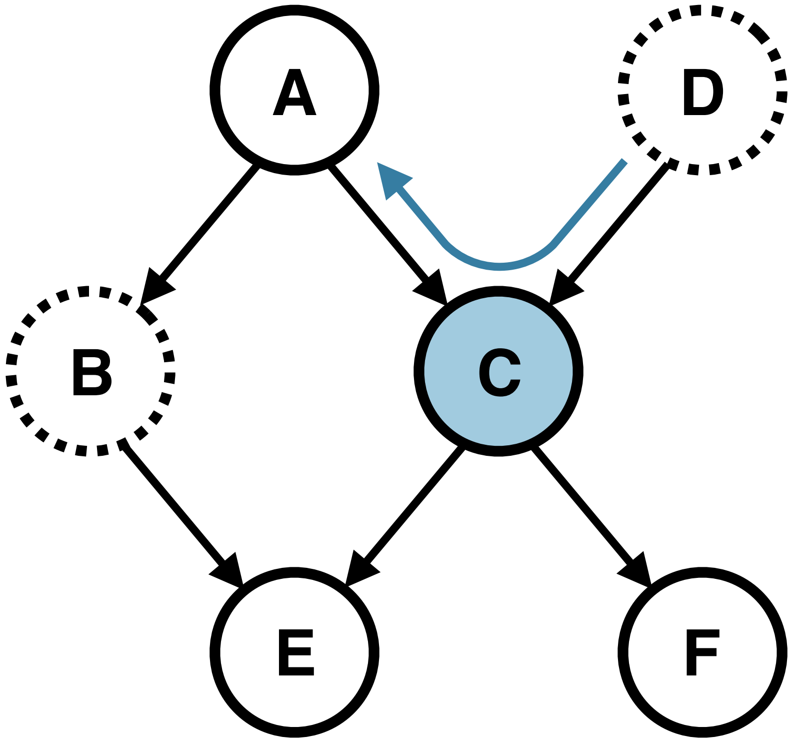
Back to our Example

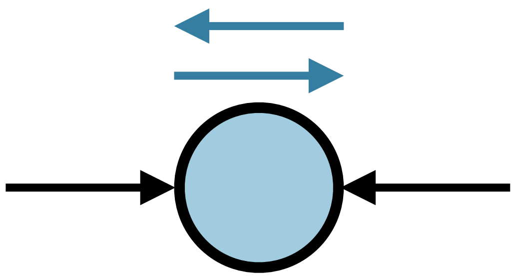
Is **B** independent of **D** given **C**?



Back to our Example

Is **B** independent of **D** given **C**?



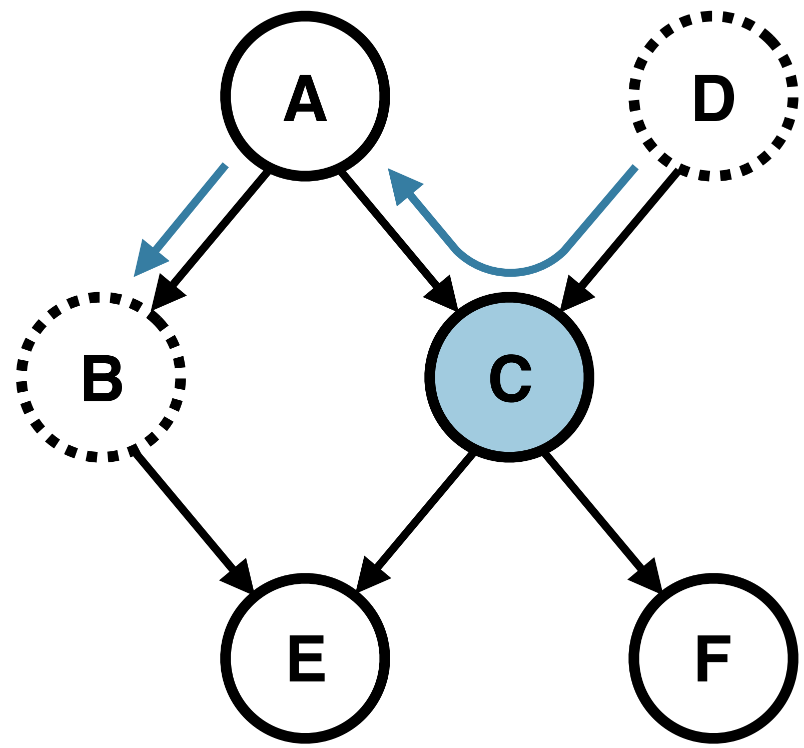


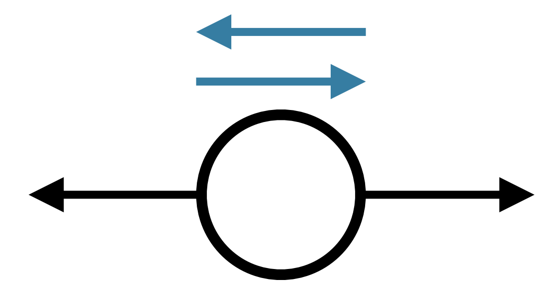
The **blue** arrows correspond to the ball path

Back to our Example

Is **B** independent of **D** given **C**?

Bayes ball could reach **D** from **B** and thus the answer is **NO**

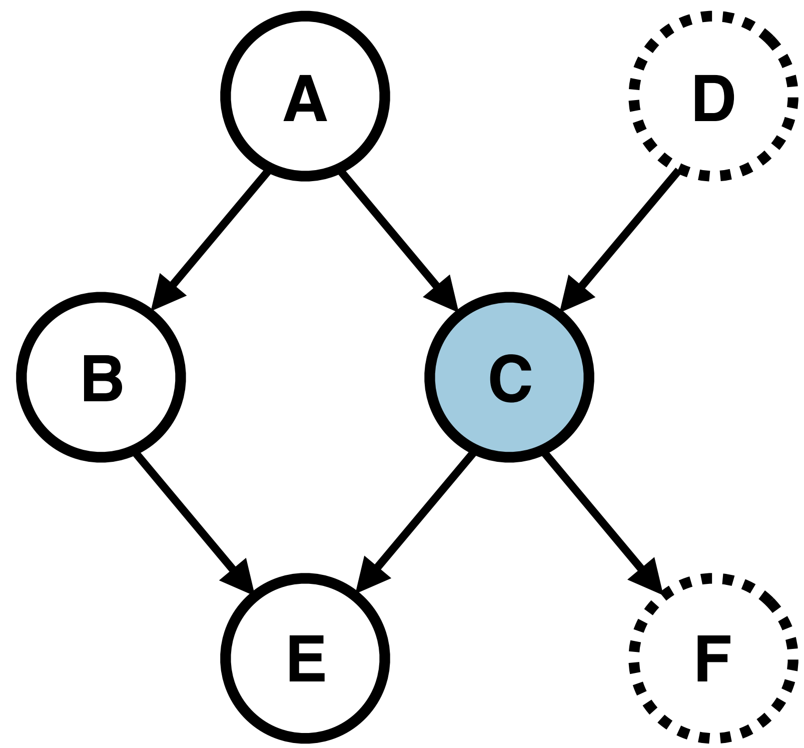




The **blue** arrows correspond to the ball path

Back to our Example

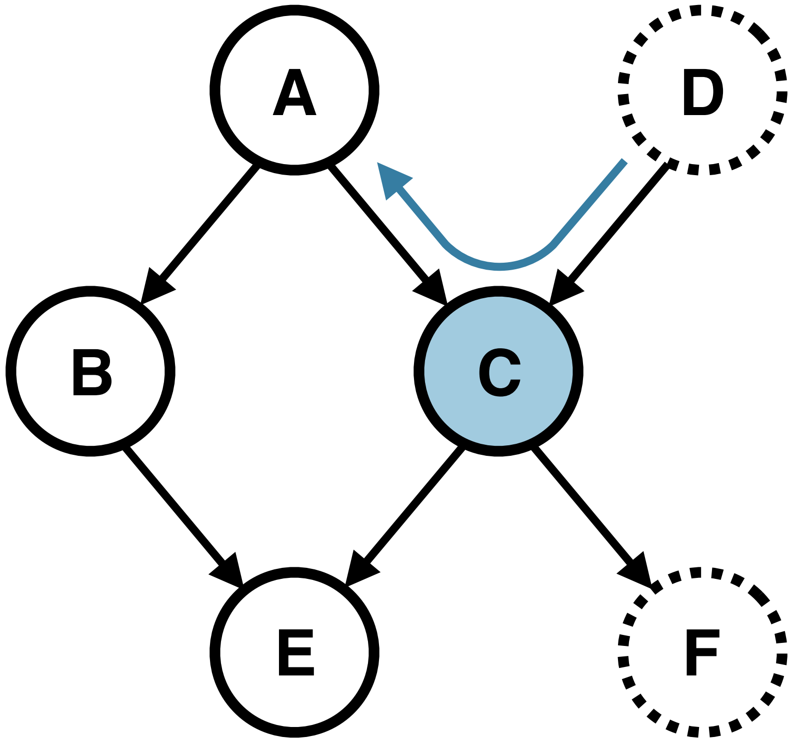
Is **F** independent of **D** given **C**?

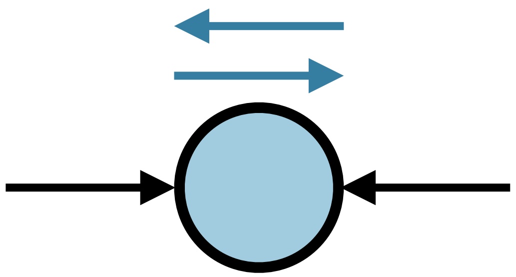


The **blue** arrows correspond to the ball path

Back to our Example

Is **F** independent of **D** given **C**?

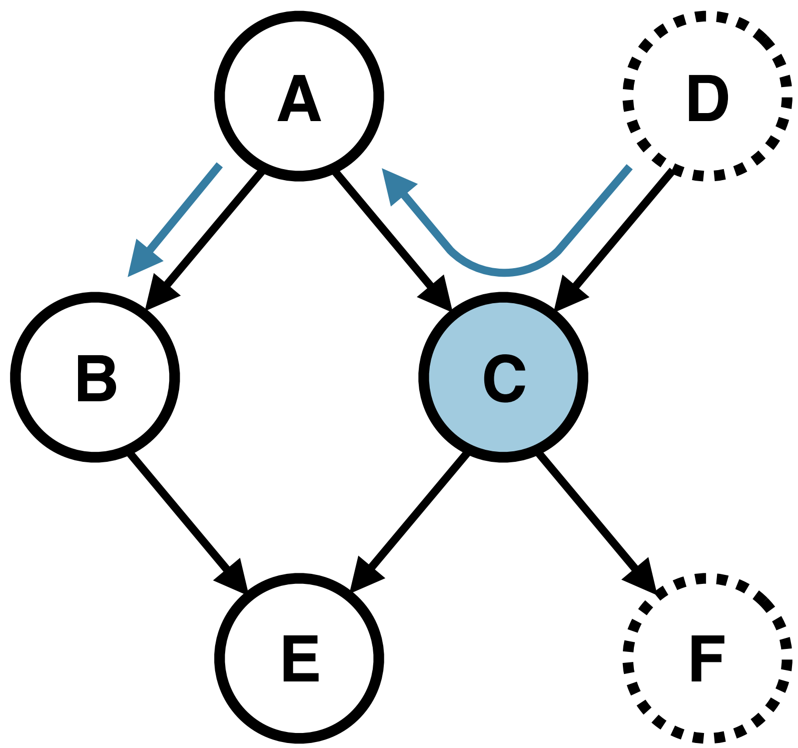


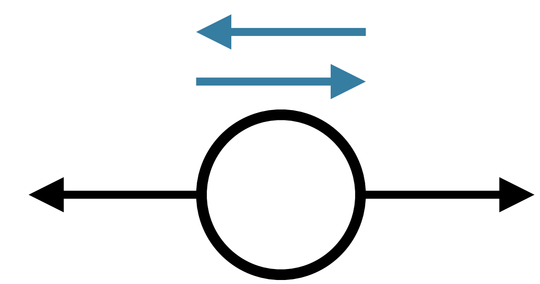


The **blue** arrows correspond to the ball path

Back to our Example

Is **F** independent of **D** given **C**?



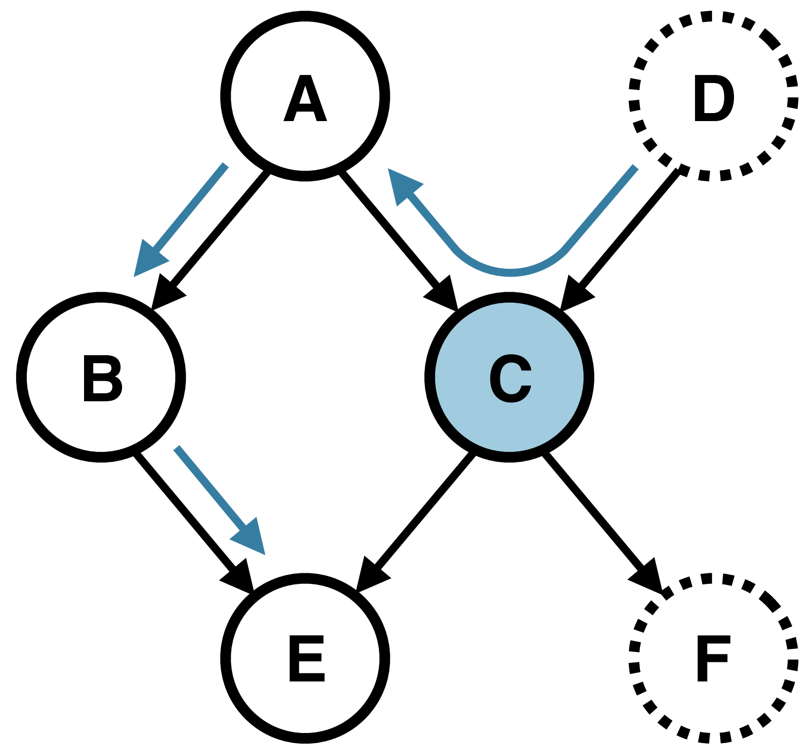


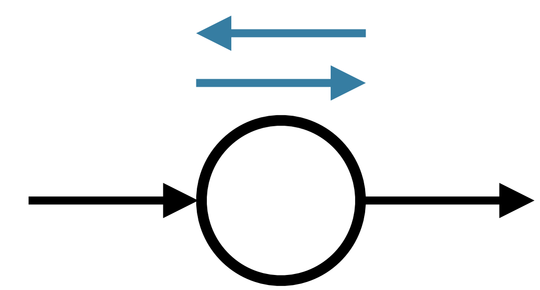
The **blue** arrows correspond to the ball path

Back to our Example

Is **F** independent of **D** given **C**?

Bayes ball got stuck going from **D** from **F** and thus the answer is **YES**

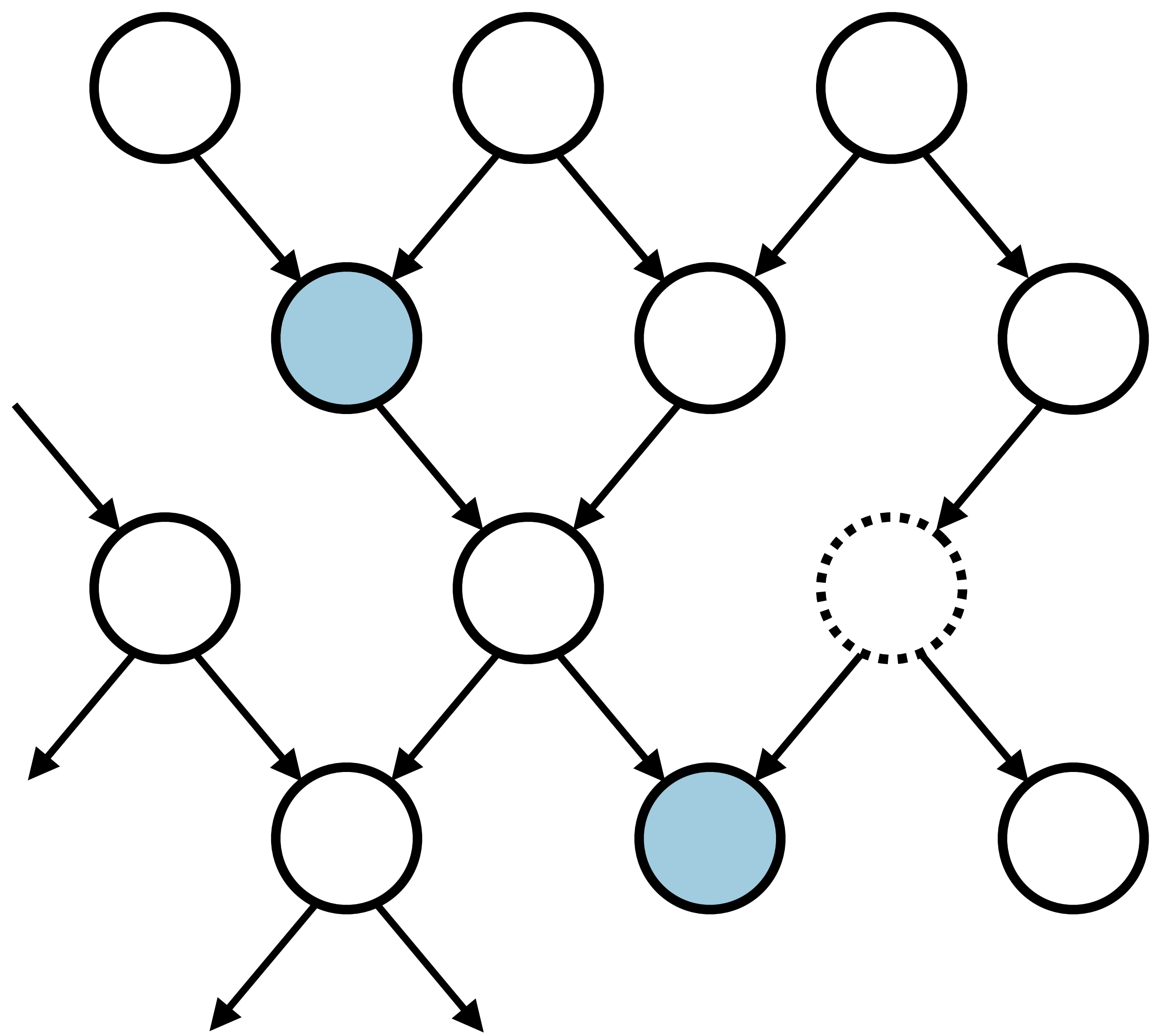




The **blue** arrows correspond to the ball path

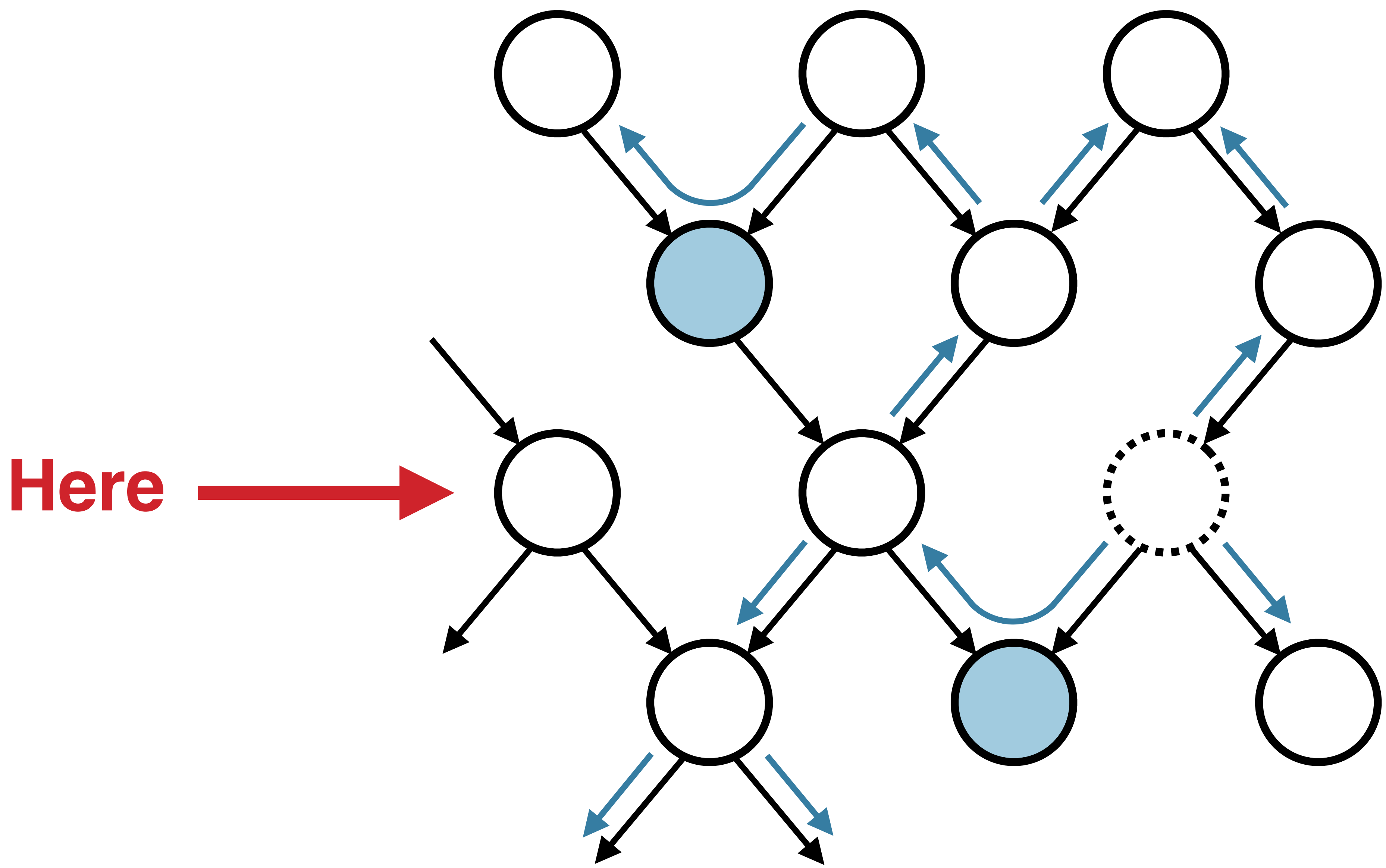
A More Complicated Example

Where can Bayes ball not go to, starting from the dashed variable?

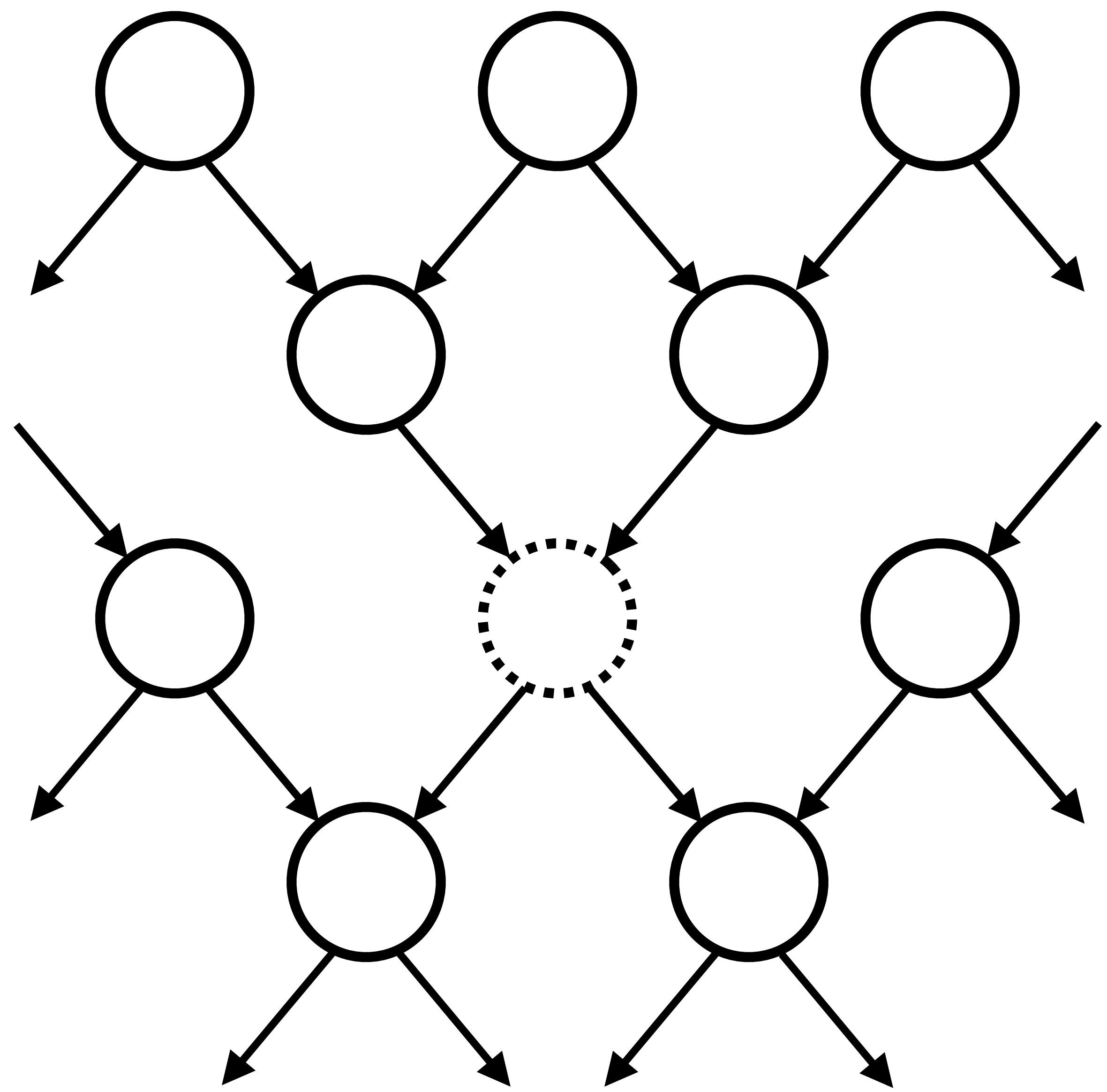


A More Complicated Example

Where can Bayes ball not go to, starting from the dashed variable?

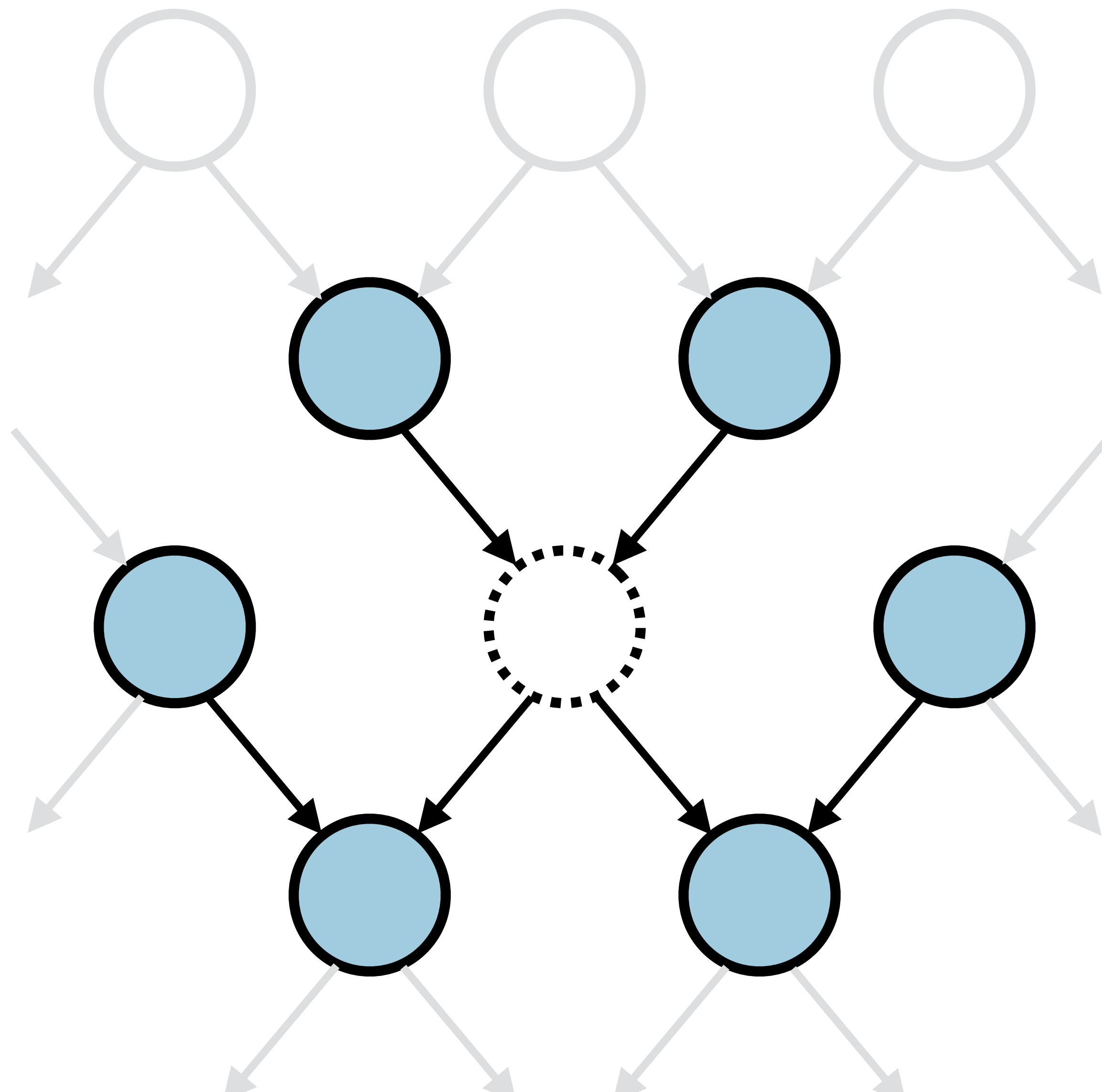


Markov Blanket



Markov Blanket

A variable is independent of every other variable in the model, given its **parents**, its **children**, and its **children's parents**



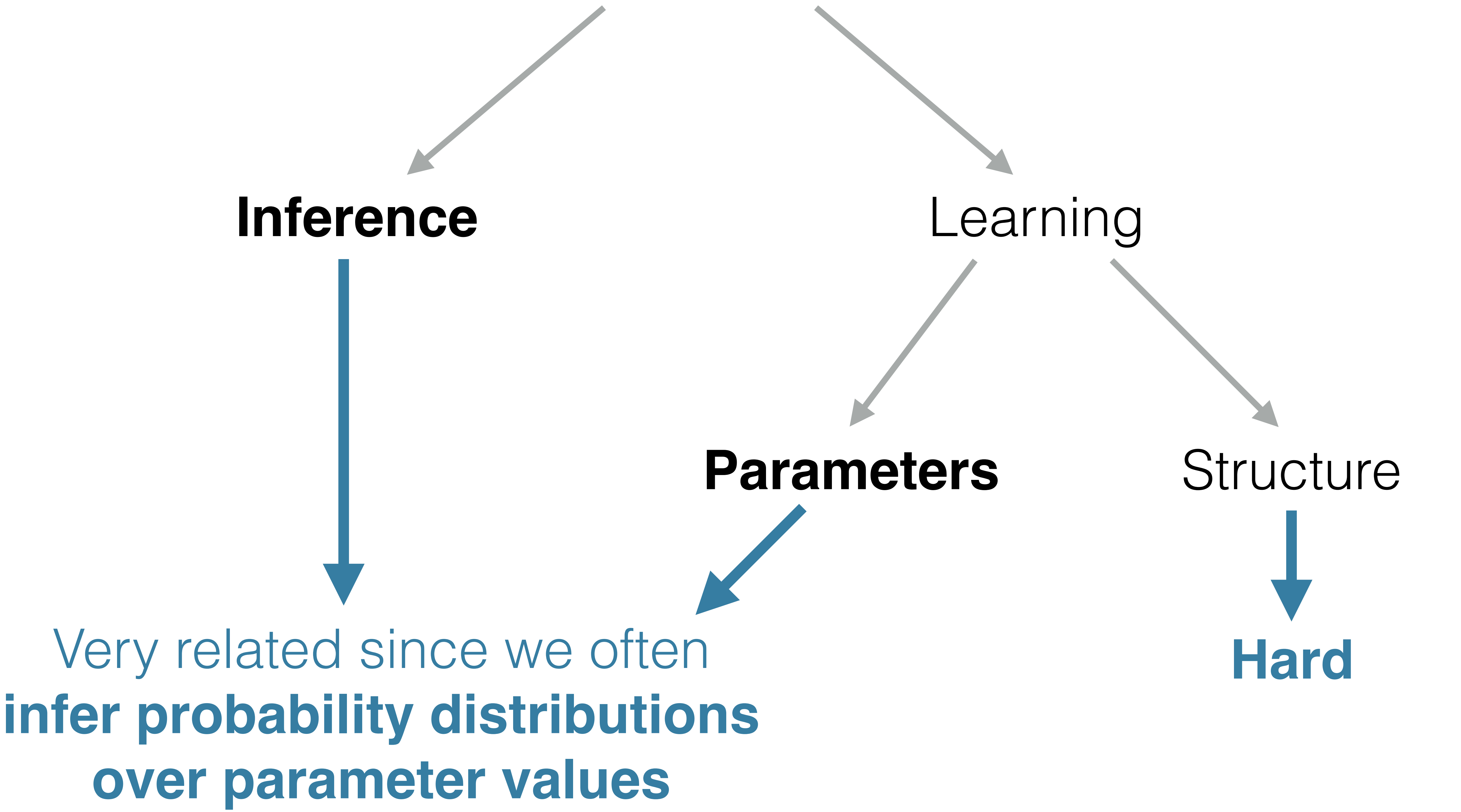
Bayes ball is stuck!!!

That is called the **Markov blanket** of a variable

Dependencies…so what?

We talked so much about understanding when there are dependencies/independencies between variables and how to model them, but **why are they important?**

Graphical Model Problems



Inference in Graphical Models

Probability of joint assignment over all **n** variables is **easy** — we just have to lookup the conditional probability tables (O(n) complexity)

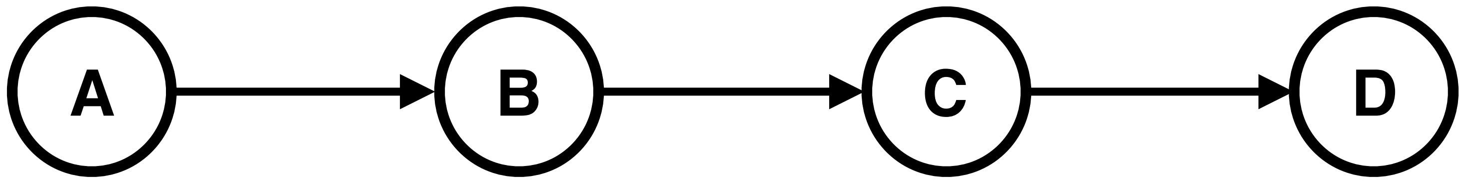
Marginal probability distribution of a single variable is generally **hard** — we need to sum over all possible assignments to the rest of the variables ($\mathcal{O}(|\mathcal{D}|^n)$ complexity)

$$\mathbb{M}(X_i) = \sum_{X_1 \in \mathcal{D}_1} \cdots \sum_{X_{i-1} \in \mathcal{D}_{i-1}} \sum_{X_{i+1} \in \mathcal{D}_{i+1}} \cdots \sum_{X_n \in \mathcal{D}_n} \mathbb{P}(X_1, \ldots, X_n)$$

Conditional independencies help us "move those sums around" and reduce complexity

Marginal Example

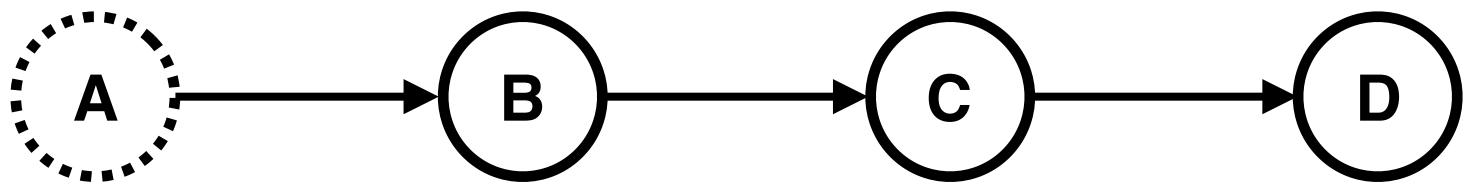
Linear Chain



$$\mathbb{P}(A, B, C, D) = \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(C \mid B)\mathbb{P}(D \mid C)$$

Marginal Example

Linear Chain



$$\mathbb{P}(A, B, C, D) = \mathbb{P}(A)\mathbb{P}(B \mid A)\mathbb{P}(C \mid B)\mathbb{P}(D \mid C)$$

$$\mathbb{M}(A) = \mathbb{P}(A) \sum_{B \in \mathrm{Val}(B)} \mathbb{P}(B \mid A) \sum_{C \in \mathrm{Val}(C)} \mathbb{P}(C \mid B) \sum_{D \in \mathrm{Val}(D)} \mathbb{P}(D \mid C)$$

Each step costs $|\mathcal{D}|^2$ operations and so the complexity is now quadratic:

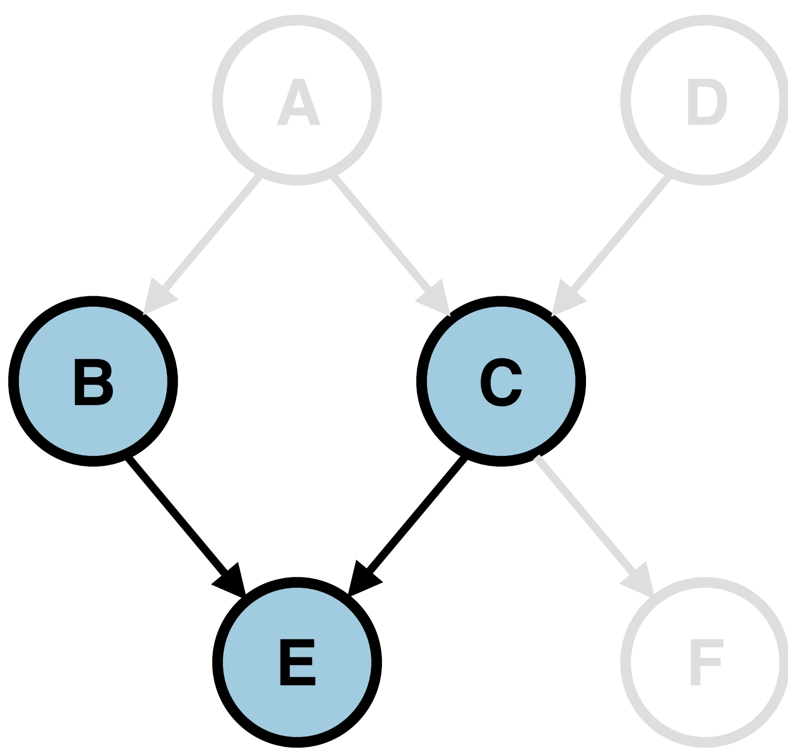
$$\mathcal{O}(n|\mathcal{D}|^2)$$

Learning in Graphical Models

	Fully Observed Variables	Partially Observed Variables
Known Structure	**Easy**	**Interesting** **Frequent**
Unknown Structure	**Doable**	**Hard**

Learning in Graphical Models

We want to estimate the model parameters for a **known structure** and with **fully observed data**



Easy

Let $\mathbb{P}(E = e \mid B = b, C = c) \triangleq \theta_{e|b,c}$

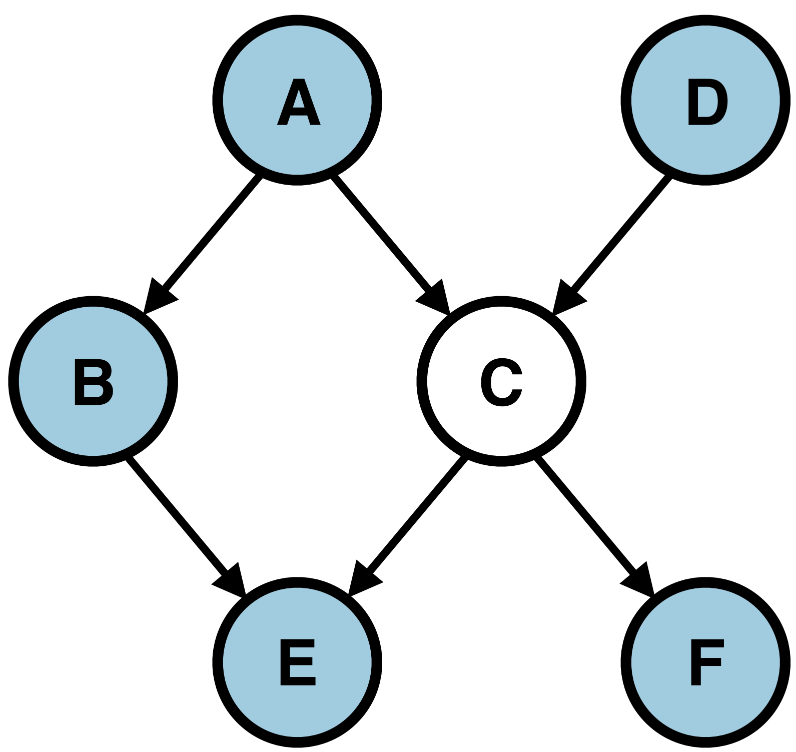
$$\log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta}) = \sum_{k=1}^{K} \log \mathbb{P}(a_k) + \log \mathbb{P}(b_k \mid a_k) + \log \mathbb{P}(d_k) + \log \mathbb{P}(c_k \mid a_k, d_k) + \log \mathbb{P}(e_k \mid b_k, c_k) + \log \mathbb{P}(f_k \mid c_k)$$

$$\frac{\partial \log \mathbb{P}(\mathcal{D} \mid \boldsymbol{\theta})}{\partial \theta_{e|b,c}} = 0 \Rightarrow$$

$$\hat{\theta}_{e|b,c} = \frac{\sum_{k=1}^{K} \mathbb{1}_{e_k = e \wedge b_k = b \wedge c_k = c}}{\sum_{k=1}^{K} \mathbb{1}_{b_k = b \wedge c_k = c}}$$

Learning in Graphical Models

We now want to estimate the model parameters for a **known structure** and with **partially observed data**



Before

$$\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta}} \log \mathbb{P}(A, B, C, D, E, F \mid \boldsymbol{\theta})$$

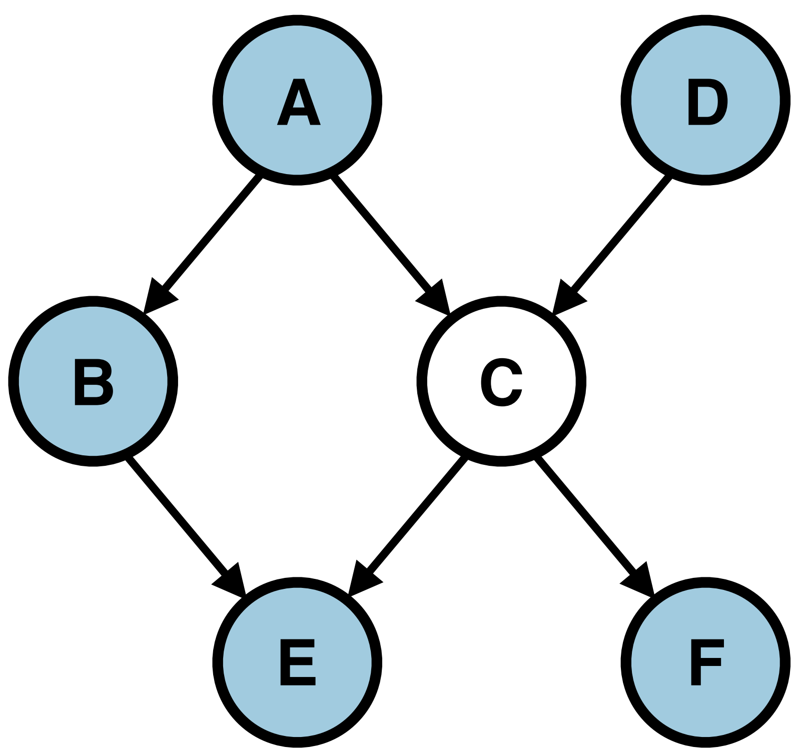
Now

$$\boldsymbol{\theta} \leftarrow \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{C|A,D}\{\log \mathbb{P}(A, B, C, D, E, F \mid \boldsymbol{\theta}\}$$

Expectation-Maximization (EM) Algorithm

EM Algorithm

We begin with an arbitrary choice for our parameters and iterate over the following steps, until convergence



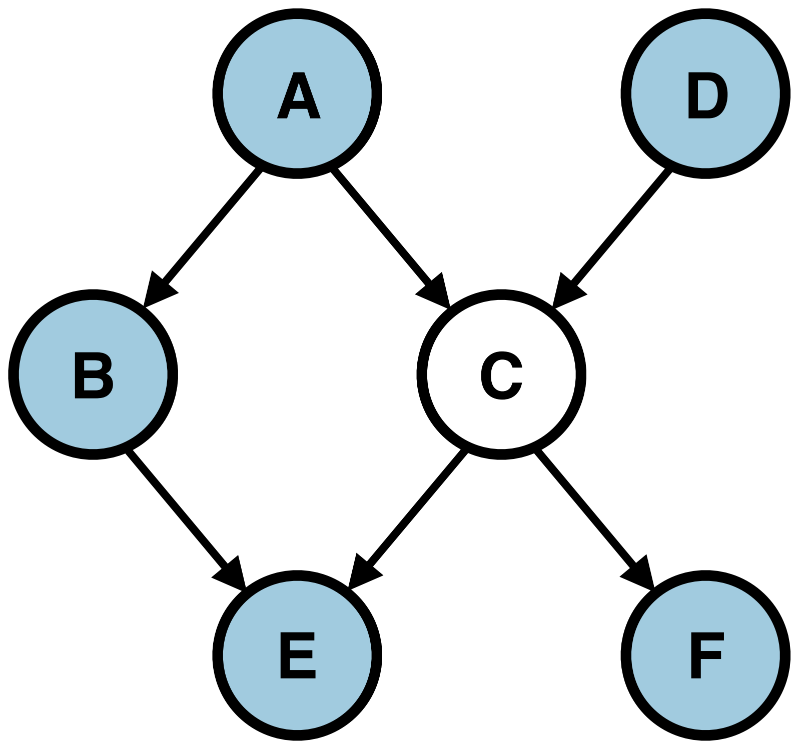
Guaranteed to find a local maximum

E Step: Estimate the values of the unobserved variables using the current parameters

M Step: Use the observed variables along with our estimates of the unobserved variables from the previous E step to compute a maximum likelihood estimate for our parameters and update them

EM Algorithm

We begin with an arbitrary choice for our parameters and iterate over the following steps, until convergence



E Step

$$\begin{aligned}\mathbb{E}\{C \mid A = a, D = d\} &= \mathbb{P}(C = 1 \mid A = a, D = d) \\ &= \theta_{C=1|a,d}\end{aligned}$$

M Step

$$\begin{aligned}\hat{\theta}_{c|a,d} &= \frac{\sum_{k=1}^{K} \mathbb{1}_{c_k=1 \wedge c_k=c \wedge d_k=d} \mathbb{E}\{C \mid A = a_k, D = d_k\}}{\sum_{k=1}^{K} \mathbb{1}_{a_k=a \wedge d_k=d}} \\ &= \frac{\sum_{k=1}^{K} \mathbb{1}_{c_k=1 \wedge c_k=c \wedge d_k=d} \theta_{C=1|a,d}}{\sum_{k=1}^{K} \mathbb{1}_{a_k=a \wedge d_k=d}}\end{aligned}$$

Guaranteed to find a local maximum

EM Algorithm

In the simple **discrete variable** case, we do the same thing as in **maximum likelihood estimation**, but we simply **replace each unobserved variable count by its expected count**

EM is **not exact**, but **why is it guaranteed to find a local maximum?**

EM Algorithm

Intuition: The following holds for any distribution $q(Z)$

$$\log p(X \mid \theta) = \mathcal{L}(q, \theta) + \mathrm{KL}(q \mid\mid p)$$

$$\mathcal{L}(q, \theta) = \sum_Z q(Z) \log \frac{p(X, Z \mid \theta)}{q(Z)}$$

$$\mathrm{KL}(q \mid\mid p) = -\sum_Z q(Z) \log \frac{p(Z \mid X, \theta)}{q(Z)}$$

→ **KL Divergence** (non-negative from homework)

X : observed variables

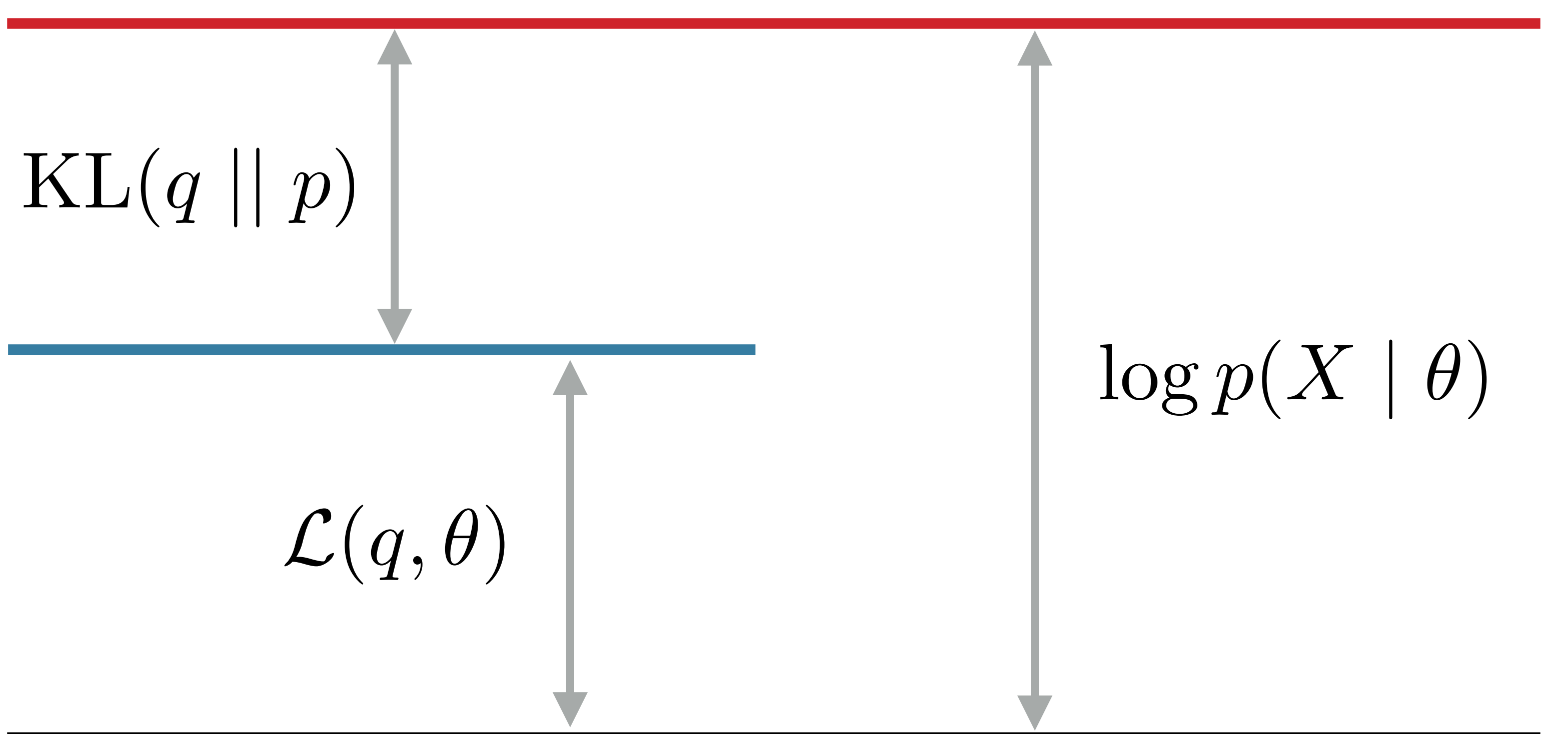
Z : unobserved variables

EM Algorithm

Intuition: The following holds for any distribution $q(Z)$

$$\log p(X \mid \theta) = \mathcal{L}(q, \theta) + \mathrm{KL}(q \mid\mid p)$$

$$\mathcal{L}(q, \theta) = \sum_{Z} q(Z) \log \frac{p(X, Z \mid \theta)}{q(Z)} \leq \log p(X \mid \theta)$$



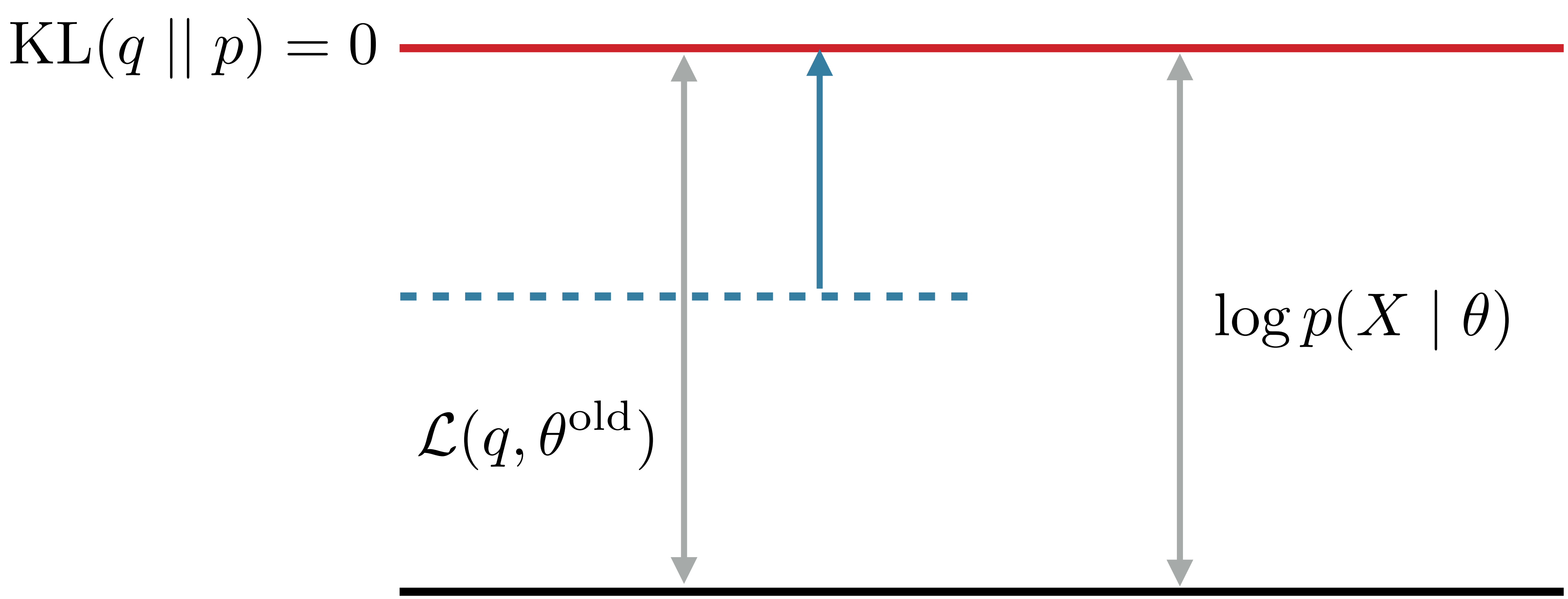
Visualization idea borrowed from Chris Bishop's book

EM Algorithm

E Step Intuition: We maximize $\mathcal{L}(q, \theta^{\text{old}})$ while holding θ^{old} fixed. We thus set

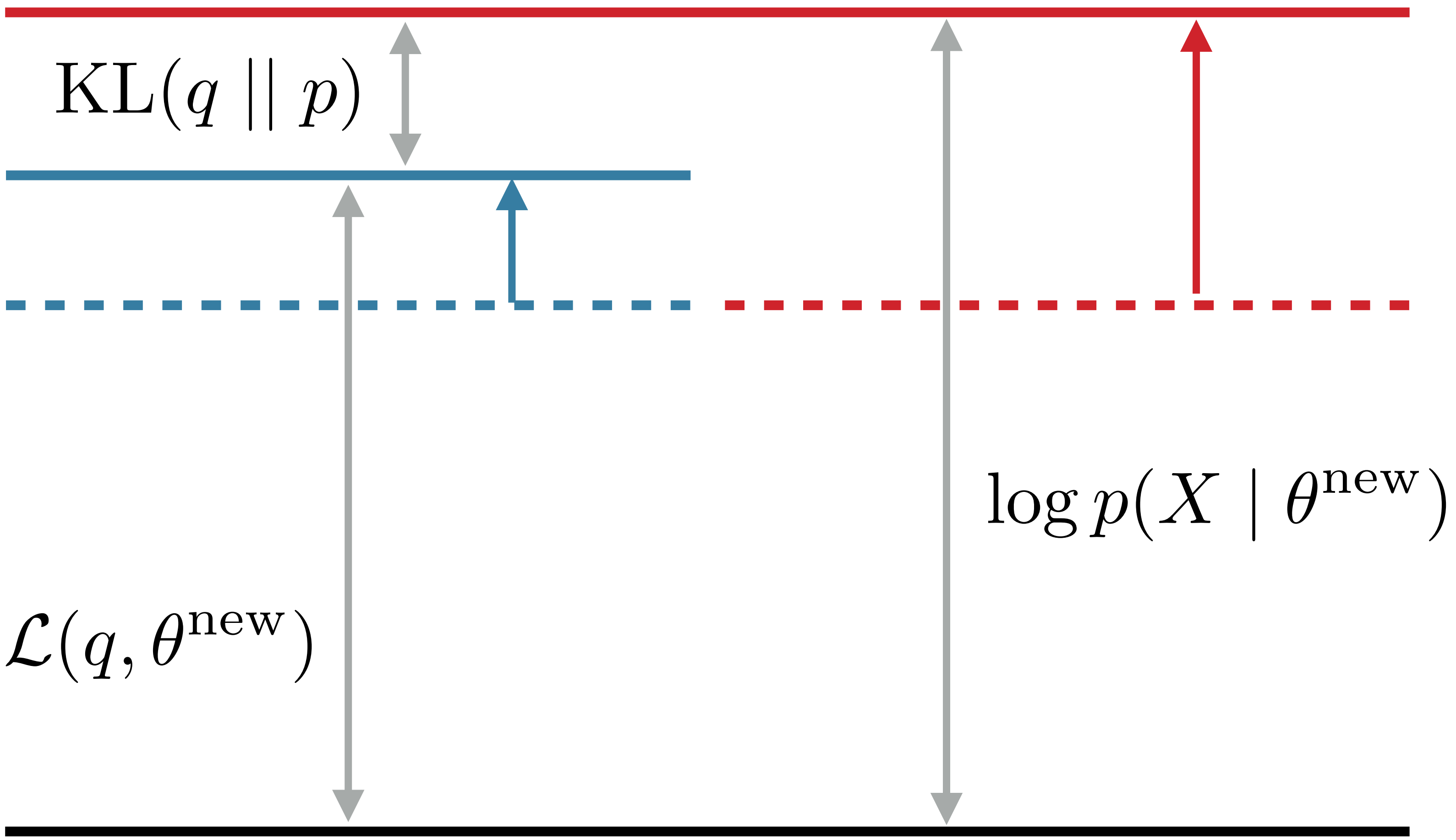
$$q(Z) = p(Z \mid X, \theta^{\text{old}})$$

Remember that $\log p(X \mid \theta) = \mathcal{L}(q, \theta) + \mathrm{KL}(q \,||\, p)$



EM Algorithm

M Step Intuition: We maximize the lower bound with respect to θ while holding $q(Z)$ fixed



Markov Chain Monte Carlo (MCMC)

Conditional probability distributions when dealing with a **known structure and partially observed variables** can be computed using **sampling** — **Markov Chain Monte Carlo (MCMC) methods are often used**

Gibbs sampling is a very common such method

- Initialize unobserved variables to some values
- Sample each unobserved variable in sequence, while fixing the rest to their last sampled value
- **Burn** (i.e., throw away) the first few samples and then **thin** the rest (e.g., keep every 10^{th} sample)

The distribution of the samples converges to the true posterior distribution of the unobserved variables

Bayesian Network **Structure Learning**

Learning structure is not that easy

- In general requires lots of data (can **overfit** easily)
- **Huge search space** — we use priors to constrain it

But there exist some algorithms for certain special cases

e.g., **Chow-Liu** for **tree structures**

Next time
Finds the tree structure that
minimizes KL divergence
(i.e., mutual information)

Questions?