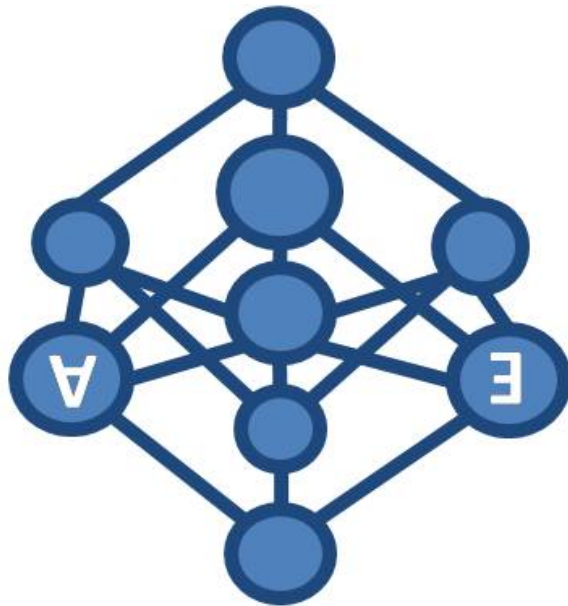


# Probabilistic Graphical Models\*

## Bayesian Networks - Learning



TECHNISCHE  
UNIVERSITÄT  
DARMSTADT



\*Thanks to Carlos Guestrin, Pedro Domingos and many others for making their slides publically available



# What you need to know so far about learning

- Different Learning Settings
  - Fully vs. partially observable variables
  - Structure known vs. unknown
  - Latent variables vs. scientific discovery
- Maximum Likelihood and Decomposable Scores
- EM and Gradient-based ML parameter estimation
  - Update rules for binomial and multinomial vars

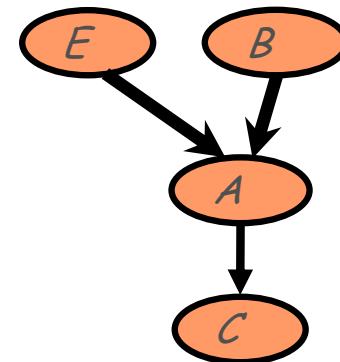
# What's next

- Bayesian Parameter Estimation

# Bayesian Inference

- Recall, training data has the form:

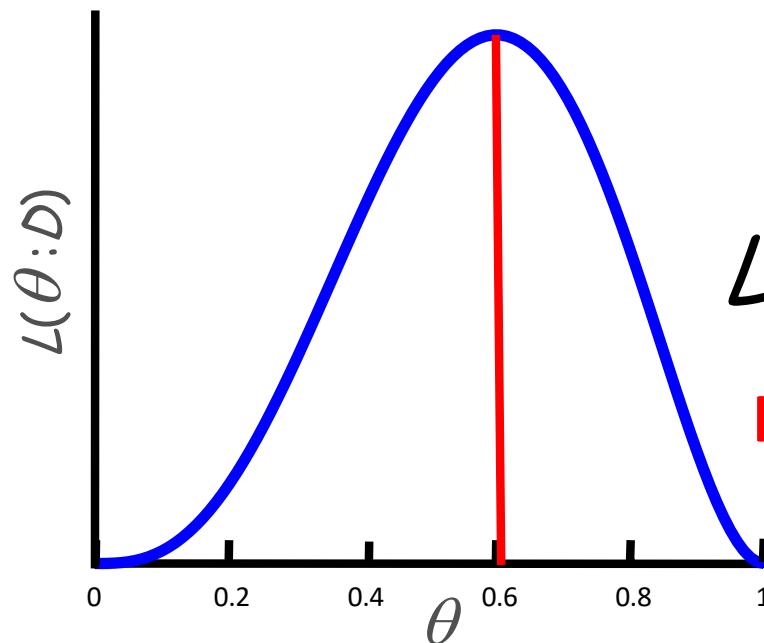
$$D = \begin{bmatrix} E[1] & B[1] & A[1] & C[1] \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ E[M] & B[M] & A[M] & C[M] \end{bmatrix}$$



# Recall, Likelihood Function for Multinomials

$$L(\theta : D) = P(D | \theta) = \prod_m P(x[m] | \theta)$$

- The likelihood for the sequence H, T, T, H, H is



$$L(\theta : D) = \theta \cdot (1 - \theta) \cdot (1 - \theta) \cdot \theta \cdot \theta$$

head is more likely

General case:

$$L(\Theta : D) = \prod_{k=1}^K \theta_k^{N_k}$$

Count of  $k^{\text{th}}$  outcome in D

Probability of  $k^{\text{th}}$  outcome

$$LL(\Theta : D) = \sum_{k=1}^K N_k \cdot \log \theta_k$$

## Now, what is Bayesian Inference?

- In contrast to MLE, it **represents uncertainty about parameters using a probability distribution over parameters and data**
- It is **“Learning by inference” using Bayes’ rule**

$$P(\theta \mid x[1], \dots, x[M]) = \frac{P(x[1], \dots, x[M] \mid \theta) P(\theta)}{P(x[1], \dots, x[M])}$$

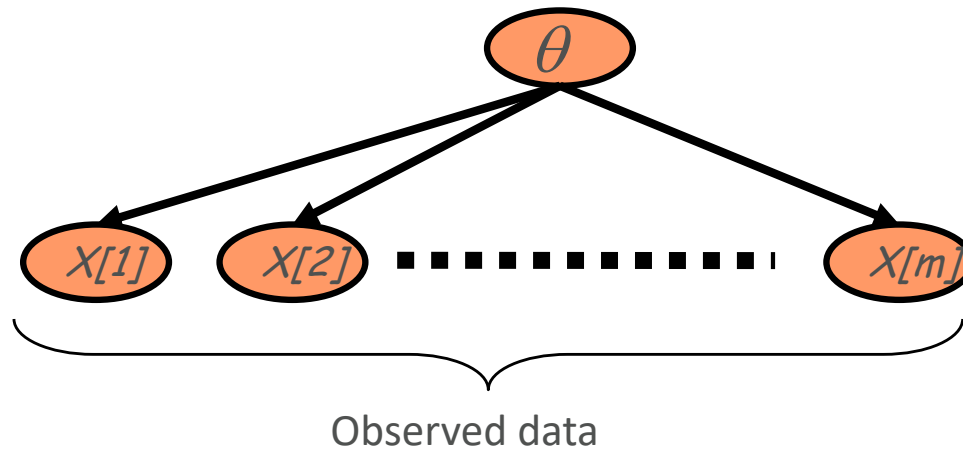
Diagram illustrating Bayes' rule with labels:

- Likelihood**: Points to the numerator term  $P(x[1], \dots, x[M] \mid \theta)$ .
- Prior of parameters**: Points to the numerator term  $P(\theta)$ .
- Posterior of parameters**: Points to the left side of the equation,  $P(\theta \mid x[1], \dots, x[M])$ .
- Probability of data**: Points to the denominator term  $P(x[1], \dots, x[M])$ .



# Bayesian Inference (for the Heads/Tails Example)

- Represent Bayesian distribution as Bayesian net



- Thus, the values of  $X$  are independent given  $\theta$  (which we do not know though)

$$P(x[m] / \theta) = \theta \text{ (keep this in mind!)}$$

- Bayesian prediction is inference in this network



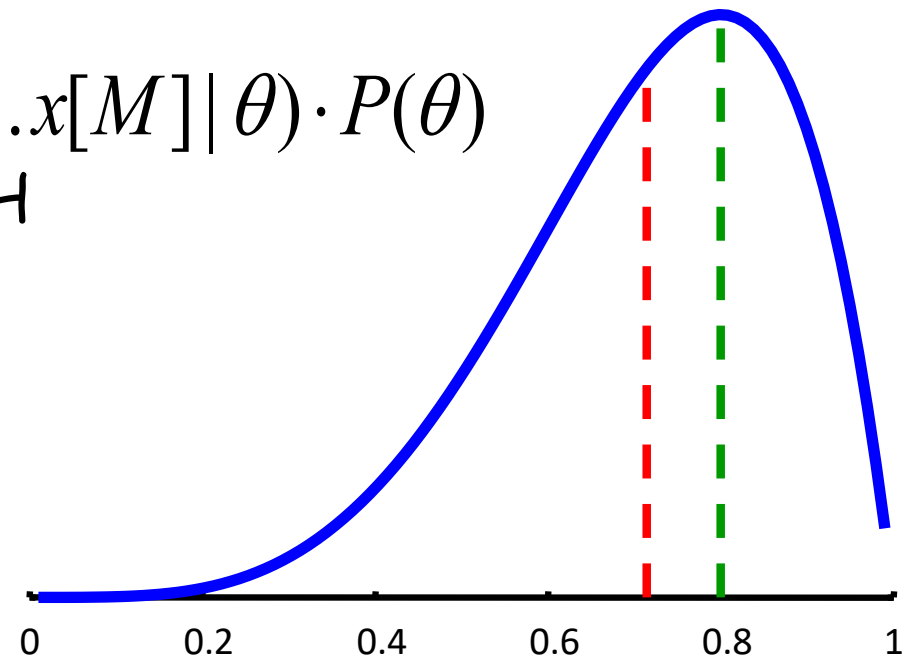
## Example: Binomial Data

- Prior: uniform for  $\theta$  in  $[0,1]$   
 $\Rightarrow P(\theta | D) \propto$  the likelihood  $L(\theta : D)$

$$P(\theta | x[1], \dots, x[M]) \propto P(x[1], \dots, x[M] | \theta) \cdot P(\theta)$$

$(N_H, N_T) = (4, 1)$ , e.g., H, T, H, H, H

- MLE for  $P(X = H)$  is  $4/5 = 0.8$
- Bayesian prediction is



$$P(x[M+1] = H | D) = \int \theta \cdot P(\theta | D) d\theta = \frac{5}{7} = 0.7142 \dots$$

We do not know theta! So, we have to integrate over all values. Bayesian prediction = Inference in the extended network



# Why is this the case?

We start with

- $P(\theta)$  - **prior distribution** about the values of  $\theta$
- $P(x_1, \dots, x_n | \theta)$  - **likelihood** of examples given a known value  $\theta$

Given examples  $x_1, \dots, x_n$ , we can compute **posterior distribution** on  $\theta$

$$P(\theta | x_1, \dots, x_n) = \frac{P(x_1, \dots, x_n | \theta) P(\theta)}{P(x_1, \dots, x_n)}$$

Where the **marginal likelihood** is

$$P(x_1, \dots, x_n) = \int P(x_1, \dots, x_n | \theta) P(\theta) d\theta$$



# Binomial Distribution aka Laplace Est.

- In this case the unknown parameter is  $\theta = P(H)$
- Simplest **prior**  $P(\theta) = 1$  for  $0 < \theta < 1$
- **Likelihood**  $P(x_1, \dots, x_n \mid \theta) = \theta^k (1 - \theta)^{n-k}$   
where  $k$  is number of heads in the sequence
- **Marginal Likelihood:**

$$P(x_1, \dots, x_n) = \int_0^1 \theta^k (1 - \theta)^{n-k} d\theta$$

- Now, we use **integration by parts**




# Recap: Integration by parts

First, start with the Product Rule for differentiation.

$$\frac{d}{dx} (uv) = u \frac{dv}{dx} + v \frac{du}{dx}$$

Integrate both sides of the equation to obtain:


$$\int \frac{d}{dx} (uv) \cancel{dx} = \int u \frac{dv}{dx} \cancel{dx} + \int v \frac{du}{dx} \cancel{dx}$$
$$uv = \int u \, dv + \int v \, du$$

and by manipulating the equation we get...

The Fundamental Theorem of Calculus tells us that if we take the derivative of the integral, then we are left with the original function. The derivative and the integral “cancel” each other out.

$$\int u \, dv = uv - \int v \, du$$

This is the formula for Integration by Parts.

Just as the Substitution Method could be thought of as the Chain Rule for integration, Integration by Parts could also be thought of as the Product Rule for integration.

When using this formula, we need to choose which part of the integrand (what we are taking the integral of) is  $u$ , and which part of it is  $dv$ .

When choosing  $u$  and  $dv$ , we want to find a  $u$  that will be simplified after we take its derivative, and a  $dv$  that won't be too complex after integrating it.



# Marginal Likelihood

$$\int u \, dv = uv - \int v \, du$$

Using **integration by parts** we have:

$$\begin{aligned} P(x_1, \dots, x_n) &= \int_0^1 \underbrace{\theta^k}_{dv} \underbrace{(1-\theta)^{n-k}}_u d\theta \\ &= \left[ \frac{1}{k+1} \theta^{k+1} (1-\theta)^{n-k} \right]_0^1 + \frac{n-k}{k+1} \int_0^1 \theta^{k+1} (1-\theta)^{n-k-1} d\theta \\ &= \frac{n-k}{k+1} \int_0^1 \theta^{k+1} (1-\theta)^{n-k-1} d\theta \end{aligned}$$

Multiply both side by n choose k, we have

$$\binom{n}{k} \int_0^1 \theta^k (1-\theta)^{n-k} d\theta = \binom{n}{k+1} \int_0^1 \theta^{k+1} (1-\theta)^{n-k-1} d\theta$$



# Marginal Likelihood - Cont

- The recursion terminates when (the initial)  $k = n$

Thus

$$\binom{n}{n} \int_0^1 \theta^n (1-\theta)^{n-n} d\theta = \int_0^1 \theta^n d\theta = \frac{1}{n+1}$$

$$P(x_1, \dots, x_n) = \int_0^1 \theta^k (1-\theta)^{n-k} d\theta = \frac{1}{n+1} \binom{n}{k}^{-1}$$

We conclude that the **posterior** is (just plug in all terms and recall that  $P(\theta)=1$ )

$$P(\theta | x_1, \dots, x_n) = \frac{P(x_1, \dots, x_n | \theta) P(\theta)}{P(x_1, \dots, x_n)} = (n+1) \binom{n}{k} \theta^k (1-\theta)^{n-k} \cdot 1$$



$$\underline{P(\theta \mid x_1, \dots, x_n)} = \frac{P(x_1, \dots, x_n \mid \theta)P(\theta)}{P(x_1, \dots, x_n)}$$

## Bayesian Prediction

- So, we have a **posterior over the parameters**. But how do we predict a probability using this posterior?
- We can think of this as computing the probability of the next element in the sequence

$$\begin{aligned} P(x_{n+1} \mid x_1, \dots, x_n) &= \int P(x_{n+1}, \theta \mid x_1, \dots, x_n) d\theta \\ \text{chain rule} \quad &= \int P(x_{n+1} \mid \theta, x_1, \dots, x_n) \underline{P(\theta \mid x_1, \dots, x_n)} d\theta \\ &= \int \underline{P(x_{n+1} \mid \theta)} \underline{P(\theta \mid x_1, \dots, x_n)} d\theta \end{aligned}$$

- Assumption: if we know  $\theta$ , the probability of  $x_{n+1}$  is independent of  $x_1, \dots, x_n$

$$P(x_{n+1} \mid \theta, x_1, \dots, x_n) = \underline{P(x_{n+1} \mid \theta)}$$

# Bayesian Prediction

- Thus, we conclude that

$$\begin{aligned} P(x_{n+1} = H \mid x_1, \dots, x_n) &= \int P(x_{n+1} \mid \theta) P(\theta \mid x_1, \dots, x_n) d\theta \\ &= \int \theta P(\theta \mid x_1, \dots, x_n) d\theta \\ &= (n+1) \binom{n}{k} \int \theta^{k+1} (1-\theta)^{n-k} d\theta \\ &= (n+1) \binom{n}{k} \frac{1}{n+2} \binom{n+1}{k+1}^{-1} = \frac{k+1}{n+2} \end{aligned}$$

Plug-in formula for the posterior and move everything out of the integral that does not depend on theta

Same type of derivation as before





# How do we choose the prior?

- Many possible answers...
- **Pragmatic approach:**
  - Want computationally “simple” (and still flexible) prior: conjugate priors

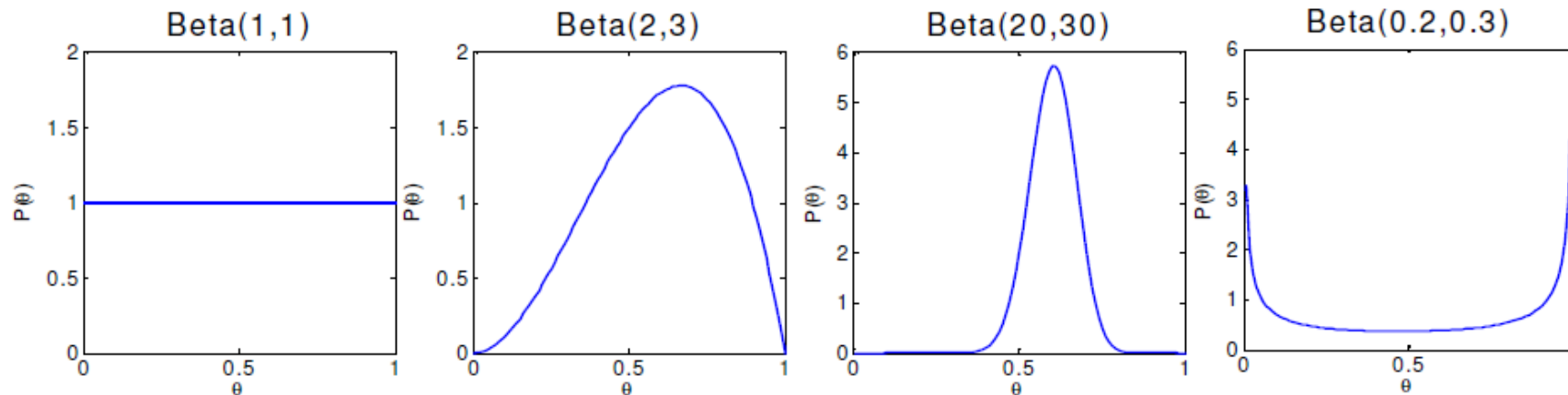
# Conjugate Prior

- Consider parametric families of prior distributions:
  - $P(\theta) = f(\theta; \alpha)$
  - $\alpha$  is called “hyperparameters” of prior
- A prior  $P(\theta) = f(\theta; \alpha)$  is called **conjugate** for a likelihood function  $P(D \mid \theta)$  if  $P(\theta \mid D) = f(\theta; \alpha')$ 
  - Posterior has same parametric form
  - Hyperparameters are updated based on data  $D$

# Conjugate for Binomial

- Beta distribution

$$\text{Beta}(\theta; \alpha_H, \alpha_T) = \frac{\theta^{\alpha_H-1} (1-\theta)^{\alpha_T-1}}{\underbrace{B(\alpha_H, \alpha_T)}_{\text{Normalization constant}}}$$



# Posterior for Binomial

- Beta distribution

$$P(\theta) = \text{Beta}(\theta; \alpha_H, \alpha_T) = \frac{\theta^{\alpha_H-1} (1-\theta)^{\alpha_T-1}}{B(\alpha_H, \alpha_T)}$$

- Likelihood:

$$P(\mathcal{D} \mid \theta) = \theta^{m_H} (1-\theta)^{m_T}$$

- Posterior:

$$P(\theta \mid \mathcal{D}) \propto P(\theta) P(\mathcal{D} \mid \theta) \propto \theta^{\alpha_H+m_H-1} (1-\theta)^{\alpha_T+m_T-1}$$

$$P(\theta \mid \mathcal{D}) = \text{Beta}(\theta; \alpha_H+m_H, \alpha_T+m_T)$$

# Bayesian Prediction

- Prior  $P(\theta) = \text{Beta}(\alpha_H, \alpha_T)$       *Bernoulli:  $P(X=H) = \theta$*
- Suppose we observe  $D = \{m_H \text{ heads, and } m_T \text{ tails}\}$
- What's  $P(X=H \mid D)$ , i.e., prob. that next flip is heads?

$$P(X=H \mid D) = \underbrace{\int \theta P(\theta \mid D) d\theta}_{\text{marginal}} = \mathbb{E}[\theta \mid D] = \frac{\alpha_H + m_H}{\alpha_H + \alpha_T + m_H + m_T}$$

# Prior = Smoothing

$$\mathbb{E}[\theta] = \frac{m_H + \alpha_H}{\underbrace{m_H + m_T}_m + \underbrace{\alpha_H + \alpha_T}_{m'}} = \frac{m_H + \gamma m'}{\underbrace{m + m'}_{(*)}}$$

- Where  $m' = \alpha_H + \alpha_T$ , and  $\gamma = \alpha_H / m'$ ,  $0 \leq \gamma \leq 1$
- $m'$  is called “equivalent sample size”  
 → “hallucinated” coin flips

$$E[\theta] = \frac{m}{m+m'} \underbrace{\frac{m_H}{m}}_{MLE} + \frac{m'}{m+m'} \underbrace{\gamma}_{\text{prior mean}}$$

$m \rightarrow \infty$      $E[\theta] \rightarrow MLE$     Forget prior  
 $m = 0$     prior

→ Interpolate between MLE and prior mean

# Conjugate for Multinomial

- If  $X \in \{1, \dots, k\}$  has  $k$  states:
- Multinomial likelihood

$$P(\mathcal{D} \mid \theta) = \theta_1^{m_1} \theta_2^{m_2} \dots \theta_k^{m_k}$$

where  $\sum_i \theta_i = 1, \theta_i \geq 0$

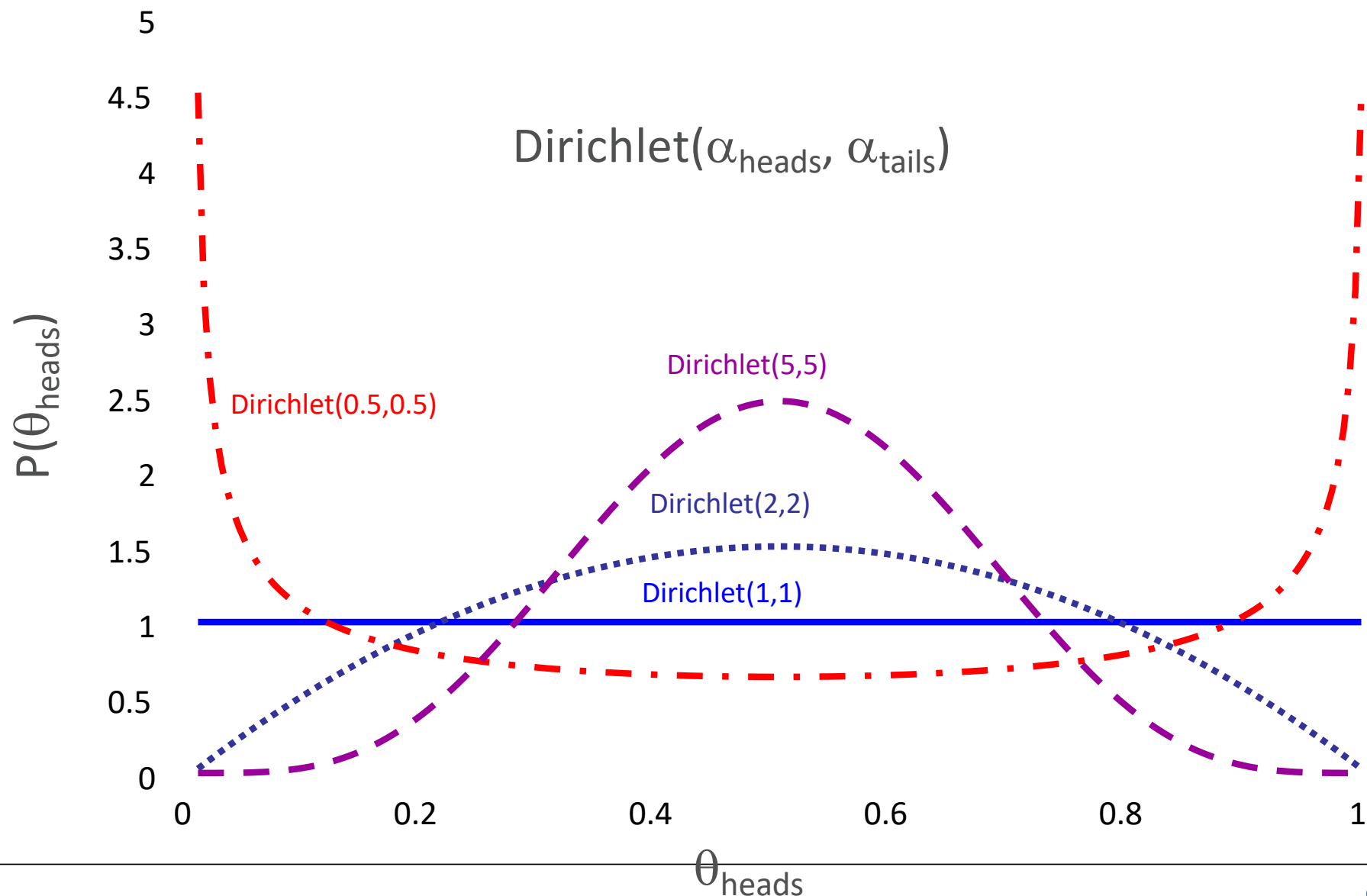
- Conjugate prior: Dirichlet distribution

$$P(\theta) = \text{Dir}(\theta; \alpha_1, \dots, \alpha_k) = \frac{1}{Z} \prod_i \theta_i^{\alpha_i - 1}$$

- If observe  $\mathcal{D} = \{m_1 \text{ 1s}, m_2 \text{ 2s}, \dots, m_k \text{ ks}\}$ , then

$$P(\theta \mid \mathcal{D}) = \text{Dir}(\theta; \alpha_1 + m_1, \dots, \alpha_k + m_k)$$

# Dirichlet Priors - Example





## Dirichlet Priors (cont.)

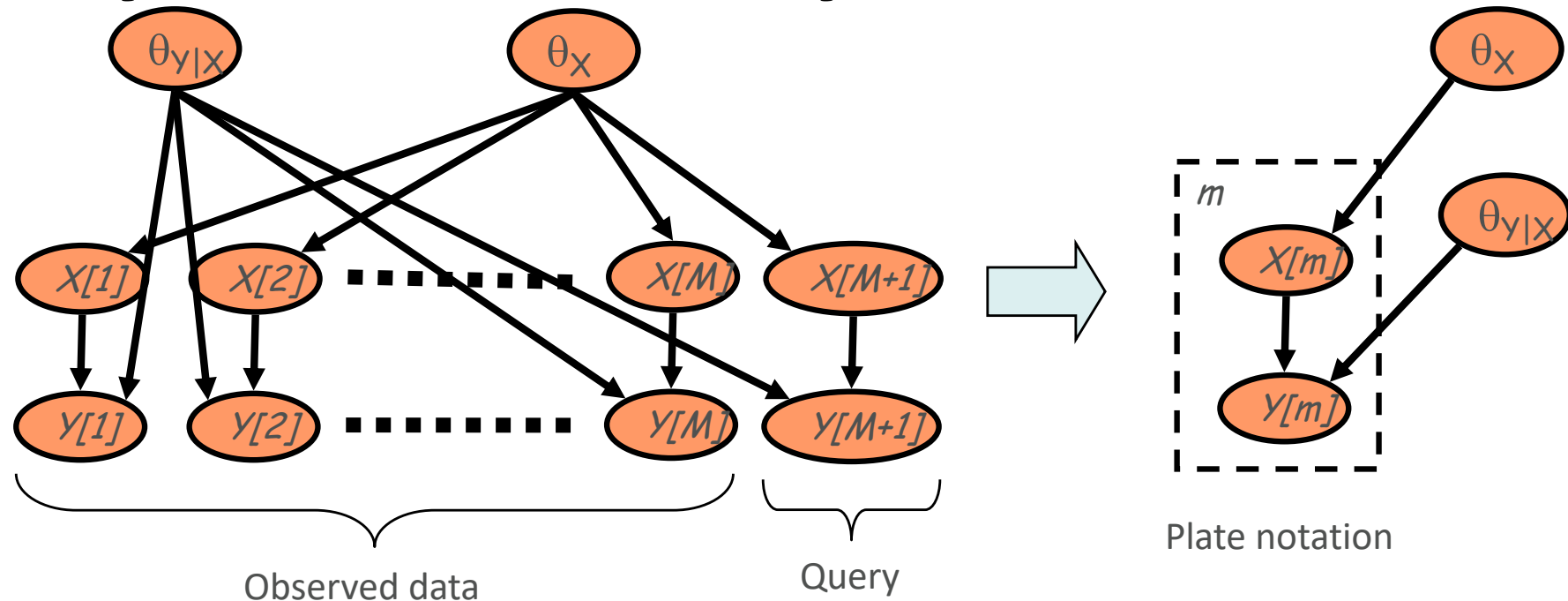
- If  $P(\Theta)$  is Dirichlet with hyperparameters  $\alpha_1, \dots, \alpha_K$

$$P(X[1] = k) = \int \theta_k \cdot P(\Theta) d\Theta = \frac{\alpha_k}{\sum_{\ell} \alpha_{\ell}}$$

- Since the posterior is also Dirichlet, we get

$$P(X[M+1] = k \mid D) = \int \theta_k \cdot P(\Theta \mid D) d\Theta = \frac{\alpha_k + N_k}{\sum_{\ell} (\alpha_{\ell} + N_{\ell})}$$

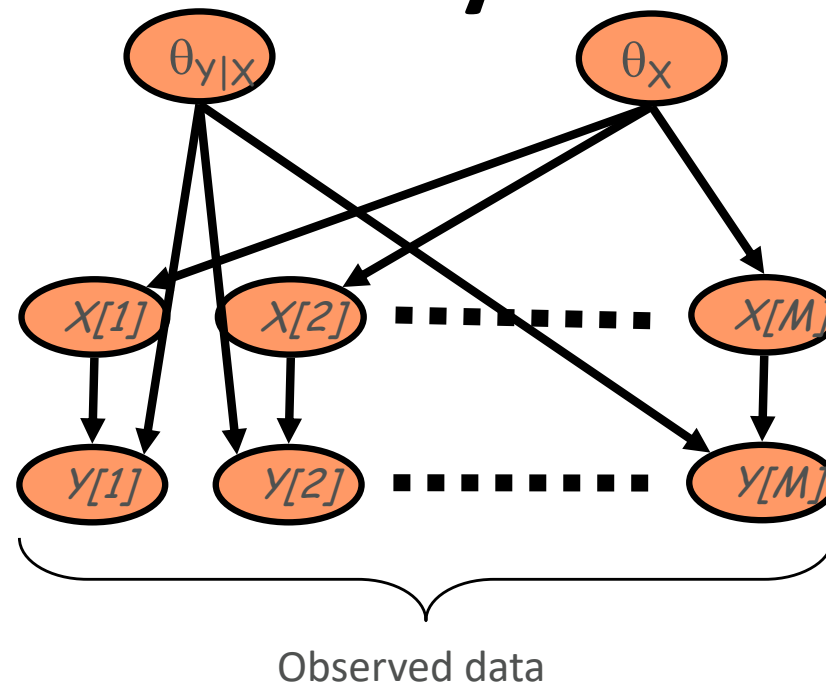
# Bayesian Nets & Bayesian Prediction



- Priors for each parameter group are independent
- Data instances are independent given the unknown parameters (so to say the “Bayesian iid”)



# Bayesian Nets & Bayesian Prediction



- We can also “read” from the network:

**Complete data  $\Rightarrow$**

**posteriors on parameters are independent**

- Can compute posterior over parameters separately!



# Learning Parameters: Summary

- Estimation relies on **sufficient statistics**
  - For multinomials: counts  $N(x_i, pa_i)$
  - Parameter estimation

$$\hat{\theta}_{x_i|pa_i} = \frac{N(x_i, pa_i)}{N(pa_i)}$$

MLE

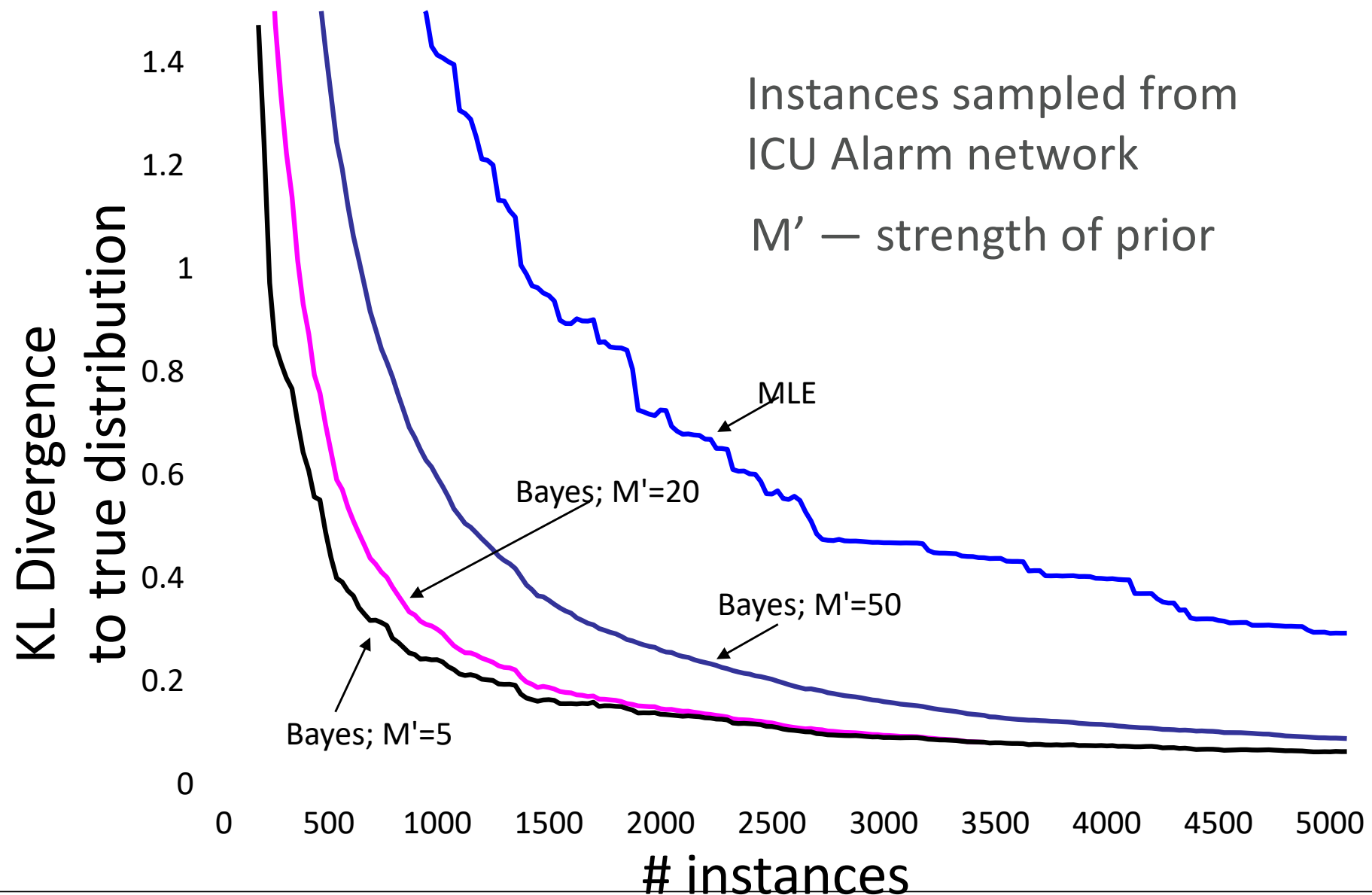
$$\tilde{\theta}_{x_i|pa_i} = \frac{\alpha(x_i, pa_i) + N(x_i, pa_i)}{\alpha(pa_i) + N(pa_i)}$$

Bayesian (Dirichlet)

- Both are asymptotically equivalent and consistent



# Learning Parameters: Case Study



## So far

- Where do the numbers come from?
  - Expectation-Maximization (EM)
  - Gradient
  - Bayesian learning



# What's next

- Structure Learning/Model Selection
  - Constrained-based
  - Score-based
- But first
  - I-equivalence, perfect maps etc.



# Recap : Building BNs from independence properties

- From d-separation we learned:
  - Start from local Markov assumptions, obtain all independence assumptions encoded by graph
  - For most  $P$ 's that factorize over  $G$ ,  $I(G) = I(P)$
  - All of this discussion was for a given  $G$  that is an I-map for  $P$





# Recap: I-Map to Factorization

G is I-map of P  P factorizes according to G

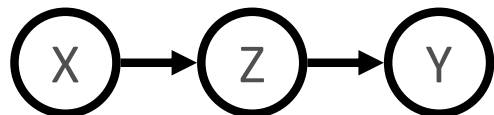
- Start with a topological ordering, wlog  $X_1, \dots, X_n$
- Apply chain rule
$$P(X_1, \dots, X_n) = P(X_1)P(X_2 | X_1) \dots P(X_n | X_1, \dots, X_{n-1})$$
- Consider  $P(X_i | X_1, \dots, X_{i-1})$
- We know that  $\text{Pa}(X_i) \subseteq \{X_1, \dots, X_{i-1}\}$ , i.e., there are no descendants of  $X_i$  in  $X_1, \dots, X_{i-1}$
- Hence, due to local Markov assumption

$$P(X_i | X_1, \dots, X_{i-1}) = P(X_i | \text{Pa}(X_i))$$

# Recap: D-Separation

Local Markov Assumption: A variable  $X$  is independent of its non-descendants given its parents and only its parents:  
 $(X_i \perp \text{NonDescendants}_{X_i} \mid \text{Pa}_{X_i})$

Indirect causal effect:



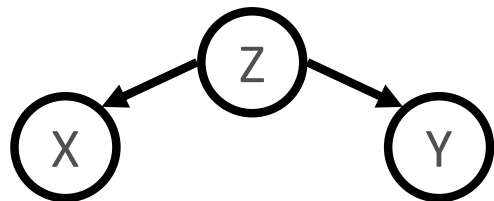
$$\begin{aligned} X &\perp Y \mid Z \\ \neg X &\perp Y \end{aligned}$$

Indirect evidential effect:



$$\begin{aligned} X &\perp Y \mid Z \\ \neg X &\perp Y \end{aligned}$$

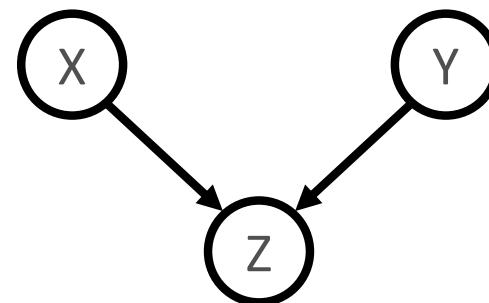
Common cause:



$$\begin{aligned} X &\perp Y \mid Z \\ \neg X &\perp Y \end{aligned}$$

Represent all the same distributions

(v-structure)  
Common effect:



*inverted*

$$\begin{aligned} X &\perp Y \\ \neg X &\perp Y \mid Z \end{aligned}$$

# Recap : Building BNs from independence properties

- From d-separation we learned:
  - Start from local Markov assumptions, obtain all independence assumptions encoded by graph
  - For most  $P$ 's that factorize over  $G$ ,  $I(G) = I(P)$
  - All of this discussion was for a given  $G$  that is an I-map for  $P$
- **Now, given a  $P$ , how can I get a  $G$ ?**
  - i.e., give me the independence assertions entailed by  $P$
  - However, many  $G$ s are “equivalent”.
  - How do we represent this?
  - **Most of this discussion is not about practical algorithms, but useful concepts that are used by practical algorithms**

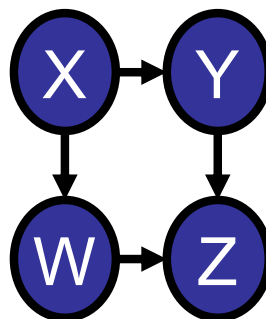
# Minimal I-maps

- Given the independence assertions that are true for  $P$ , find  $G$
- One option:
  - $G$  is an I-map for  $P$ ,  $I_l(G) \subseteq I(P)$
  - $G$  is as simple as possible
- *But what do we mean by “simple”?  $G$  is a minimal I-map for  $P$  if deleting any edges from  $G$  makes it no longer an I-map*
- Is this a good idea?

# Obtaining Minimal I-maps

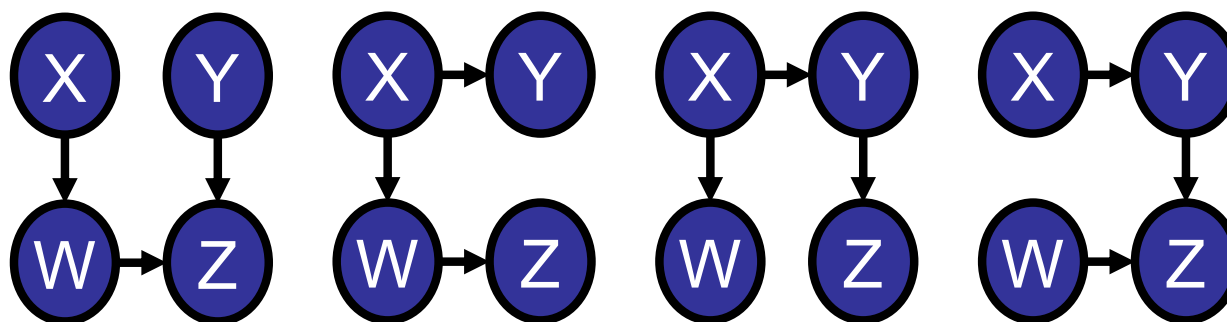
Example:

if



is an minima I-map

■ Then:

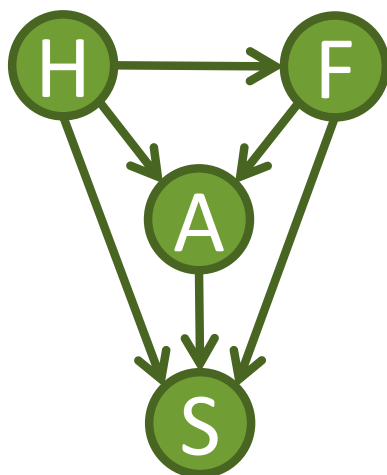


not minimal-maps

# Obtaining Minimal I-maps

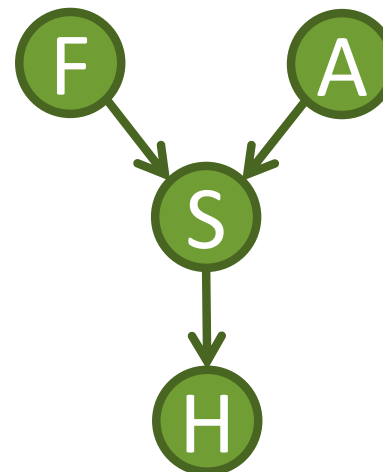
Flu, Allergy, SinusInfection, Headache

2 3 4 1



Flu, Allergy, SinusInfection, Headache

1 2 3 4



**Both are minimal I-maps (if we remove an edge, they are not I-maps anymore) but the left-hand side BN is much more complicated!**

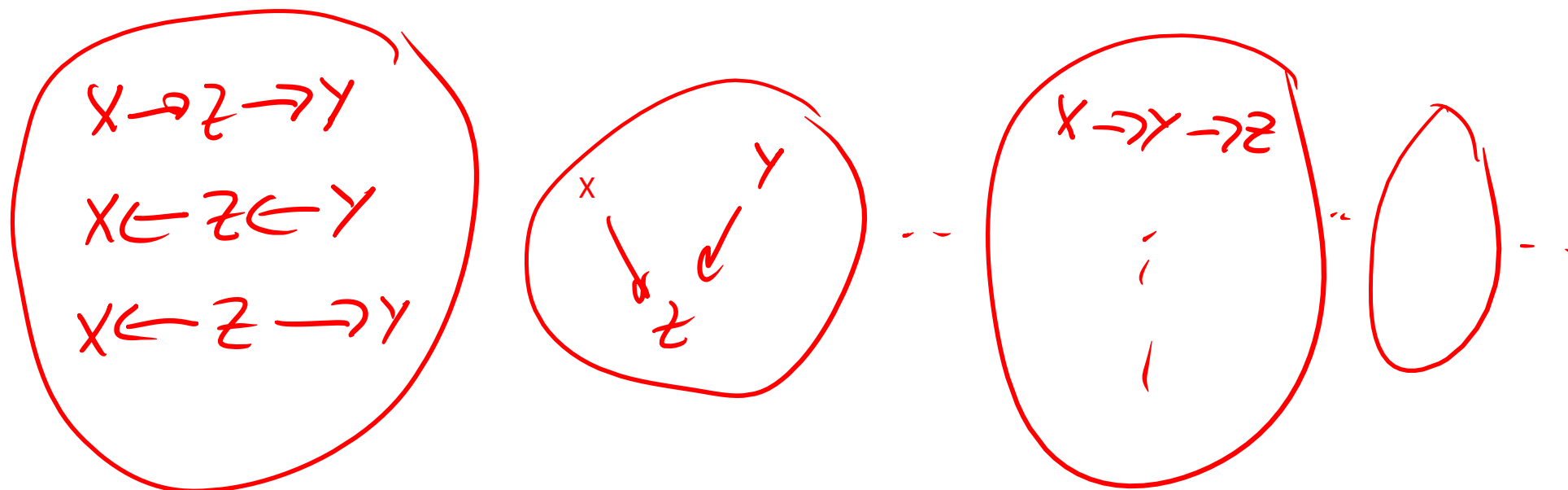
# Perfect maps (P-maps)

- I-maps are not unique and often not simple enough
- Define “simplest”  $G$  differently
  - A BN structure  $G$  is a perfect map for  $P$  if  $I(P) = I(G)$
- Our new goal is now:
  - Find a perfect map!
  - Must address **equivalent BNs**



# I-Equivalence

- Two graphs  $G_1$  and  $G_2$  are I-equivalent if  $I(G_1) = I(G_2)$
- However, p-maps are not unique
- Equivalence class of BN structures
  - Mutually-exclusive and exhaustive partition of graphs

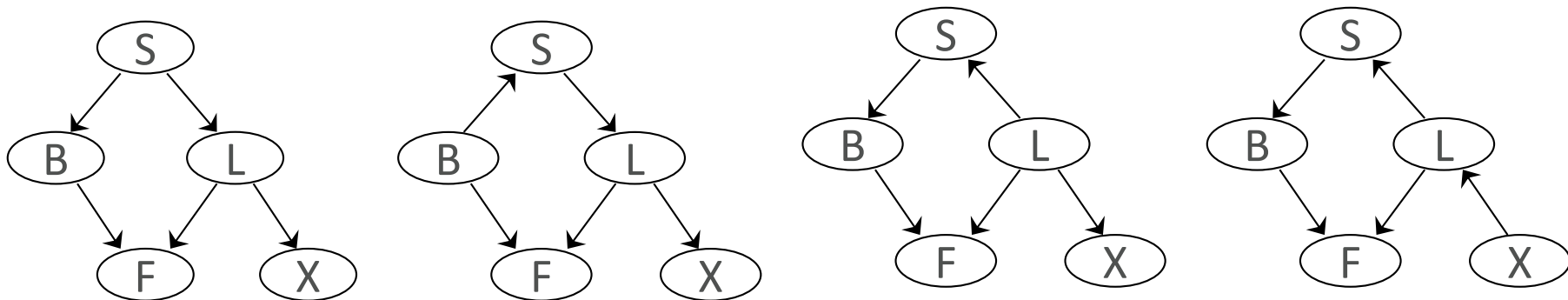


- How do we characterize these equivalence classes?

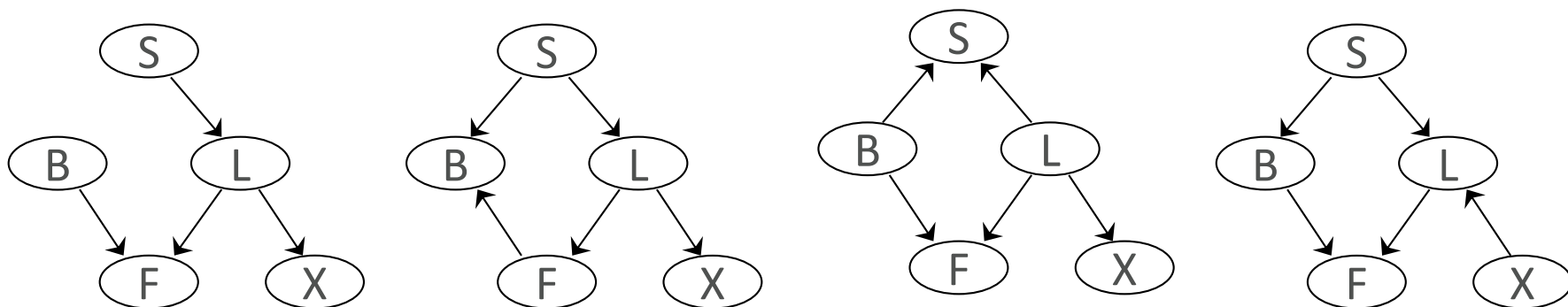


# Examples of I-Equivalence

The following four DAGs are Markov equivalent and are said to form a Markov equivalence class.

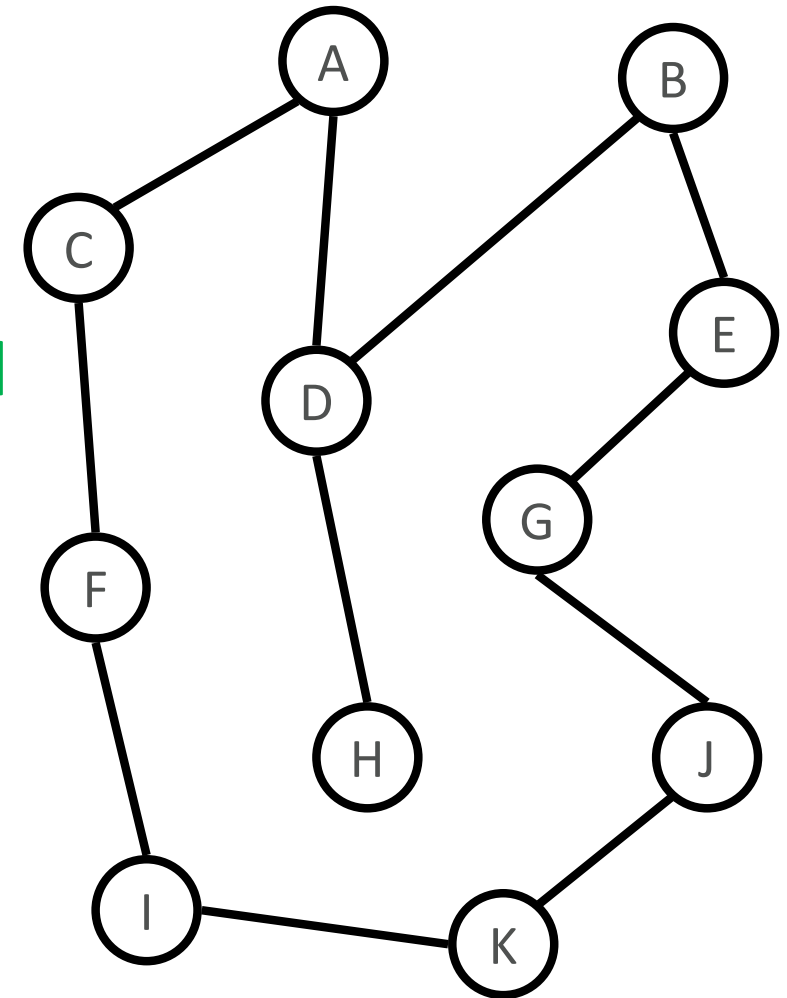


The following four DAGs are not Markov equivalent to those given above:



## Skeleton of a BN

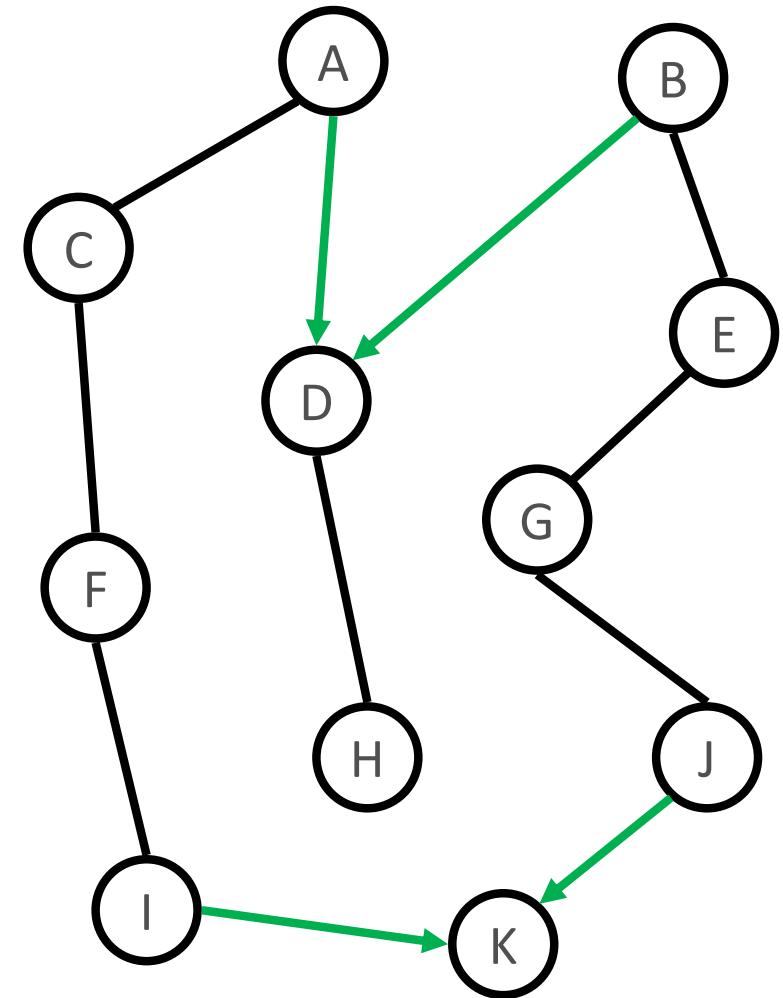
- Skeleton of a BN structure  $G$  is an undirected graph over the same variables that has an edge  $X-Y$  for every  $X \rightarrow Y$  or  $Y \rightarrow X$  in  $G$  (as we used it already for trails)
- (Little) Lemma: Two I-equivalent BN structures must have the same skeleton
- Proof via trials



What about the other „direction“?

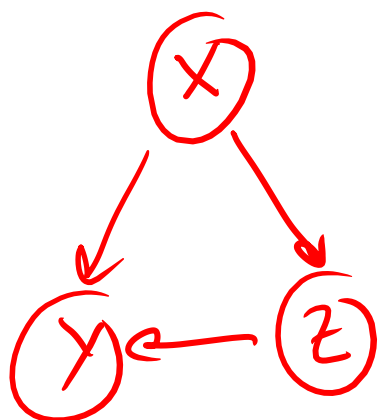
# What about V-structures?

- V-structures are key property of BN structure (as used for d-separation)
- **Theorem:** If  $G_1$  and  $G_2$  have the same skeleton and V-structures, then  $G_1$  and  $G_2$  are I-equivalent
- Proof via trials and rules of d-separation

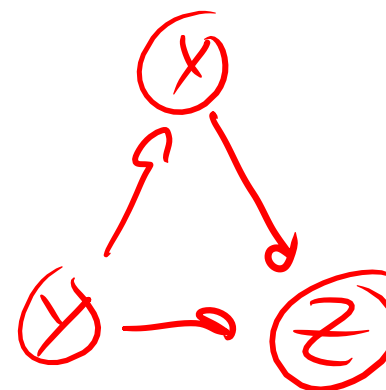


# Same V-structures not necessary

- **Theorem:** If  $G_1$  and  $G_2$  have the same skeleton and V-structures, then  $G_1$  and  $G_2$  are I-equivalent
- Though sufficient, same V-structures not necessary, i.e., no **iff**



diff. V-structures  
but  
I-equiv.



Actually, for the “none-necessity”, any complete graph can be considered. They are I-equivalent but they do not have the same V-structures



# Immoralities & I-Equivalence

- Key concept is not V-structure, but “**immoralities**” (unmarried parents 😊)
  - **$X \rightarrow Z \leftarrow Y$ , with no arrow between X and Y**
- Important pattern: **X and Y independent given their parents, but not given Z**
- (If edge exists between X and Y (moralized), we have *covered* the V-structure)
- **Theorem:**  
 **$G_1$  and  $G_2$  have the same skeleton and immoralities if and only if  $G_1$  and  $G_2$  are I-equivalent**

# Obtaining a P-map

- **Step 1: Given the independence assertions that are true for  $P$** 
  - Obtain skeleton
  - Obtain immoralities
- We have found the right equivalence class
- **Step 2: From skeleton and immoralities, obtain every (and any) BN structure from the equivalence class**

# Identifying the skeleton 1

- When is there an edge between  $X$  and  $Y$ ?
  - Difficult to answer
- When is there no edge between  $X$  and  $Y$ ?
  - Local Markov Assumption
  - Exists  $\mathbf{U} \subseteq \mathbf{X} - \{X, Y\}$ ,  $|\mathbf{U}| \leq d$ , such that  $(X \perp Y \mid \mathbf{U})$
  - $d$  maximal number of parents

# Identifying the skeleton 2

- Assume  $d \leq (n-1)$  is max number of parents
- For each  $X_i$  and  $X_j$ 
  - $E_{ij} \leftarrow \text{true}$
  - For each  $\mathbf{U} \subseteq \mathbf{X} - \{X_i, X_j\}$ ,  $|\mathbf{U}| \leq d$ 
    - Is  $(X_i \perp X_j \mid \mathbf{U})$  ? // note that we “assume “ to know this
      - $E_{ij} \leftarrow \text{false}$
      - break
  - If  $E_{ij}$  is true
    - Add edge  $X - Y$  to skeleton
- For measuring independence, there are several ways. One considers the mutual information (you will see this later)



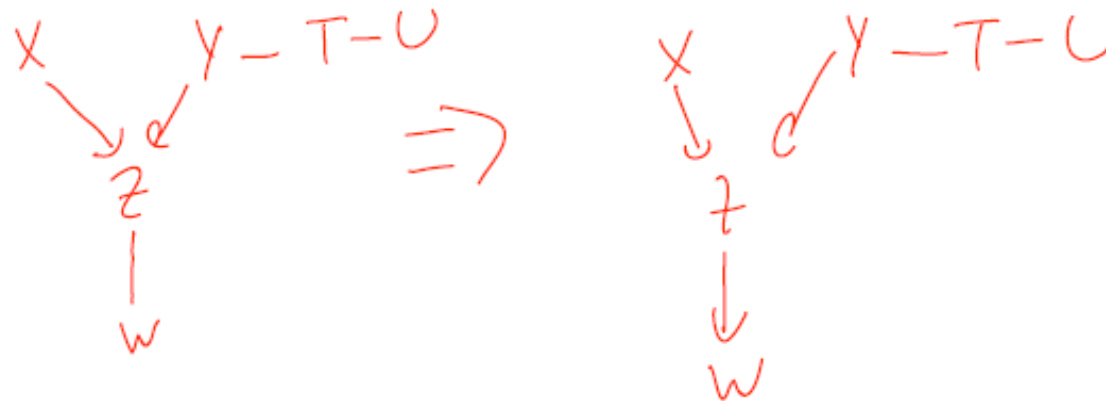


# Identifying immoralities

- Consider  $X - Z - Y$  in skeleton, when should it be an immorality  $X \rightarrow Z \leftarrow Y$ ? Essentially when  $X \perp Y$
- **Must be  $X \rightarrow Z \leftarrow Y$  (immorality):**
  - When  $X$  and  $Y$  are **never independent given  $U$** , if  $Z$  in  $U$ 
    - Proof: Assume no  $Z$  exists but  $X - Z - Y$  is not an immorality
    - Then, either  $X \rightarrow Z \rightarrow Y$  or  $X \leftarrow Z \leftarrow Y$  or  $X \leftarrow Z \rightarrow Y$  exists
    - But then, we can block  $X - Z - Y$  by  $Z$
    - Then, since  $X$  and  $Y$  are not connected, can find  $U$  that includes  $Z$  such that  $\text{Ind}(X; Y \mid \mathbf{W})$
    - **Contradiction**
- **Must not be  $X \rightarrow Z \leftarrow Y$  (not immorality):**
  - When there exists  $U$  with  $Z$  in  $U$ , such that  $X$  and  $Y$  are **independent given  $U$**

# From immoralities and skeleton to BN structures

- Representing BN equivalence class as a **partially-directed acyclic graph (PDAG)**
- Immoralities force direction on other BN edges



Otherwise we would get another immoral v-structure, which was ruled out by the algorithm




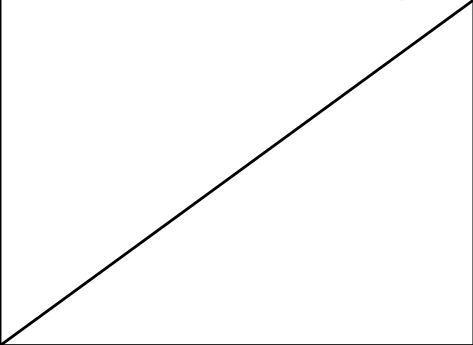
- Full (polynomial-time) procedure exists





# What you need to know

- Minimal I-map
  - every  $P$  has one, but usually many
- Perfect map
  - better choice for BN structure
  - not every  $P$  has one
  - can find one (if it exists) by considering I-equivalence
  - Two structures are I-equivalent if they have same skeleton and immoralities

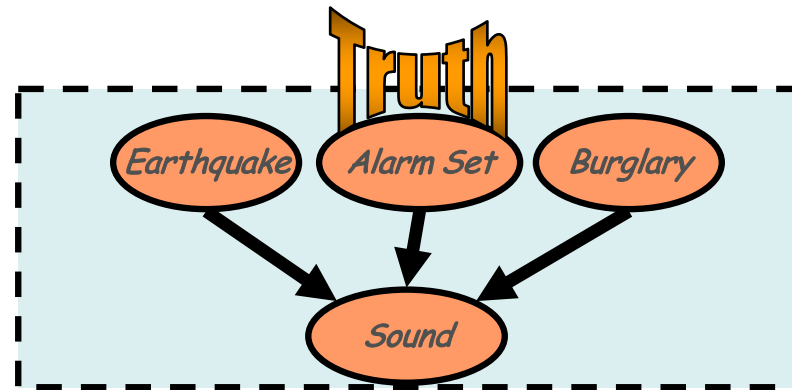
# Learning With Bayesian Networks

		Fixed structure 	Fixed variables 	Hidden variables 
observed	fully	Easiest problem counting	Selection of arcs New domain with no domain expert Data mining	
	Partially	Numerical, nonlinear optimization, Multiple calls to BNs, Difficult for large networks	Encompasses to difficult subproblem, „Only“ Structural EM is known	

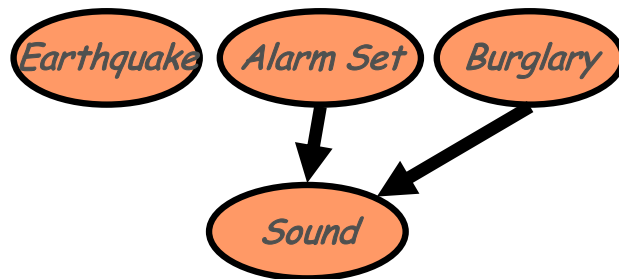



Parameter Estimation
Structure learning

# Why Struggle for Accurate Structure?

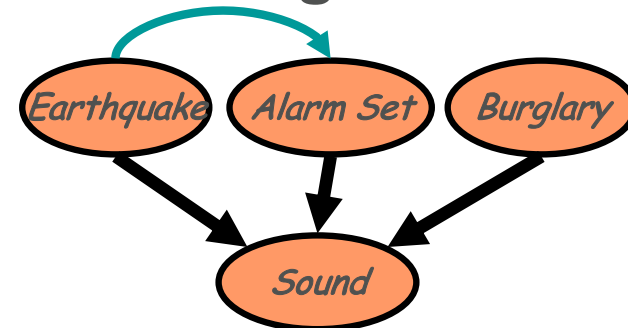


## Missing an arc



- Cannot be compensated for by fitting parameters
- Wrong assumptions about domain structure

## Adding an arc

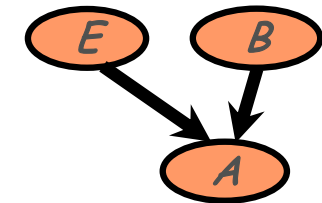


- Increases the number of parameters to be estimated
- Wrong assumptions about domain structure

# Unknown Structure, (In)complete Data

E, B, A  
 <Y,N,N>  
 <Y,N,Y>  
 <N,N,Y>  
 <N,Y,Y>  
 .  
 .  
 <N,Y,Y>

- Network structure is not specified
  - Learner needs to select arcs & estimate parameters
- Data does not contain missing values



$E$	$B$	$P(A   E, B)$	
$e$	$b$	?	?
$e$	$\bar{b}$	?	?
$\bar{e}$	$b$	?	?
$\bar{e}$	$b$	?	?



Learning  
algorithm



$E$	$B$	$P(A   E, B)$	
$e$	$b$	.9	.1
$e$	$\bar{b}$	.7	.3
$\bar{e}$	$b$	.8	.2
$\bar{e}$	$\bar{b}$	.99	.01

- Network structure is not specified
- Data contains missing values
  - Need to consider assignments to missing values

E, B, A  
 <Y,?,N>  
 <Y,N,?>  
 <N,N,Y>  
 <N,Y,Y>  
 .  
 .  
 <?,Y,Y>



# Structure Learning

- Two main approaches
- 1. Constrained-based
- 2. Score-based

# Constraint-Based Learning

- Remember: Obtaining a P-Map?
  - Given the independence assertions that are true for  $P$ 
    - Obtain skeleton and then obtain immoralities
  - From skeleton and immoralities, obtain every (and any) BN structure from the equivalence class
- Only now, we do **not** have  $I(X,Y|Z)$  statements
- **Basic task:**
  - Determine whether two variables are independent
  - Well studied question in statistics





# Testing Independence

- Null hypothesis  $H_0$  is
  - Data was sampled from  $P(X,Y)=P(X)*P(Y)$
- Need a procedure that will Accept or Reject  $H_0$
- Use  $\chi^2$ -test
  - $\chi^2(X,Y) \sim I(X,Y) * N * \ln(4)$  where  $I(X,Y)$  is the mutual information
- Or directly mutual information

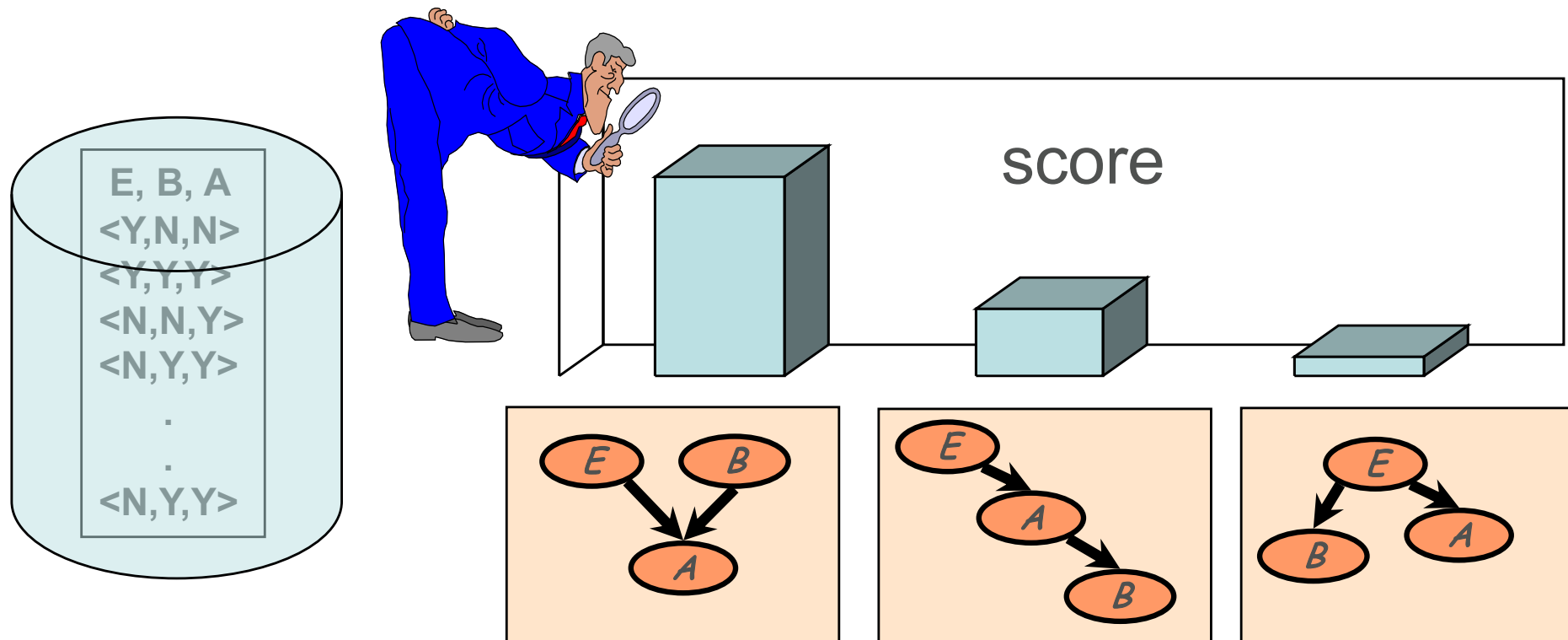
$$\hat{P}(x_i, x_j) = \frac{\text{Count}(x_i, x_j)}{m}$$

$$\hat{I}(X_i, X_j) = \sum_{x_i, x_j} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i) \hat{P}(x_j)}$$



# Score-based Learning

Define scoring function that evaluates how well a structure matches the data



Search for a structure that maximizes the score

# Likelihood Score for Structure

$$\ell(G : D) = \log L(G : D) = M \sum_i \left( I(X_i; Pa_i^G) - H(X_i) \right)$$

Mutual information between  
 $X_i$  and its parents

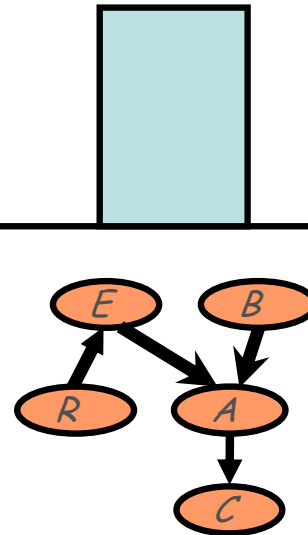
Entropy  $X_i$

- Larger dependence of  $X_i$  on  $Pa_i \Rightarrow$  higher score
- Adding arcs always helps
  - $I(X; Y) \leq I(X; \{Y, Z\})$
  - Max score attained by fully connected network
  - Overfitting: A bad idea...

# Bayesian Score

$P(G|D)$

Picking a single best model can be misleading

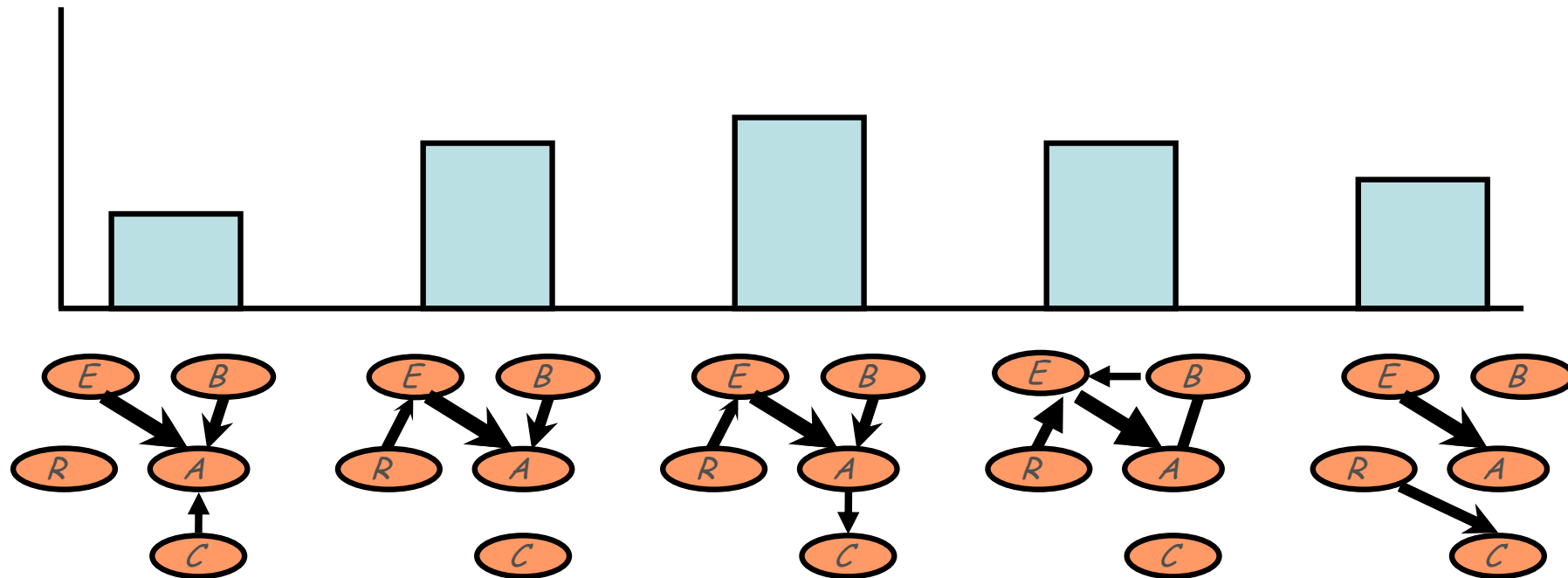


- Current practice: model selection
  - Pick a single high-scoring model
  - Use that model to infer domain structure



# Bayesian Score

$P(G|D)$



## Problem

- Small sample size  $\Rightarrow$  many high scoring models
- Answer based on one model often useless

# Bayesian Score

Likelihood score:  $L(G : D) = P(D | G, \hat{\theta}_G)$

Max likelihood params

**Bayesian approach:**

- Deal with uncertainty by assigning probability to all possibilities

$$P(D | G) = \int P(D | G, \theta) P(\theta | G) d\theta$$

Marginal Likelihood

Likelihood

Prior over parameters

$$P(G | D) = \frac{P(D | G) P(G)}{P(D)}$$

# Bayesian Score

- Bayesian has difficult integrals
- For Dirichlet prior, can use simple Bayes information criterion (BIC) approximation
- **Theorem:** for Dirichlet prior, and a BN with  $\text{Dim}(G)$  independent parameters, *as*  $m \rightarrow \infty$ :

$$\log P(D \mid G) = \log(D \mid G, \theta) - \frac{\log M}{2} \dim(G) + O(1)$$

# Bayesian Information Criterion (BIC)

$$\begin{aligned}\log P(D | G) &= \ell(G : D) - \frac{\log M}{2} \dim(G) + O(1) \\ &= \underbrace{M \sum_i \left( I(X_i; Pa_i^G) - H(X_i) \right)}_{\text{Fit dependencies in empirical distribution}} - \underbrace{\frac{\log M}{2} \dim(G)}_{\text{Complexity penalty}} + O(1)\end{aligned}$$

- As  $M$  (amount of data) grows,
  - Increasing pressure to fit dependencies in distribution
  - Complexity term avoids fitting noise
- Asymptotic equivalence to MDL score
- Bayesian score/BIC is **consistent**
  - Observed data eventually overrides prior





# Structure Search as Optimization

## Input:

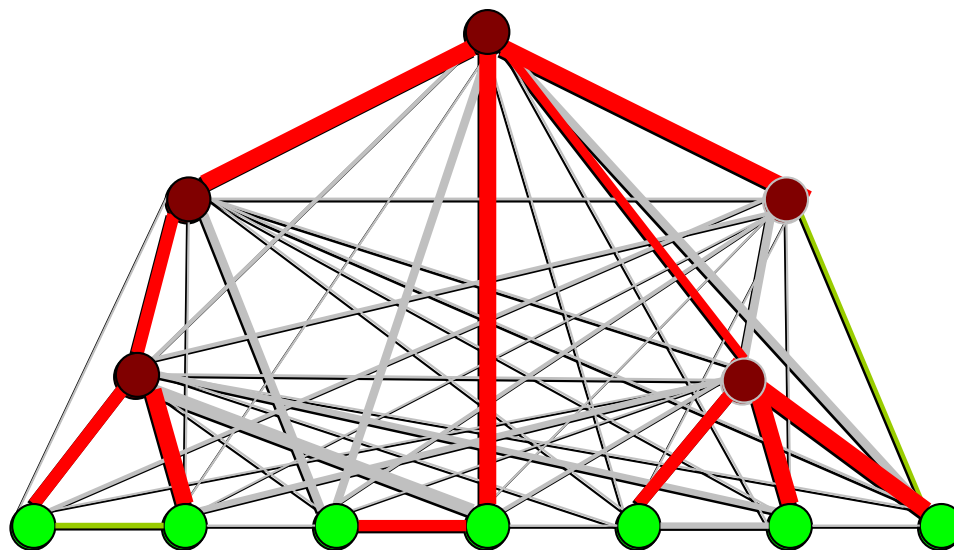
- Training data
- Scoring function (MDL, BIC, ...)
- Set of possible structures

## Output:

- A network that maximizes the score



# Learning Trees (complete data)



- Can find optimal tree structure in  $O(n^2 \log n)$  time: just find the max-weight spanning tree
- If some of the variables are hidden, problem becomes hard again, but can use EM to fit mixtures of trees

# Heuristic Search

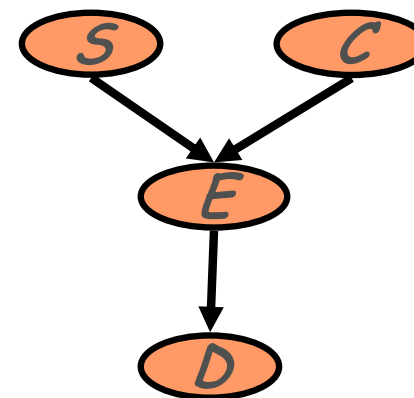
- Define a search space:
  - search states are possible structures
  - operators make small changes to structure
- Traverse space looking for high-scoring structures
- Search techniques:
  - Greedy hill-climbing
  - Best first search
  - Simulated Annealing
  - ...

**Theorem:** Finding maximal scoring structure with at most  $k$  parents per node is NP-hard for  $k > 1$



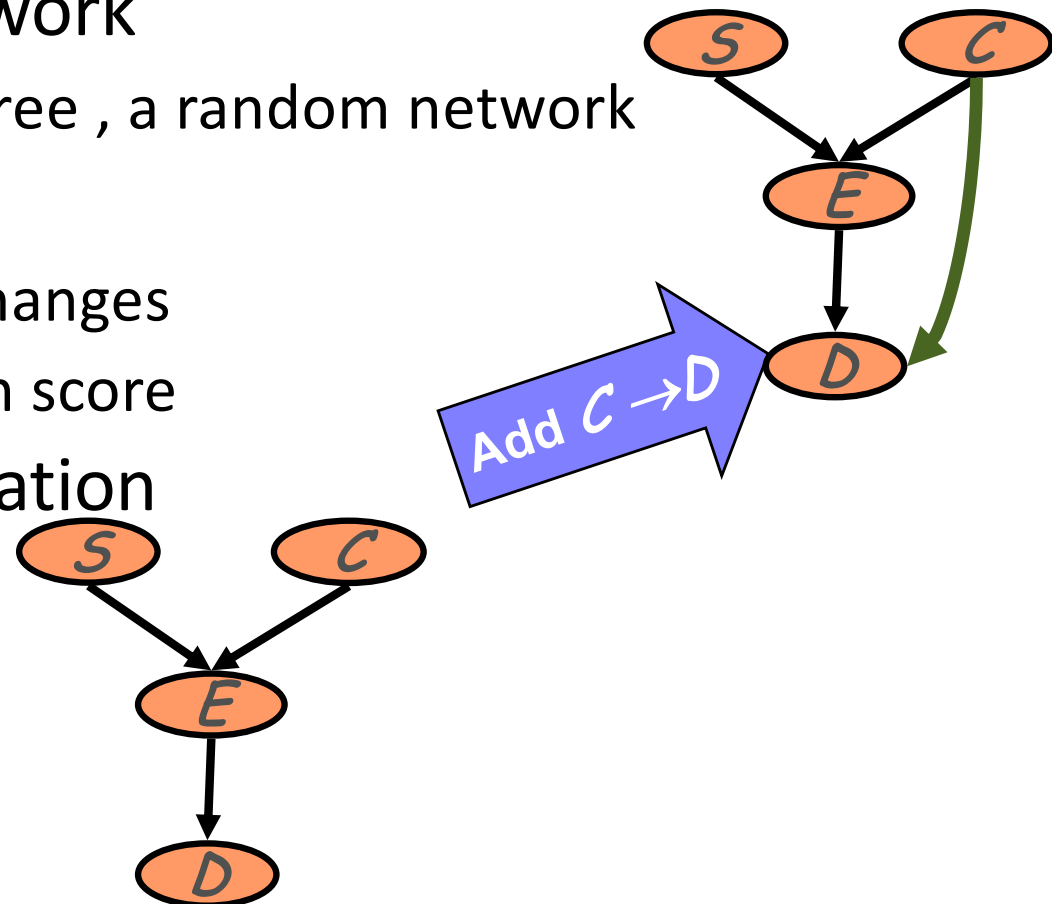
# Typically: Local Search

- Start with a given network
  - empty network, best tree , a random network
- At each iteration
  - Evaluate all possible changes
  - Apply change based on score
- Stop when no modification improves score



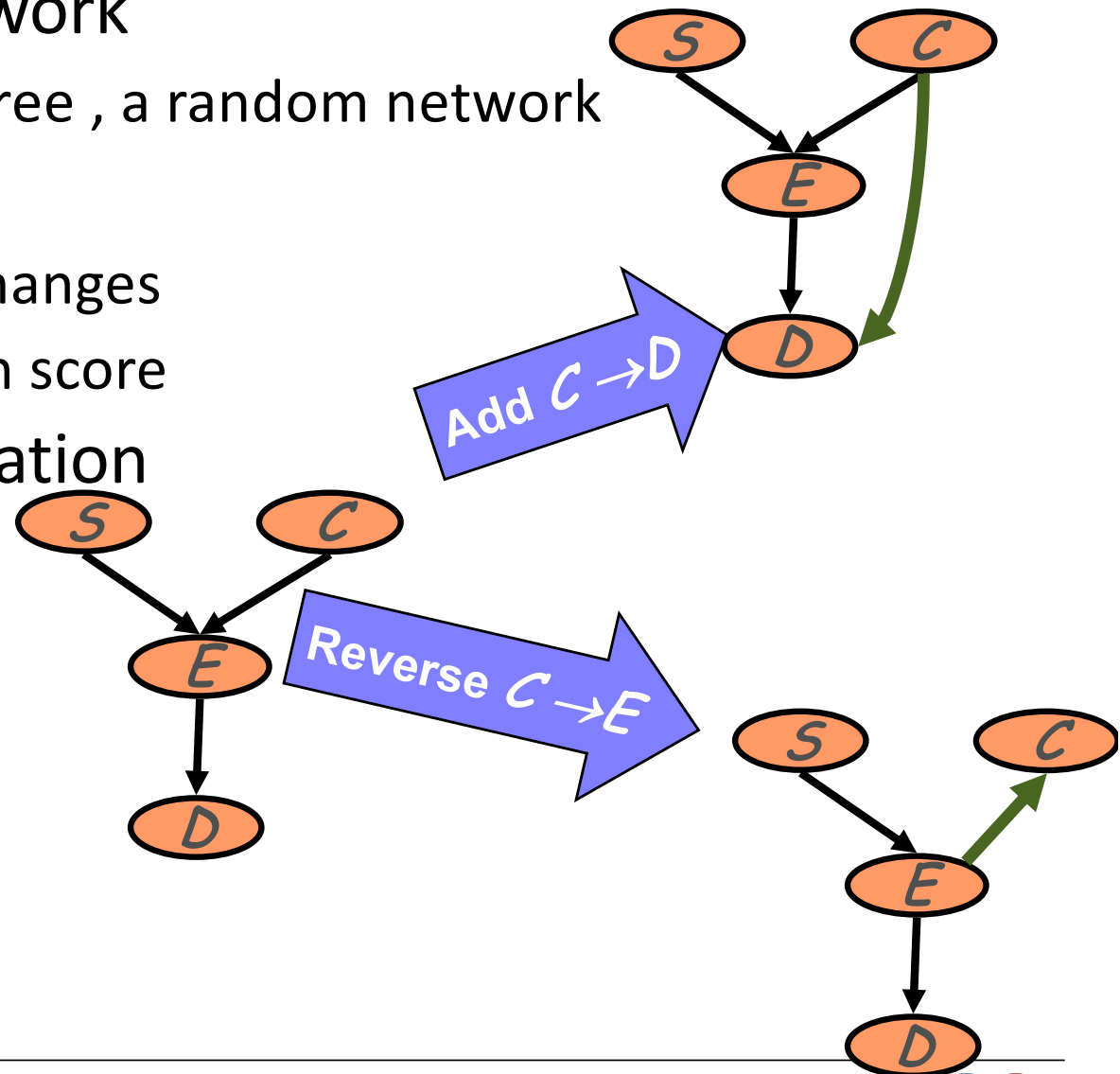
# Typically: Local Search

- Start with a given network
  - empty network, best tree, a random network
- At each iteration
  - Evaluate all possible changes
  - Apply change based on score
- Stop when no modification improves score



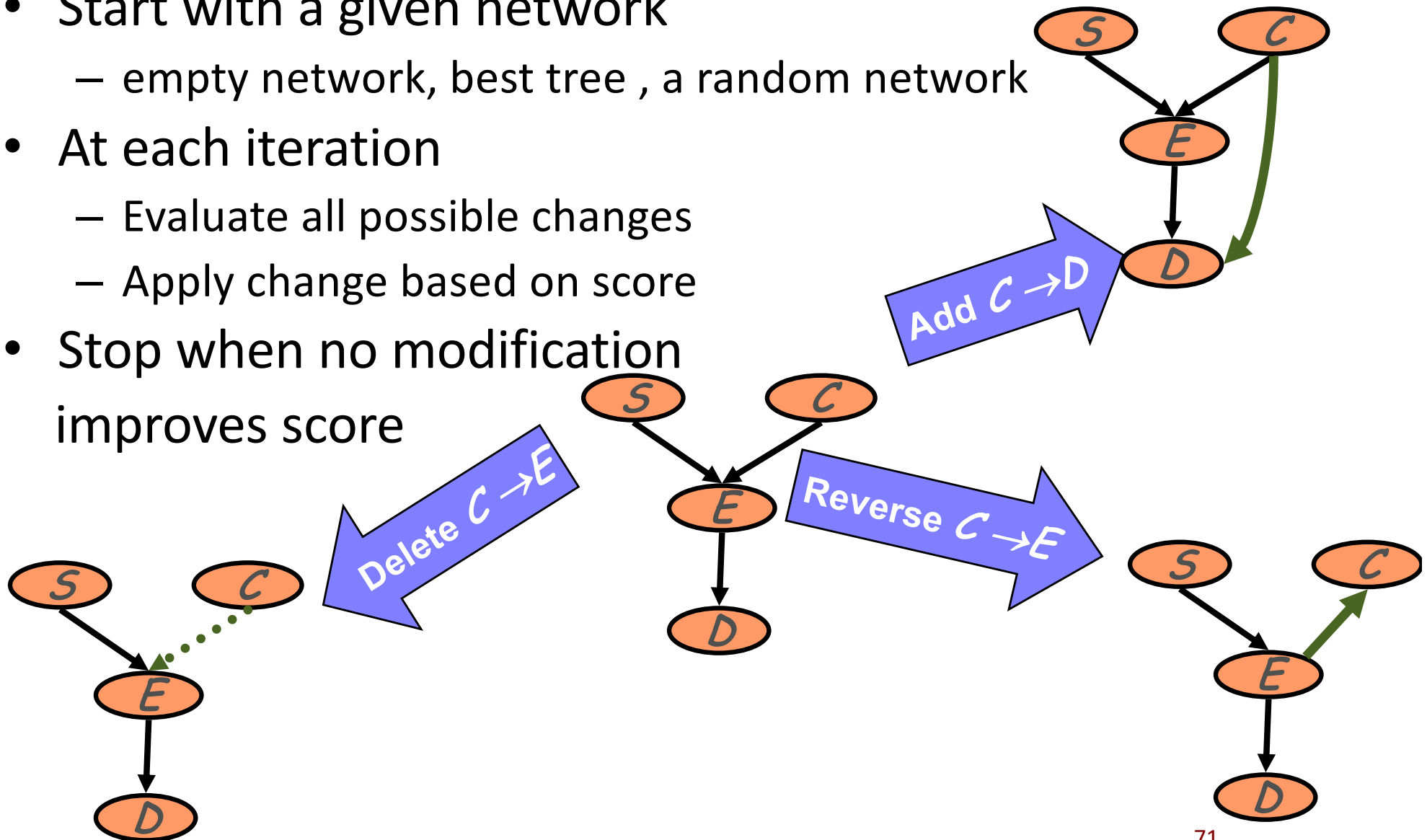
# Typically: Local Search

- Start with a given network
  - empty network, best tree, a random network
- At each iteration
  - Evaluate all possible changes
  - Apply change based on score
- Stop when no modification improves score



# Typically: Local Search

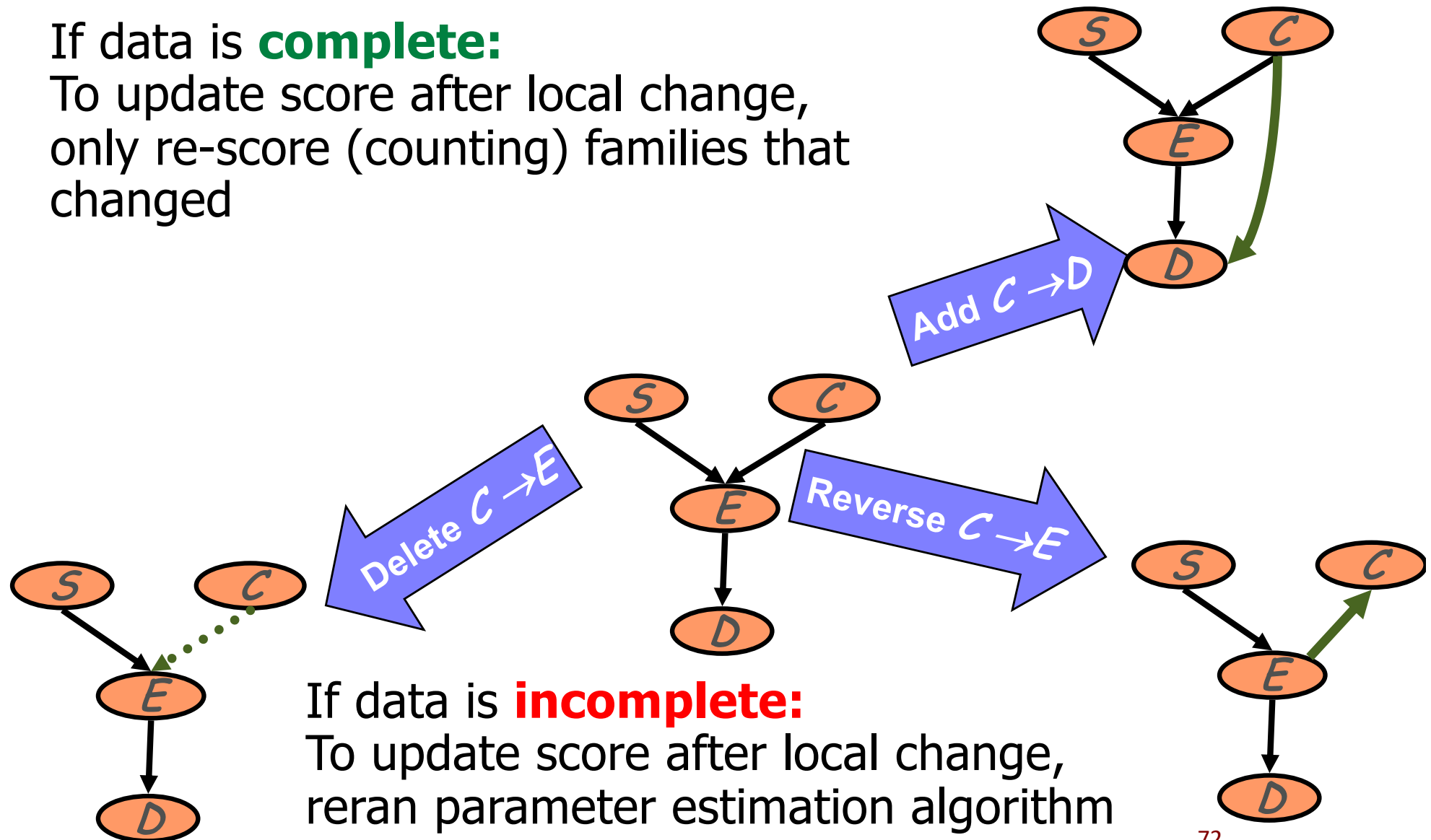
- Start with a given network
  - empty network, best tree, a random network
- At each iteration
  - Evaluate all possible changes
  - Apply change based on score
- Stop when no modification improves score



# Typically: Local Search

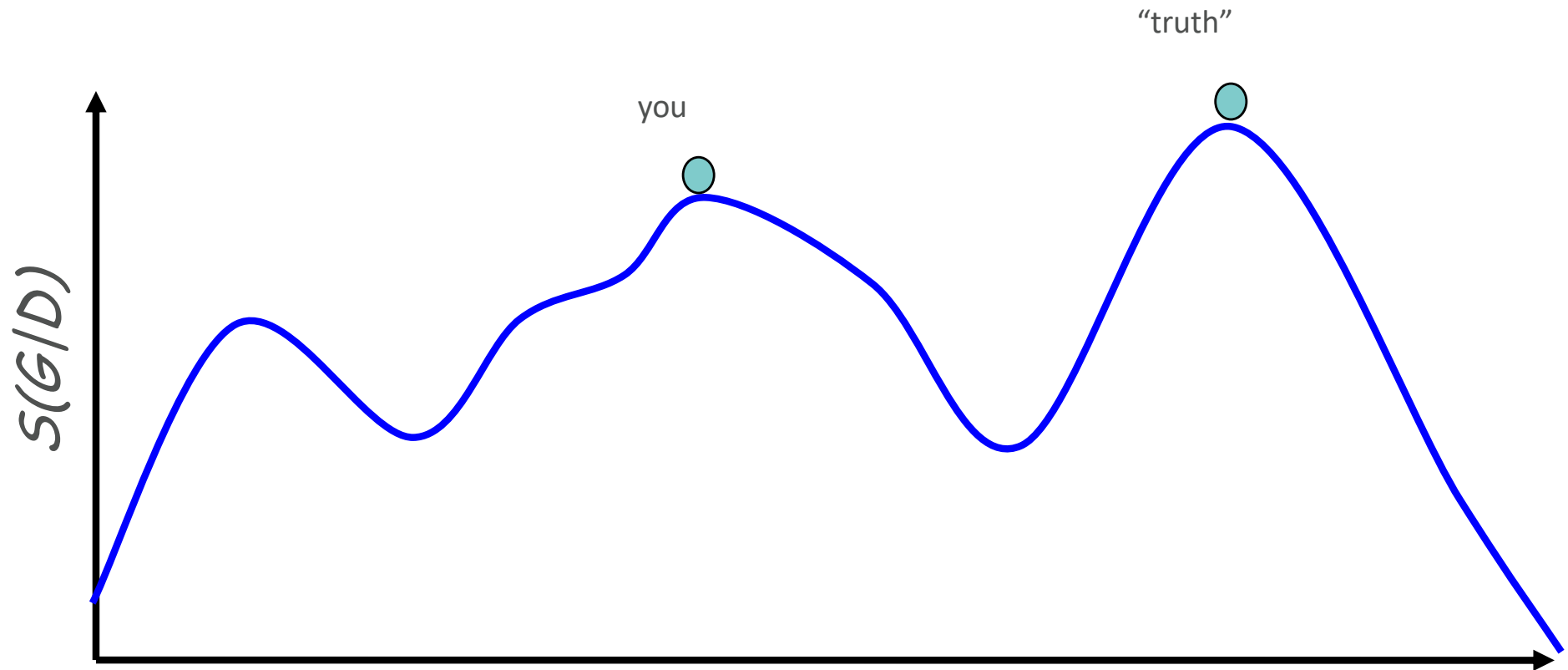
If data is **complete**:

To update score after local change,  
only re-score (counting) families that  
changed





# Other Problem with local search



Easy to get stuck in local optima

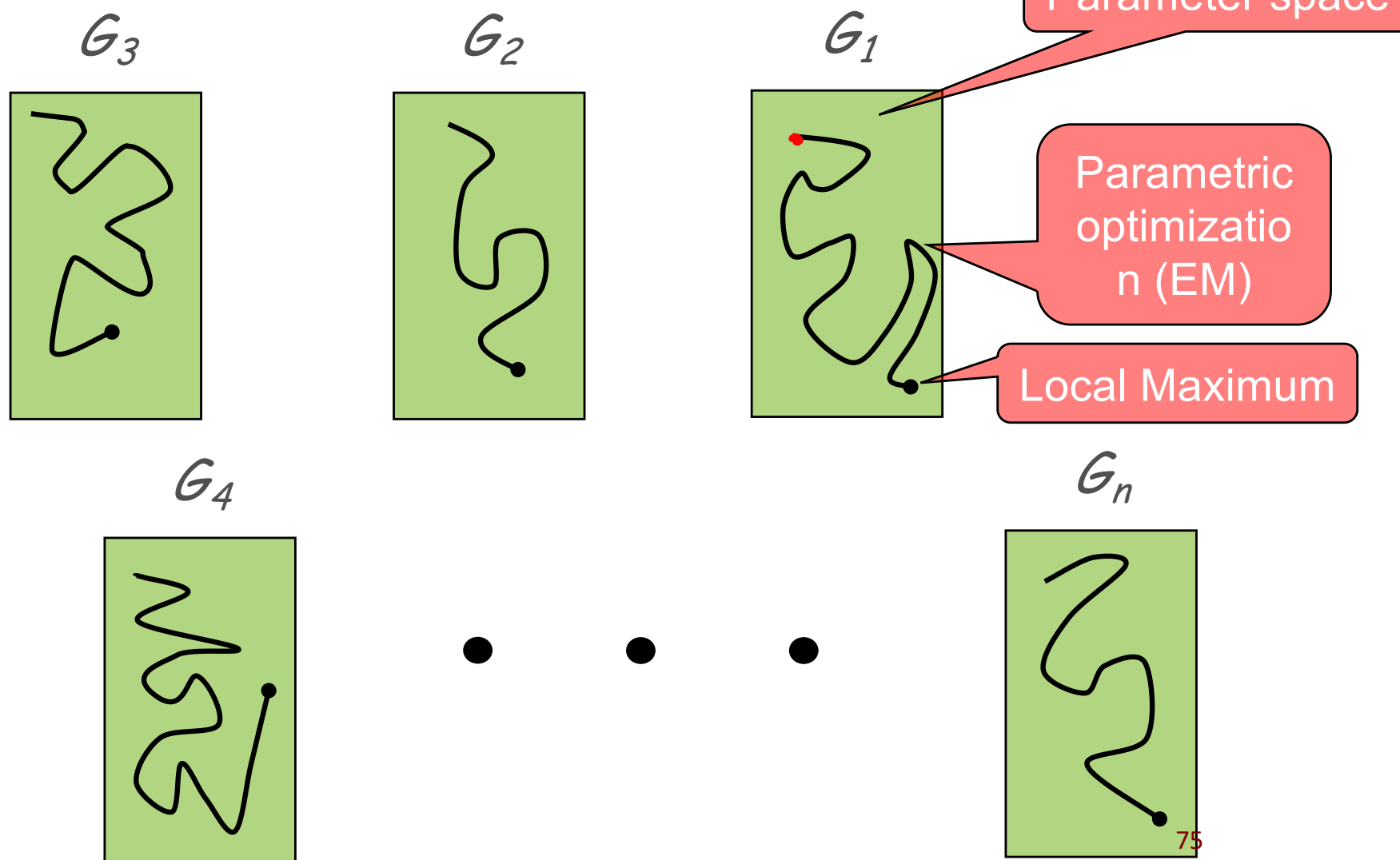
# Local Search in Practice

- Local search can get stuck in:
  - **Local Maxima:**
    - All one-edge changes reduce the score
  - **Plateaux:**
    - Some one-edge changes leave the score unchanged
- So, standard heuristics can escape both
  - Random restarts
  - TABU search
  - Simulated annealing



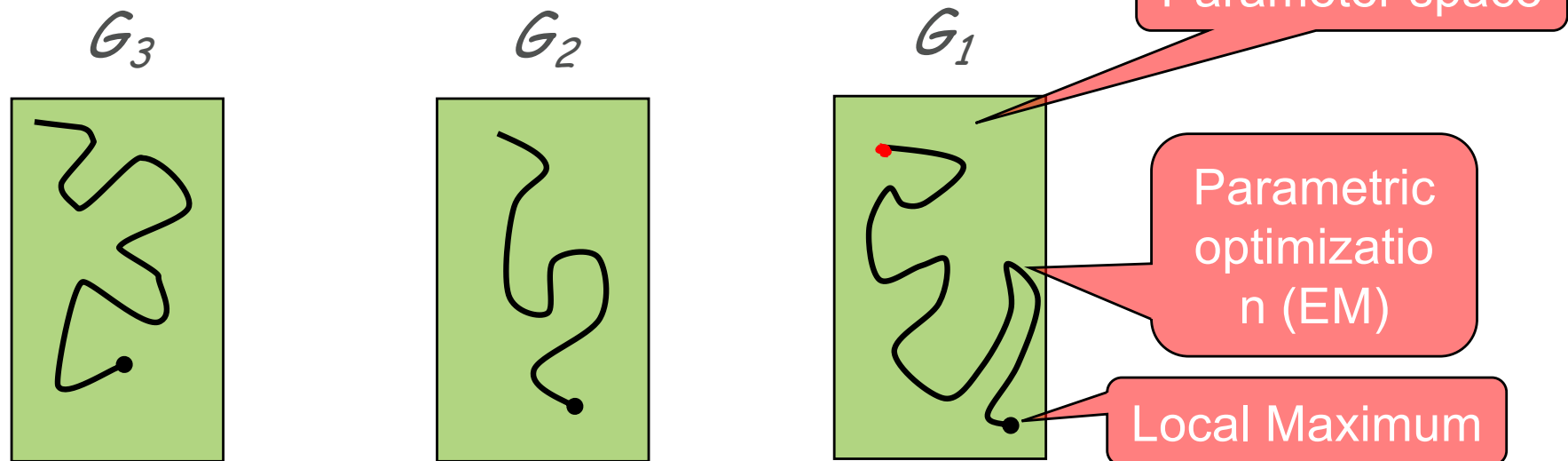
# Local Search in Practice

- Perform EM for each candidate graph



# Local Search in Practice

- Perform EM for each candidate graph



- ◆ Computationally expensive:
    - Parameter optimization via EM — non-trivial
    - Need to perform EM for all candidate structures
    - Spend time even on poor candidates
- ⇒ In practice, considers only a few candidates

# Structural EM

[Friedman et al. 98]

Recall, in complete data we had

- Decomposition  $\Rightarrow$  efficient search

**Idea:**

- Instead of optimizing the real score...
- Find **decomposable** alternative score
- Such that maximizing new score  
 $\Rightarrow$  improvement in real score



# Structural EM

[Friedman et al. 98]

## Idea:

- Use current model to help evaluate new structures

## Outline:

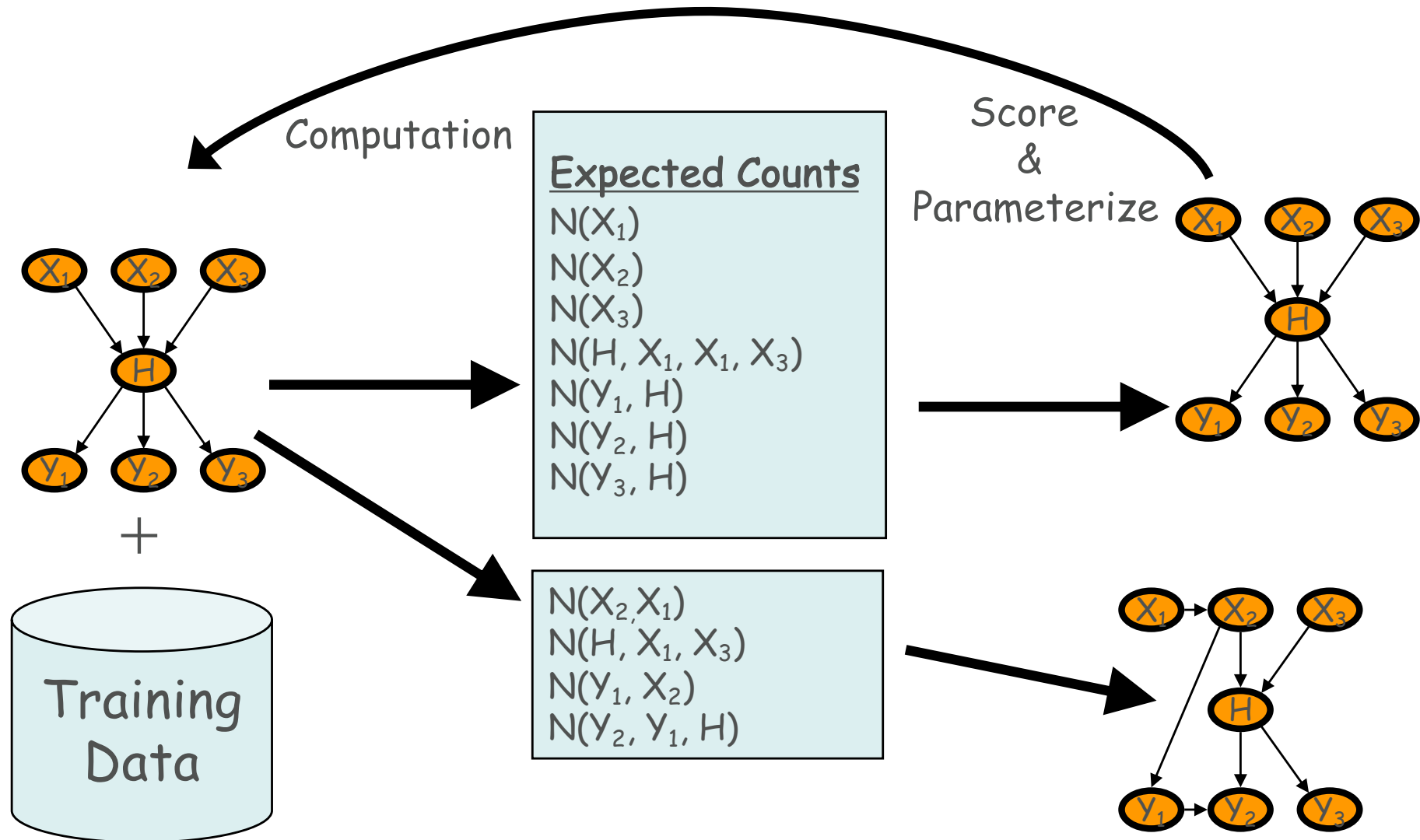
- Perform search in (Structure, Parameters) space
- At each iteration, use current model for finding
  - Better scoring parameters: “parametric” EM stepor
  - Better scoring structure: “structural” EM step



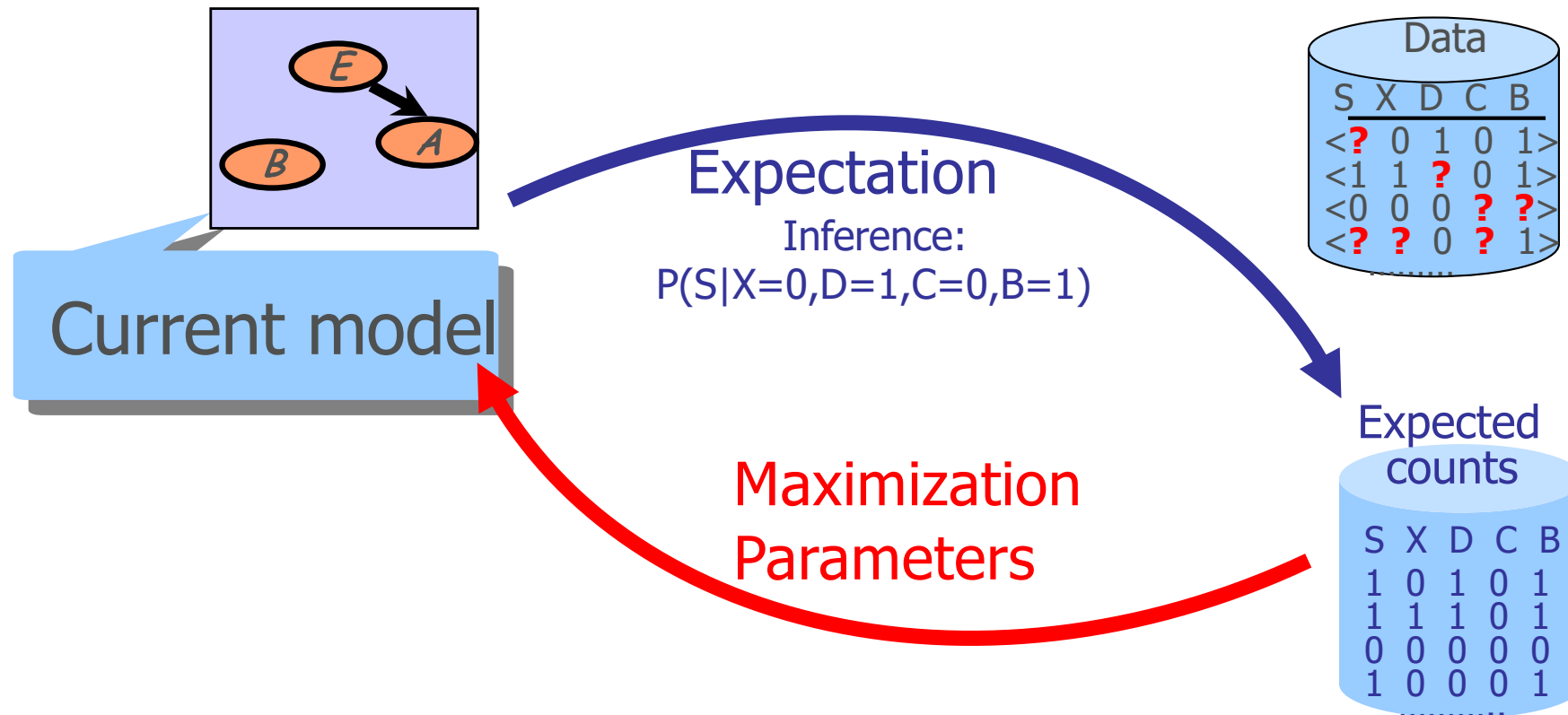
# Structural EM

[Friedman et al. 98]

Reiterate



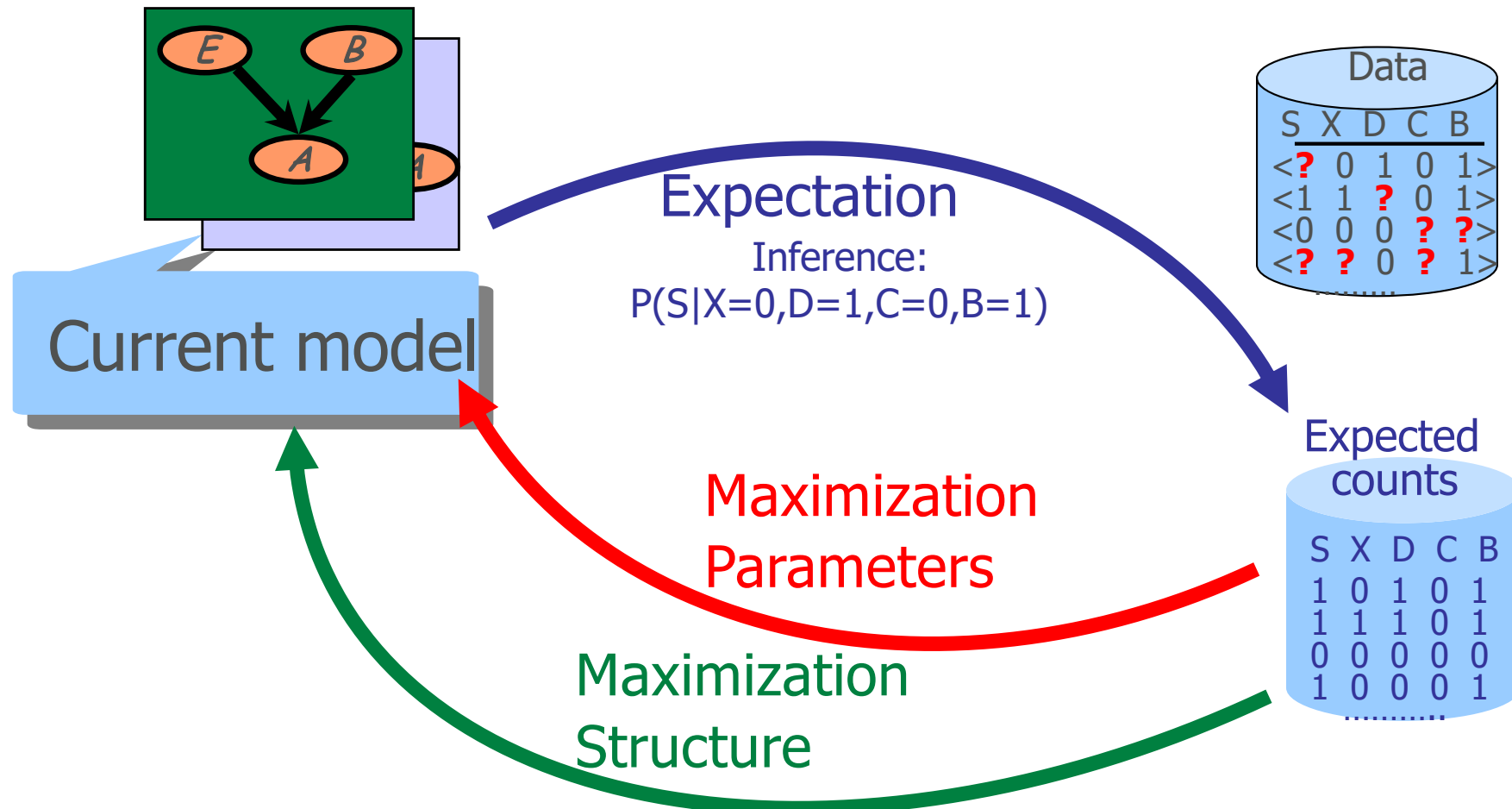
# Structure Learning: incomplete data



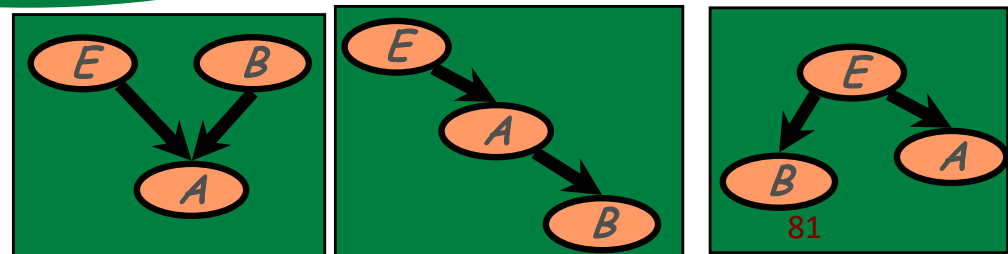
**EM**-algorithm:  
iterate until convergence



# Structure Learning: incomplete data



**SEM**-algorithm:  
iterate until convergence



# Structure Learning: Summary

- Expert **knowledge** + learning from **data**
- Structure learning involves parameter estimation (e.g. EM)
- Optimization w/ **score** functions
  - **likelihood** + complexity penalty = MDL
- **Local traversing** of space of possible structures:
  - add, reverse, delete (single) arcs
- Speed-up: **Structural EM**
  - **Score** candidates **w.r.t. current best** model

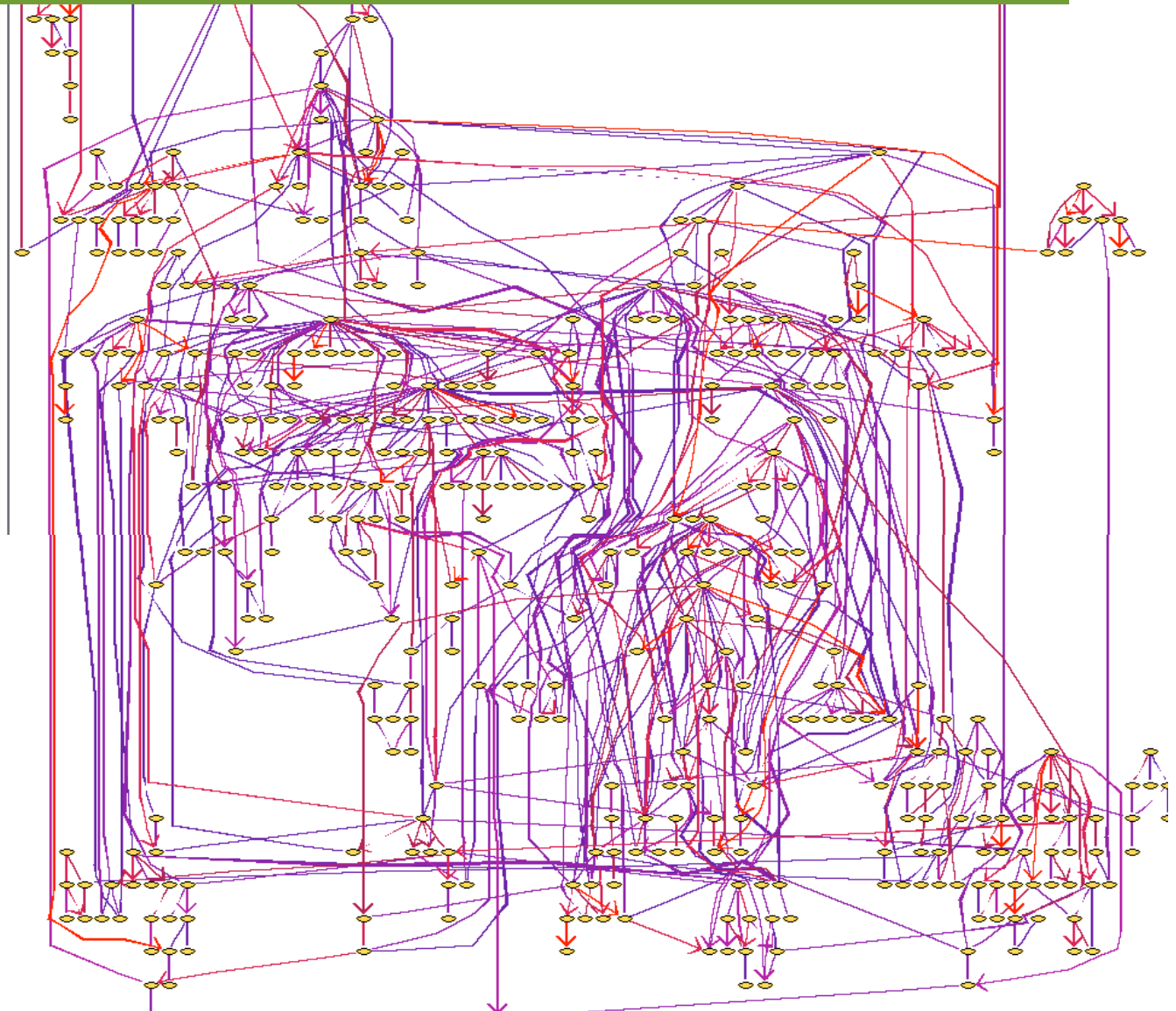


# Structure learning success stories: gene regulation network (Friedman et al.)

## Yeast data

[Hughes et al 2000]

- 600 genes
- 300 experiments



# Structure learning success stories II: Phylogenetic Tree Reconstruction (Friedman et al.)

**Input:** Biological sequences

Human CGT**T**GC...

Chimp CCT**A**GG...

Orang CGA**A**CG...

....

Uses structural EM,  
with max-spanning-tree  
in the inner loop

**Output:** a phylogeny

