

# Advanced Algorithms and Models for Computational Biology -- a machine learning approach

**Systems Biology:**  
Inferring gene regulatory network using  
graphical models

Eric Xing

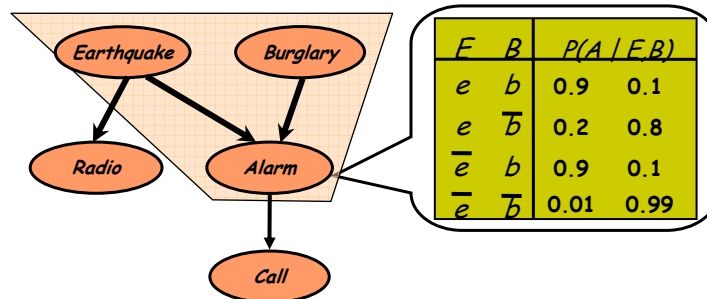
Lecture 25, April 19, 2006



## Bayesian Network – CPDs

Local Probabilities: **CPD** - conditional probability  
distribution  $P(X_i/Pa_i)$

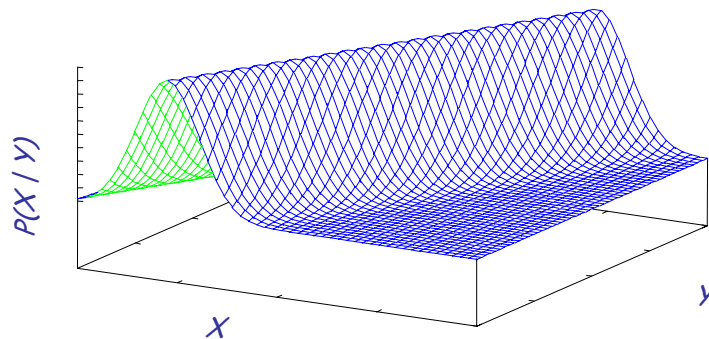
- Discrete variables: Multinomial Distribution (can represent **any** kind of statistical dependency)



## Bayesian Network – CPDs (cont.)

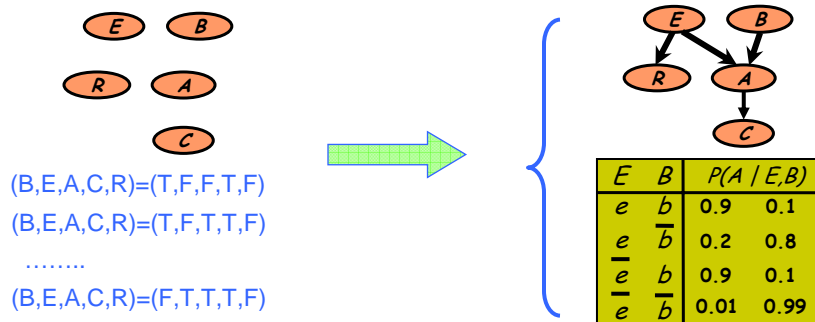
- Continuous variables: e.g. linear Gaussian

$$P(X|Y_1, \dots, Y_k) \sim N(a_0 + \sum_{i=1}^k a_i y_i, \sigma^2)$$



## Learning Bayesian Network

- The goal:**
- Given set of independent samples (**assignments** of random variables), find the **best** (the most likely?) Bayesian Network (both DAG and CPDs)



# Learning Graphical Models



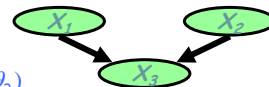
- Scenarios:
  - completely observed GMs
    - directed
    - undirected
  - partially observed GMs
    - directed
    - undirected (an open research topic)
- Estimation principles:
  - Maximal likelihood estimation (MLE)
  - Bayesian estimation
- We use **learning** as a name for the process of **estimating the parameters**, and in some cases, the topology of the network, from data.

## The basic idea underlying MLE



- Likelihood:

$$L(\theta | X) = p(X | \theta) = p(X_1 | \theta_1) p(X_2 | \theta_2) p(X_3 | X_1, X_2, \theta_3)$$



- Log-Likelihood:

$$l(\theta | X) = \log p(X | \theta) = \log p(X_1 | \theta_1) + \log p(X_2 | \theta_2) + \log p(X_3 | X_1, X_2, \theta_3)$$

- Data log-likelihood

$$\begin{aligned}
 l(\theta | DATA) &= \log \prod_n p(X^{(n)} | \theta) \\
 &= \sum_n \log p(X_1^{(n)} | \theta_1) + \sum_n \log p(X_2^{(n)} | \theta_2) + \sum_n \log p(X_3^{(n)} | X_1^{(n)} X_2^{(n)}, \theta_3)
 \end{aligned}$$

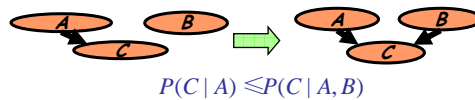
- MLE  $\{\theta_1, \theta_2, \theta_3\}_{MLE} = \arg \max l(\theta | DATA)$

$$\theta_1^* = \arg \max_n \sum \log p(X_1^{(n)} | \theta_1), \quad \theta_2^* = \arg \max_n \sum \log p(X_2^{(n)} | \theta_2), \quad \theta_3^* = \arg \max_n \sum \log p(X_3^{(n)} | X_1^{(n)} X_2^{(n)}, \theta_3)$$

# Learning Bayesian Network

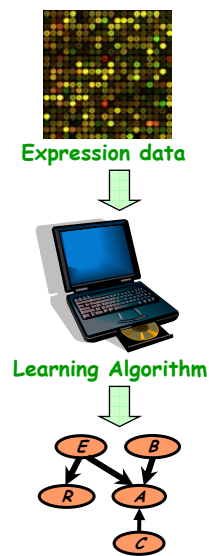


- Learning of best CPDs *given DAG* is easy
  - collect statistics of values of each node given specific assignment to its parents
- Learning of the graph topology (structure) is **NP-hard**
  - heuristic search must be applied, generally leads to a **locally** optimal network
- Overfitting
  - It turns out, that richer structures give higher likelihood  $P(D|G)$  to the data (adding an edge is always preferable)



- more parameters to fit  $\Rightarrow$  more freedom  $\Rightarrow$  always exist more "optimal" CPD(C)
- We prefer *simpler* (more explanatory) networks
  - **Practical** scores **regularize** the likelihood improvement complex networks.

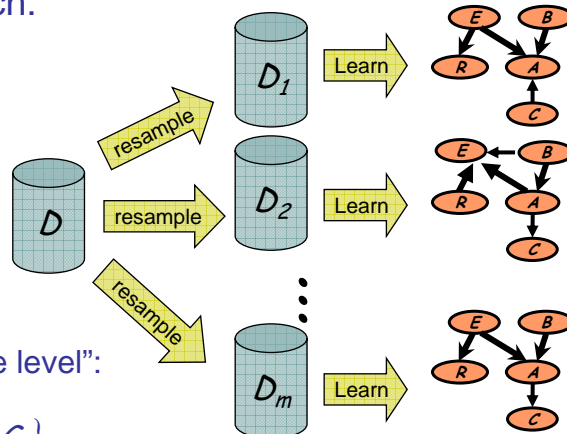
# BN Learning Algorithms



- Structural EM (Friedman 1998)
  - The original algorithm
- Sparse Candidate Algorithm (Friedman et al.)
  - Discretizing array signals
  - Hill-climbing search using local operators: add/delete/swap of a single edge
  - Feature extraction: Markov relations, order relations
  - Re-assemble high-confidence sub-networks from features
- Module network learning (Segal et al.)
  - Heuristic search of structure in a "module graph"
  - Module assignment
  - Parameter sharing
  - Prior knowledge: possible regulators (TF genes)

## Confidence Estimates

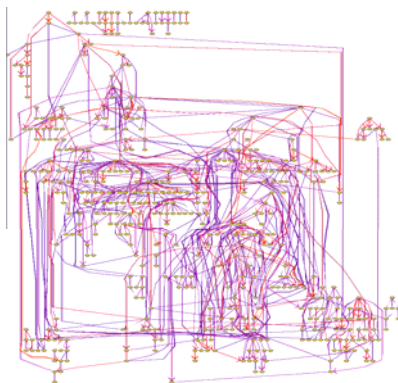
Bootstrap approach:



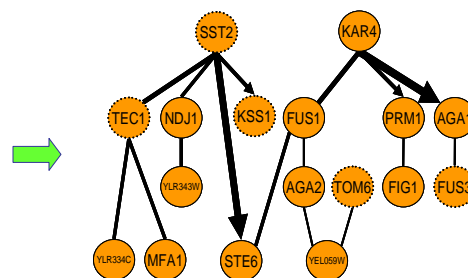
Estimate "Confidence level":

$$C(f) = \frac{1}{m} \sum_{i=1}^m 1\{f \in G_i\}$$

## Results from SCA + feature extraction (Friedman et al.)



The initially learned network of ~800 genes



The "mating response" substructure





## Probabilistic inference on Graphical Models

## Recap of Basic Prob. Concepts



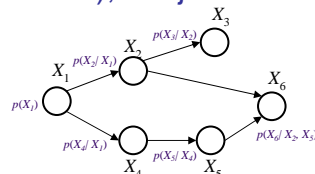
- Joint probability dist. on multiple variables:

$$P(X_1, X_2, X_3, X_4, X_5, X_6) \\ = P(X_1)P(X_2 | X_1)P(X_3 | X_1, X_2)P(X_4 | X_1, X_2, X_3)P(X_5 | X_1, X_2, X_3, X_4)P(X_6 | X_1, X_2, X_3, X_4, X_5)$$

- If  $X_i$ 's are **independent**: ( $P(X_i | \cdot) = P(X_i)$ )

$$P(X_1, X_2, X_3, X_4, X_5, X_6) \\ = P(X_1)P(X_2)P(X_3)P(X_4)P(X_5)P(X_6) = \prod_i P(X_i)$$

- If  $X_i$ 's are **conditionally independent** (as described by a **GM**), the joint can be factored to simpler products, e.g.,



$$P(X_1, X_2, X_3, X_4, X_5, X_6) \\ = P(X_1) P(X_2 | X_1) P(X_3 | X_2) P(X_4 | X_1) P(X_5 | X_4) P(X_6 | X_2, X_5)$$

## Probabilistic Inference



- We now have compact representations of probability distributions: **Graphical Models**
- A GM  $\mathcal{M}$  describes a unique probability distribution  $\mathcal{P}$
- How do we answer **queries** about  $\mathcal{P}$ ?
- We use **inference** as a name for the process of computing answers to such queries

## Query 1: Likelihood



- Most of the queries one may ask involve **evidence**
  - Evidence  $e$  is an assignment of values to a set  $E$  variables in the domain
  - Without loss of generality  $E = \{X_{k+1}, \dots, X_n\}$
- Simplest query: compute probability of evidence

$$P(e) = \sum_{x_1} \dots \sum_{x_k} P(x_1, \dots, x_k, e)$$

- this is often referred to as computing the **likelihood** of  $e$

## Query 2: Conditional Probability



- Often we are interested in the **conditional probability distribution** of a variable given the evidence

$$P(X|e) = \frac{P(X,e)}{P(e)} = \frac{P(X,e)}{\sum_x P(X=x,e)}$$

- this is the **a posteriori belief** in  $X$ , given evidence  $e$
- We usually query a subset  $Y$  of all domain variables  $X=\{Y,Z\}$  and "don't care" about the remaining,  $Z$ :

$$P(Y|e) = \sum_z P(Y,Z=z|e)$$

- the process of summing out the "don't care" variables  $z$  is called **marginalization**, and the resulting  $P(Y|e)$  is called a **marginal** prob.

## Applications of a posteriori Belief



- Prediction:** what is the probability of an outcome given the starting condition



- the query node is a descendent of the evidence

- Diagnosis:** what is the probability of disease/fault given symptoms



- the query node an ancestor of the evidence

- Learning** under partial observation

- fill in the unobserved values under an "EM" setting (more later)

- The directionality of information flow between variables is not restricted by the directionality of the edges in a GM

- probabilistic inference can combine evidence from all parts of the network

## Query 3: Most Probable Assignment



- In this query we want to find the **most probable joint assignment** (MPA) for *some* variables of interest
- Such reasoning is usually performed under some given evidence  $e$ , and ignoring (the values of) other variables  $z$  :

$$\text{MPA}(Y | e) = \arg \max_y P(y | e) = \arg \max_y \sum_z P(y, z | e)$$

- this is the **maximum a posteriori** configuration of  $y$ .

## Applications of MPA



- Classification
  - find most likely label, given the evidence
- Explanation
  - what is the most likely scenario, given the evidence

Cautionary note:

- The MPA of a variable depends on its "context"---the set of variables been jointly queried
- Example:
  - MPA of  $X$ ?
  - MPA of  $(X, Y)$ ?

$x$	$y$	$P(x, y)$
0	0	0.35
0	1	0.05
1	0	0.3
1	1	0.3

## Complexity of Inference



### Thm:

Computing  $P(X = x \mid e)$  in a GM is NP-hard

- Hardness does not mean we cannot solve inference
  - It implies that we cannot find a general procedure that works efficiently for arbitrary GMs
  - For particular families of GMs, we can have provably efficient procedures

## Approaches to inference



- Exact inference algorithms
  - The elimination algorithm ✓
  - The junction tree algorithms ✓ (but will not cover in detail here)
- Approximate inference techniques
  - Stochastic simulation / sampling methods ✓
  - Markov chain Monte Carlo methods ✓
  - Variational algorithms (later lectures)

## Marginalization and Elimination

A signal transduction pathway:



What is the likelihood that protein E is active?

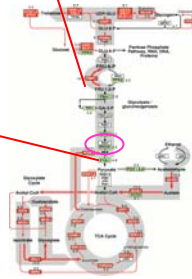
- Query:  $P(e)$

$$P(e) = \sum_d \sum_c \sum_b \sum_a P(a, b, c, d, e)$$

a naïve summation needs to enumerate over an exponential number of terms

- By chain decomposition, we get

$$= \sum_d \sum_c \sum_b \sum_a P(a)P(b|a)P(c|b)P(d|c)P(e|d)$$



## Elimination on Chains



- Rearranging terms ...

$$\begin{aligned} P(e) &= \sum_d \sum_c \sum_b \sum_a P(a)P(b|a)P(c|b)P(d|c)P(e|d) \\ &= \sum_d \sum_c \sum_b P(c|b)P(d|c)P(e|d) \sum_a P(a)P(b|a) \end{aligned}$$

## Elimination on Chains



- Now we can perform innermost summation

$$\begin{aligned}
 P(e) &= \sum_d \sum_c \sum_b P(c|b)P(d|c)P(e|d) \underbrace{\sum_a P(a)P(b|a)}_{\text{local cost}} \\
 &= \sum_d \sum_c \sum_b P(c|b)P(d|c)P(e|d)p(b)
 \end{aligned}$$

- This summation "eliminates" one variable from our summation argument at a "local cost".

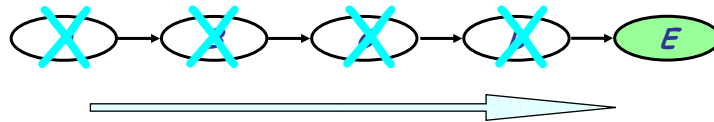
## Elimination in Chains



- Rearranging and then summing again, we get

$$\begin{aligned}
 P(e) &= \sum_d \sum_c \sum_b P(c|b)P(d|c)P(e|d)p(b) \\
 &= \sum_d \sum_c P(d|c)P(e|d) \underbrace{\sum_b P(c|b)p(b)}_{\text{local cost}} \\
 &= \sum_d \sum_c P(d|c)P(e|d)p(c)
 \end{aligned}$$

## Elimination in Chains



- Eliminate nodes one by one all the way to the end, we get

$$P(e) = \sum_d P(e | d) p(d)$$

Complexity:

- Each step costs  $O(|Val(X_i)| * |Val(X_{i+1})|)$  operations:  $O(kn^2)$
- Compare to naïve evaluation that sums over joint values of  $n-1$  variables  $O(n^k)$

## Inference on General GM via Variable Elimination



**General idea:**

- Write query in the form

$$P(X_1, e) = \sum_{x_n} \dots \sum_{x_3} \sum_{x_2} \prod_i P(x_i | pa_i)$$

- this suggests an "elimination order" of latent variables to be marginalized
- Iteratively
  - Move all irrelevant terms outside of innermost sum
  - Perform innermost sum, getting a new term
  - Insert the new term into the product

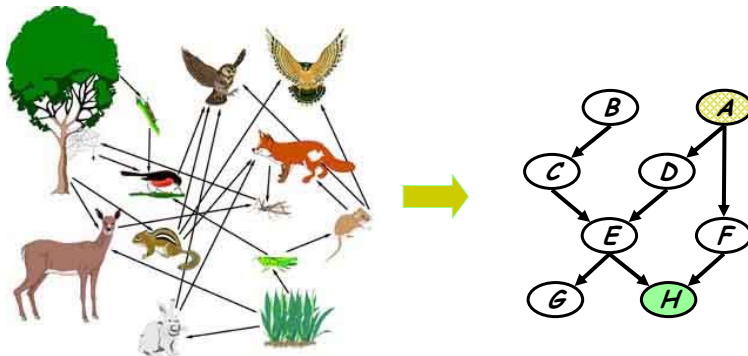
- wrap-up

$$P(X_1 | e) = \frac{P(X_1, e)}{P(e)}$$

## A more complex network



### A food web

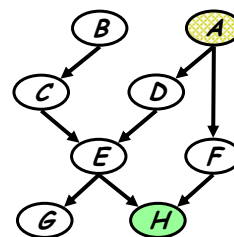


What is the probability that hawks are leaving given that the grass condition is poor?

## Example: Variable Elimination



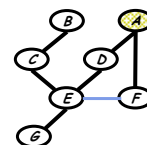
- Query:  $P(A | h)$ 
  - Need to eliminate:  $B, C, D, E, F, G, H$
- Initial factors:
 
$$P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f)$$
- Choose an elimination order:  $H, G, F, E, D, C, B$



A regulatory network

- Step 1:
  - **Conditioning** (fix the evidence node (i.e.,  $h$ ) to its observed value (i.e.,  $\tilde{h}$ ):
 
$$m_h(e, f) = p(h = \tilde{h} | e, f)$$
  - This step is isomorphic to a marginalization step:

$$m_h(e, f) = \sum_h p(h | e, f) \delta(h = \tilde{h})$$



## Example: Variable Elimination

- Query:  $P(B | h)$ 
  - Need to eliminate:  $B, C, D, E, F, G$

- Initial factors:

$$P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f)$$

$$\Rightarrow P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f)$$

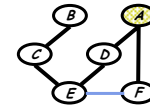
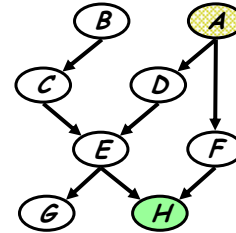
- Step 2: Eliminate  $G$

- compute

$$m_g(e) = \sum p(g|e) = 1$$

$$\Rightarrow P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)m_g(e)m_h(e,f)$$

$$= P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)m_h(e,f)$$



## Example: Variable Elimination

- Query:  $P(B | h)$ 
  - Need to eliminate:  $B, C, D, E, F$

- Initial factors:

$$P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f)$$

$$\Rightarrow P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f)$$

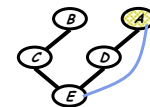
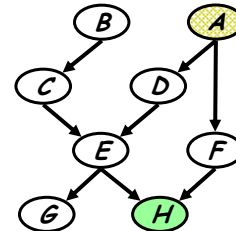
$$\Rightarrow P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)m_h(e,f)$$

- Step 3: Eliminate  $F$

- compute

$$m_f(e, a) = \sum_f p(f|a)m_h(e, f)$$

$$\Rightarrow P(a)P(b)P(c|b)P(d|a)P(e|c,d)m_f(a, e)$$

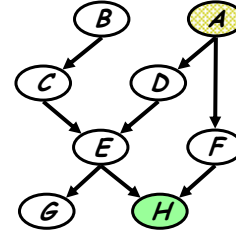


## Example: Variable Elimination

- Query:  $P(B | h)$ 
  - Need to eliminate:  $B, C, D, E$

- Initial factors:

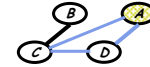
$$\begin{aligned}
 &P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
 \Rightarrow &P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|b)P(d|a)\underline{P(e|c,d)m_f(a,e)}
 \end{aligned}$$



- Step 4: Eliminate  $E$

- compute  $m_e(a, c, d) = \sum_e p(e|c, d)m_f(a, e)$

$$\Rightarrow P(a)P(b)P(c|b)P(d|a)\underline{m_e(a, c, d)}$$

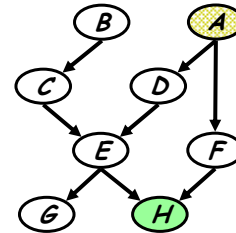


## Example: Variable Elimination

- Query:  $P(B | h)$ 
  - Need to eliminate:  $B, C, D$

- Initial factors:

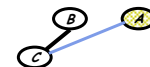
$$\begin{aligned}
 &P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
 \Rightarrow &P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|b)P(d|a)P(e|c,d)P(f|a)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|b)P(d|a)P(e|c,d)m_f(a, e) \\
 \Rightarrow &P(a)P(b)P(c|b)\underline{P(d|a)m_e(a, c, d)}
 \end{aligned}$$



- Step 5: Eliminate  $D$

- compute  $m_d(a, c) = \sum_d p(d|a)m_e(a, c, d)$

$$\Rightarrow P(a)P(b)P(c|d)\underline{m_d(a, c)}$$



## Example: Variable Elimination

- Query:  $P(B | h)$ 
  - Need to eliminate:  $B, C$

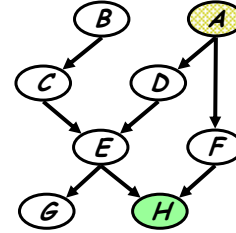
- Initial factors:

$$\begin{aligned}
 &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)m_f(a,e) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)m_e(a,c,d) \\
 \Rightarrow &P(a)P(b)P(c|d)m_d(a,c)
 \end{aligned}$$

- Step 6: Eliminate  $C$

- compute  $m_c(a,b) = \sum_c p(c|b)m_d(a,c)$

$$\Rightarrow P(a)P(b)m_c(a,b)$$



## Example: Variable Elimination

- Query:  $P(B | h)$ 
  - Need to eliminate:  $B$

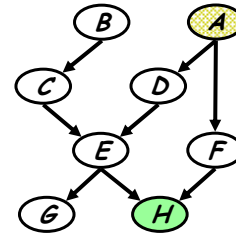
- Initial factors:

$$\begin{aligned}
 &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)m_f(a,e) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)m_e(a,c,d) \\
 \Rightarrow &P(a)P(b)P(c|d)m_d(a,c) \\
 \Rightarrow &P(a)P(b)m_c(a,b)
 \end{aligned}$$

- Step 7: Eliminate  $B$

- compute  $m_b(a) = \sum_b p(b)m_c(a,b)$

$$\Rightarrow P(a)m_b(a)$$



## Example: Variable Elimination

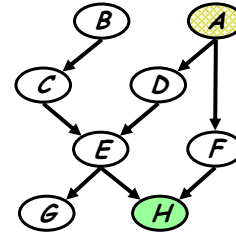


- Query:  $P(B | h)$

- Need to eliminate:  $\{ \}$

- Initial factors:

$$\begin{aligned}
 &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)m_h(e,f) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)P(e|c,d)m_f(a,e) \\
 \Rightarrow &P(a)P(b)P(c|d)P(d|a)m_e(a,c,d) \\
 \Rightarrow &P(a)P(b)P(c|d)m_d(a,c) \\
 \Rightarrow &P(a)P(b)m_c(a,b) \\
 \Rightarrow &P(a)m_b(a)
 \end{aligned}$$



- Step 8: Wrap-up  $p(a, \tilde{h}) = p(a)m_b(a)$ ,  $p(\tilde{h}) = \sum_a p(a)m_b(a)$   
 $\Rightarrow p(a | \tilde{h}) = \frac{p(a)m_b(a)}{\sum_a p(a)m_b(a)}$

## Complexity of variable elimination



- Suppose in one elimination step we compute

$$\begin{aligned}
 m_x(y_1, \dots, y_k) &= \sum_x m'_x(x, y_1, \dots, y_k) \\
 m'_x(x, y_1, \dots, y_k) &= \prod_{i=1}^k m_i(x, y_{c_i})
 \end{aligned}$$

This requires

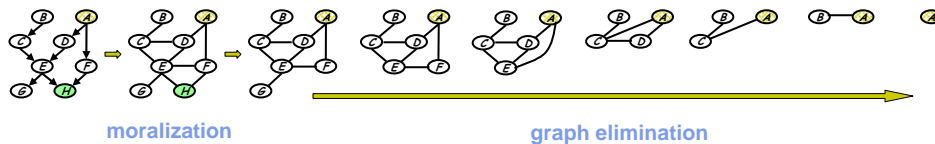
- $k \cdot |\text{Val}(X)| \cdot \prod_i |\text{Val}(Y_{c_i})|$  multiplications
  - For each value for  $x, y_1, \dots, y_k$  we do  $k$  multiplications
- $|\text{Val}(X)| \cdot \prod_i |\text{Val}(Y_{c_i})|$  additions
  - For each value of  $y_1, \dots, y_k$ , we do  $|\text{Val}(X)|$  additions

Complexity is **exponential** in number of variables in the intermediate factor

# Understanding Variable Elimination



- A graph elimination algorithm

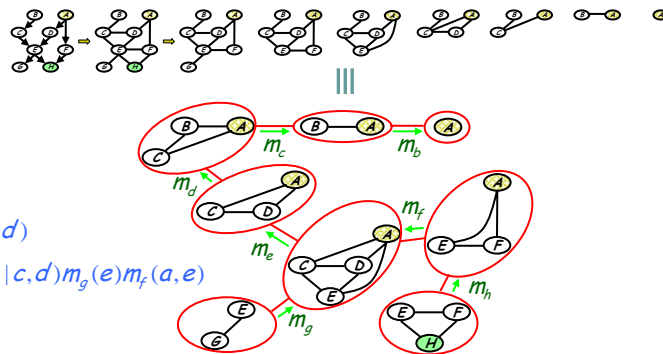


- Intermediate terms correspond to the **cliques** resulted from elimination
  - “good” elimination orderings lead to **small cliques** and hence reduce complexity (what will happen if we eliminate "e" first in the above graph?)
  - finding the optimum ordering is NP-hard, but for many graph optimum or near-optimum can often be heuristically found
- Applies to undirected GMs

# From Elimination to Message Passing



- Our algorithm so far answers only one query (e.g., on one node), do we need to do a complete elimination for every such query?
- Elimination  $\equiv$  message passing on a **clique tree**

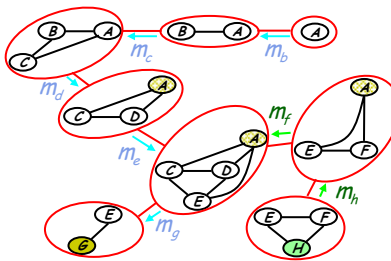


- Messages can be reused

## From Elimination to Message Passing



- Our algorithm so far answers only one query (e.g., on one node), do we need to do a complete elimination for every such query?
- Elimination  $\equiv$  message passing on a **clique tree**
  - Another query ...



- Messages  $m_f$  and  $m_h$  are reused, others need to be recomputed

## A Sketch of the Junction Tree Algorithm



- **The algorithm**
  - Construction of junction trees --- a special **clique tree**
  - Propagation of probabilities --- a message-passing protocol
- Results in marginal probabilities of all cliques --- solves all queries in a single run
- A **generic** exact inference algorithm for any GM
- **Complexity**: exponential in the size of the maximal clique --- a good elimination order often leads to small maximal clique, and hence a good (i.e., thin) JT
- Many well-known algorithms are special cases of JT
  - Forward-backward, Kalman filter, Peeling, Sum-Product ...

## Approaches to inference



- Exact inference algorithms
  - The elimination algorithm ✓
  - The junction tree algorithms ✓ (but will not cover in detail here)
- Approximate inference techniques
  - Stochastic simulation / sampling methods ✓
  - Markov chain Monte Carlo methods ✓
  - Variational algorithms (later lectures)

## Monte Carlo methods



- Draw random samples from the desired distribution
- Yield a stochastic representation of a complex distribution
  - marginals and other expectations can be approximated using sample-based averages

$$E[f(x)] = \frac{1}{N} \sum_{t=1}^N f(x^{(t)})$$

- Asymptotically exact and easy to apply to arbitrary models
- Challenges:
  - how to draw samples from a given dist. (not all distributions can be trivially sampled)?
  - how to make better use of the samples (not all sample are useful, or equally useful, see an example later)?
  - how to know we've sampled enough?

- 
- Bayesian network diagram showing the relationships between variables and their conditional probability tables (CPTs):
- Burglary** (Node): CPT  $P(B)$ 

	.001
--	------
  - Earthquake** (Node): CPT  $P(E)$ 

	.002
--	------
  - Alarm** (Node): CPT  $P(A|B, E)$ 

B	E	P(A)
T	T	.95
T	F	.94
F	T	.29
F	F	.001
  - JohnCalls** (Node): CPT  $P(J|A)$ 

A	P(J)
T	.90
F	.05
  - MaryCalls** (Node): CPT  $P(M|A)$ 

A	P(M)
T	.70
F	.01

E0	B0	A0	M0	J0
E0	B0	A0	M0	J0
E0	B0	A0	M0	J1
E0	B0	A0	M0	J0
E0	B0	A0	M0	J0
E0	B0	A0	M0	J0
E1	B0	A1	M1	J1
E0	B0	A0	M0	J0
E0	B0	A0	M0	J0
E0	B0	A0	M0	J0

- |           |           |           |           |           |
|-----------|-----------|-----------|-----------|-----------|
| E0        | B0        | A0        | M0        | J0        |
| E0        | B0        | A0        | M0        | J0        |
| E0        | B0        | A0        | M0        | J1        |
| E0        | B0        | A0        | M0        | J0        |
| E0        | B0        | A0        | M0        | J0        |
| E0        | B0        | A0        | M0        | J0        |
| <b>E1</b> | <b>B0</b> | <b>A1</b> | <b>M1</b> | <b>J1</b> |
| E0        | B0        | A0        | M0        | J0        |
| E0        | B0        | A0        | M0        | J0        |
| E0        | B0        | A0        | M0        | J0        |

## Monte Carlo methods (cond.)



- Direct Sampling
  - We have seen it.
  - Very difficult to populate a high-dimensional state space
- Rejection Sampling
  - Create samples like direct sampling, only count samples which is consistent with given evidences.
- ....
- Markov chain Monte Carlo (MCMC)

## Markov chain Monte Carlo



- Samples are obtained from a Markov chain (of sequentially evolving distributions) whose stationary distribution is the desired  $p(x)$
- Gibbs sampling
  - we have variable set to  $X = \{x_1, x_2, x_3, \dots, x_N\}$
  - at each step one of the variables  $X_i$  is selected (at random or according to some fixed sequences)
  - the conditional distribution  $p(x_i | X_{-i})$  is computed
  - a value  $x_i$  is sampled from this distribution
  - the sample  $x_i$  replaces the previous of  $X_i$  in  $X$ .

# MCMC



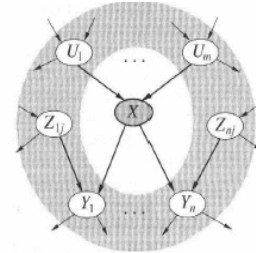
- Markov-Blanket

- A variable is independent from others, given its parents, children and children's parents. d-separation.

$$\Rightarrow p(X_i | X_{-i}) = p(X_i | MB(X_i))$$

- Gibbs sampling

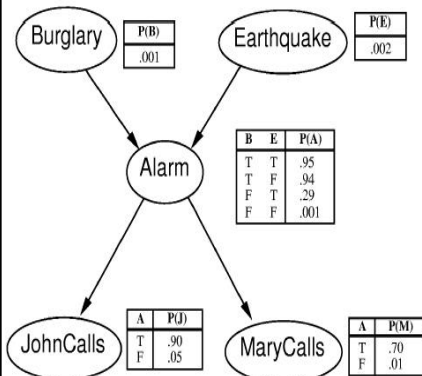
- Create a random sample. Every step, choose one variable and sample it by  $P(X|MB(X))$  based on previous sample.



$MB(A) = \{B, E, J, M\}$

$MB(E) = \{A, B\}$

# MCMC

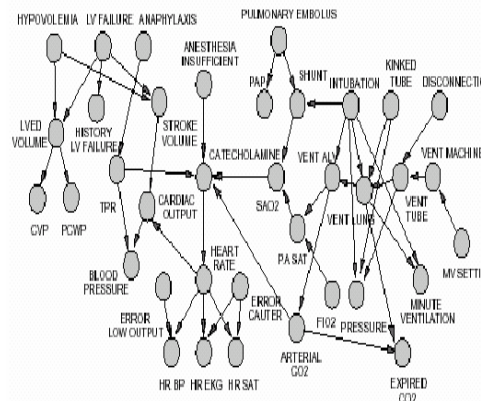


- To calculate  $P(J|B1, M1)$
- Choose  $(B1, E0, A1, M1, J1)$  as a start
- **Evidences** are  $B1, M1$ , **variables** are  $A, E, J$ .
- Choose next variable as A
- Sample A by  $P(A|MB(A)) = P(A|B1, E0, M1, J1)$  suppose to be false.
- $(B1, E0, A0, M1, J1)$
- Choose next random variable as E, sample  $E \sim P(E|B1, A0)$
- ...

## Complexity for Approximate Inference



- Inference problem is NP-hard.
- Approximate Inference will not reach the exact probability distribution in finite time, but only close to the value.
- Often much faster than exact inference when BN is **big** and **complex** enough. In MCMC, only consider  $P(X|MB(X))$  but not the whole network.



## Covariance Selection



- Multivariate Gaussian over all continuous expressions

$$p([x_1, \dots, x_n]) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\bar{x} - \mu)^T \Sigma^{-1}(\bar{x} - \mu)\right\}$$

- The precision matrix  $K = \Sigma^{-1}$  reveals the topology of the (undirected) network

$$E(x_i | x_{-i}) = \sum_j (K_{ij} / K_{ii}) x_j$$

- Edge  $\sim |K_{ij}| > 0$

- Learning Algorithm: Covariance selection

- Want a sparse matrix
  - Regression for each node with degree constraint (Dobra et al.)
  - Regression for each node with hierarchical Bayesian prior (Li, et al)

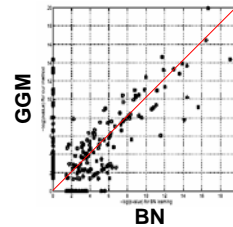
# Gene modules from identified using GGM (Li, Yang and Xing)



Table 1: modules with multiple regulators and more than 5 regulated genes.

regulator set	annotated/ALL	reference of co-regulators	common processes or function of regulated genes
ACE2,SWI5	6/6		4/6 cell proliferation, $p=0.023$
ASH1,SWI4	11/12		4/11 cell wall organization and biogenesis, $p=0.003$
CIN5,MET4	6/6		2/6 copper ion import, $p=0.0002$
DIG1,STE12	14/14	functional and physical interaction(Tedford et al 1997)	10/14 conjugation, $p=1.75e-14$
PHL1,RAP1	25/25	share motif(Davide et al)	24/25 protein biosynthesis, $p=7.28e-21$
PRH2,MCM1,NDD1	12/12	co-TPs(CYGD)	6/12 cell proliferation, $p=0.01$
GAT3,MAL13,RGM1	5/16		5/5 telomerase-independent telomere maintenance, $p=1.84e-13$
GAT3,RAP1,YAP5	34/45		25/34 protein biosynthesis, $p=1.12e-15$
GCR1,GCR2,RAP1	6/6		6/6 energy pathways, $p=1.65e-08$
HIR1,HIR2	6/6	co-TPs(Spector et al 1997)	6/6 chromatin assembly or disassembly, $p=1.23e-14$
HIR2,STP2	6/6		6/6 regulation of transcription, mating-type specific, $p=7.35e-12$
HSF1,MSN4	8/8	co-TPs(Jaffrey et al, 2002)	3/8 protein folding, $p=9.83e-4$
MAL13,MSN4,RGM1	5/7		3/5 telomerase-independent telomere maintenance, $p=1.05e-6$
MBP1,SWI4	8/8		3/8 microtubule cytoskeleton organization and biogenesis, $p=0.005$
MBP1,SWI4,SWI6	10/10	functional and physical interaction(CYGD)	6/10 mitotic cell cycle, $p=7.7e-06$
MBP1,SWI6	24/24	functional and physical interaction(CYGD)	10/24 cell cycle, $p=5.9e-04$
MET31,MET4	5/8	in the same complex(Pierre-Louis et al 1998)	8/8 sulfur metabolism, $p=7.75e-11$
NDD1,SWI6	6/6		5/6 cell organization and biogenesis, $p=0.03$

A comparison of BN and GGM:



## 2: Protein-DNA Interaction Network



- Expression networks are not necessarily **causal**

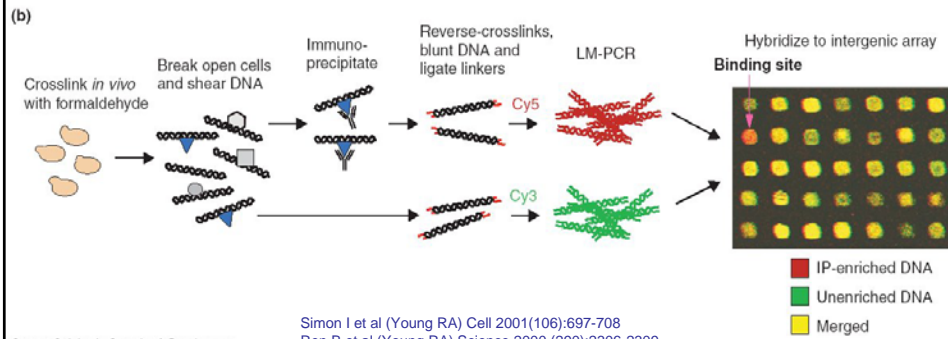
- BNs are indefinable only up to **Markov equivalence**:



can give the same optimal score, but not further distinguishable under a likelihood score unless further experiment from perturbation is performed

- GGM have yields functional modules, but no causal semantics
- TF-motif interactions provide direct evidence of casual, regulatory dependencies among genes
  - stronger evidence than expression correlations
  - indicating presence of binding sites on target gene -- more easily verifiable
  - disadvantage: often very noisy, only applies to cell-cultures, restricted to known TFs ...

# ChIP-chip analysis



## Advantages:

- Identifies "all" the sites where a TF binds "*in vivo*" under the experimental condition.

## Limitations:

- Expense: Only 1 TF per experiment
- Feasibility: need an antibody for the TF.
- Prior knowledge: need to know what TF to test.