

# Graphical Models for Inference and Decision Making

## Unit 6: Inference in Graphical Models: Other Algorithms

Instructor: Kathryn Blackmond Laskey  
Spring 2019

## Learning Objectives

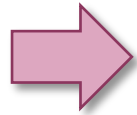
- Describe the major categories of exact inference algorithms and their strengths and weaknesses
- Describe basics of bucket elimination algorithm
- Describe the underlying idea behind Monte Carlo approximation
- Describe how the major Monte Carlo inference algorithms work
- Implement likelihood weighting inference (fixed network, fixed query) in a computing environment with programming and/or macro capability and random number generator
  - Generate the sample
  - Generating the estimate from the sample
- Implement Gibbs sampling inference (fixed network, fixed query) in a computing environment with programming and/or macro capability and random number generator
  - Generate the sample
  - Generate the estimate from the sample
- Describe what a variational approximation is and how variational methods work

# Taxonomy of Graphical Model Inference Algorithms

- Exact graph theoretic
  - Pearl's tree algorithm and its modifications
  - Junction tree algorithm
  - Influence diagram reduction
- Exact factorization based
  - Symbolic probabilistic inference
  - Bucket elimination
- Approximate (deterministic)
  - Loopy belief propagation
  - Variational approximation
  - Various special case approaches
- Approximate (Monte Carlo)
  - Monte Carlo simulation
- Lifted inference in relational models

***All these algorithms can be used to solve the canonical inference problem: find the posterior probability distribution for some variable(s) given exact or virtual evidence about other variable(s)***

## Unit 6 Outline



- Exact Algorithms
- Monte Carlo Approximation Algorithms
- Deterministic Approximation Algorithms
- Inference in Relational Models

## Inference as Factor Manipulation

- We have treated inference as a problem of message passing along links in a graph
- Algebraic algorithms view inference as a problem of efficient factorization of a complex expression

- A simple example:

- Goal: compute  $P(C)$

- Method 1:

- »  $P(c_i) = \sum_j \sum_k P(c_i | a_j, b_k) P(a_j) P(b_k)$

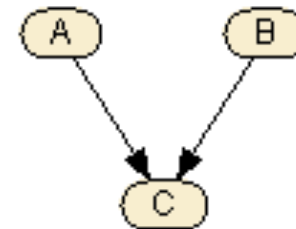
- »  $\#A \times \#B \times 2$  multiplications and  $\#A \times \#B - 1$  additions for each  $P(c_i)$

- Method 2: Eliminate B and then eliminate A

- »  $P(c_i) = \sum_j P(a_j) \sum_k P(c_i | a_j, b_k) P(b_k)$

- »  $\#A \times (\#B + 1)$  multiplications and  $(\#A - 1) \times (\#B - 1)$  additions for each  $P(c_i)$

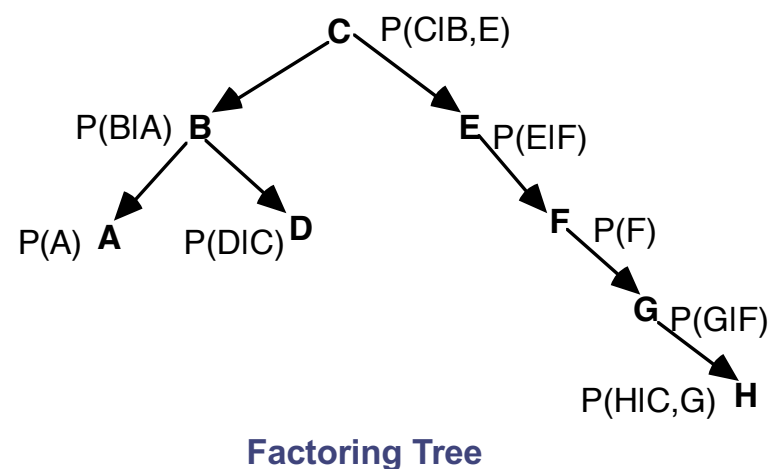
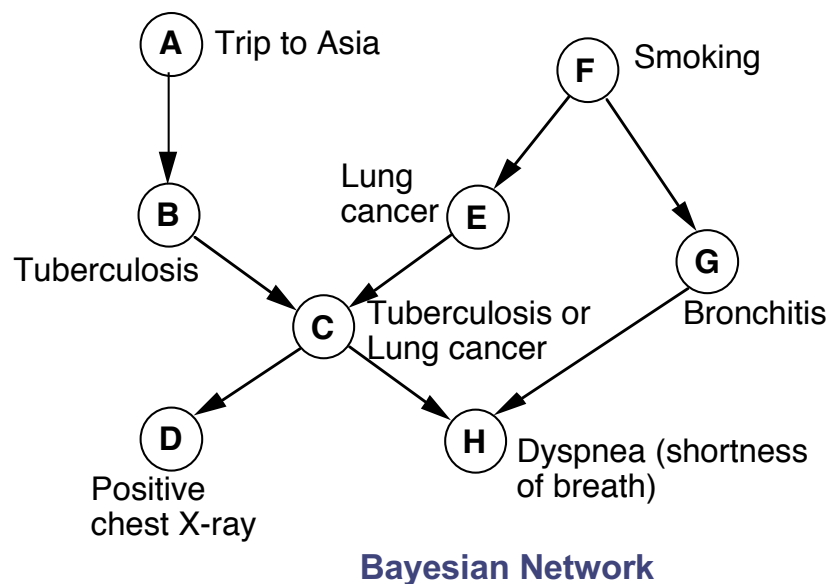
- Method 2 saves computation, especially as  $\#A$  and  $\#B$  get large



## Symbolic Probabilistic Inference: Inference as Factor Manipulation

- Compile network into a factorization tree.
  - The tree represents a way of factorizing the joint distribution
  - Nodes of the factorization tree are nodes in the Bayesian network
  - There are many possible trees but some are more efficient for specific queries
  - Tree construction algorithm is heuristic
- Query  $P(U \mid V)$  processing:
  - Query is received at root node
  - When a node receives a query, it:
    - » decomposes it into queries to send to nodes below in the graph
    - » waits for responses from below
    - » reaggregates responses (summing out variables that can be summed out) and reports result
- Features of SPI algorithm:
  - Query processing is designed so that multiplications are brought outside of sums whenever possible (eliminate variables before combining factors)
  - Algorithm does only computations needed to process particular query
  - Efficient implementations cache results so that they can be reused

## Search Tree and Factorization

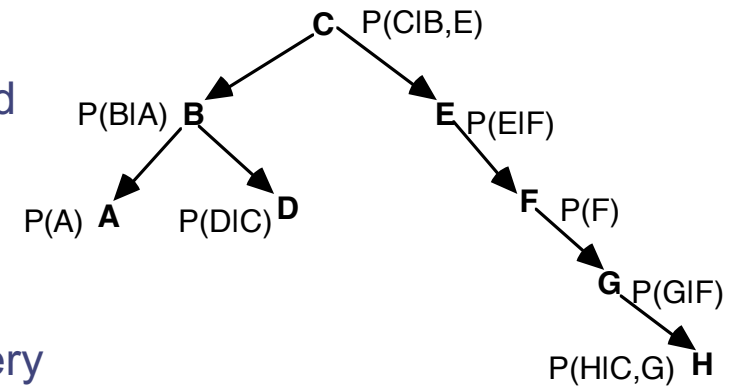


This tree corresponds to combining the factors in the following order:

$$P(C|B,E) \left( P(B|A) \left( P(A) P(D|C) \right) \right) \left( P(E|F) \left( P(F) \left( P(G|F) \left( P(H|C,G) \right) \right) \right) \right)$$

## SPI Query

- An SPI query consists of
  - A set of nodes for which a distribution is required
    - » e.g., compute  $P(U|V, W=w)$
  - A set of nodes needed to respond to the query
- Example query: compute  $P(D)$ 
  - Compute distributions needed to respond to query
    - » Needed:  $P(A) P(B|A) P(C|B,E) P(D|C) P(E|F) P(F)$
    - » Not needed:  $P(G|F) P(H|C,G)$
  - Node C receives query: Target node D, distributions needed are A,B,C,D,E,F
    - » Node C decomposes query to send to its children
      - Send to children parts of query they can process
      - C's factor is  $P(C|B,E)$
      - C is added to target for all children, B to left branch, E to right branch
      - Needed factors include only factors on branch being queried
    - » Query to B: Compute factor  $f(C,D,B)$ ; distributions needed are B, A, D
    - » Query to E: Compute factor  $f(C,D,E)$ ; distributions needed are E, F
  - When node C receives response from children it computes  $P(C|B,E)f(C,D,B)f(C,D,E)$  and sums out variables B,C and E
  - Note that A and F are summed out before queries from children are sent back up to node C





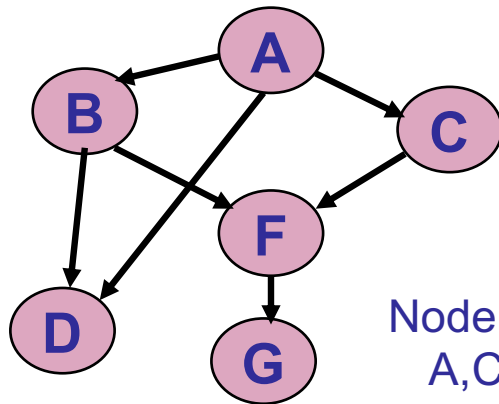
## Bucket Elimination - Overview

- Bucket elimination (Dechter, 1999) is a unifying framework for complex reasoning problems such as:
  - Computing posterior probabilities
  - Most probable explanation (MPE)
  - Maximum a posteriori hypothesis (MAP)
  - Maximum expected utility (MEU)
  - Constraint satisfaction
  - Propositional satisfiability
- Generalizes dynamic programming
- “Buckets” are an organizational device for algebraic manipulations involved in efficiently evaluating an algebraic expression

## Bucket Elimination Algorithm

1. Select a node order
2. Partition functions on graph into buckets
  - Place functions mentioning  $X_i$  but not any variable with higher index into bucket for  $X_i$
  - For posterior inference these functions are the belief tables in the Bayesian network and the findings
  - For posterior inference bucket function is product of belief tables mentioning  $X_i$
3. Process buckets backwards relative to node order
  - Process bucket for  $X_i$  means to eliminate variable  $X_i$
  - For posterior inference eliminating means summing
4. After elimination, place new function in bucket of highest-numbered variable in its scope

## Bucket Elimination Example



Node Ordering:  
A, C, B, F, D, G

Goal: Find  $P(A|G=1)$

### 1. Initial buckets:

$\text{bucket}_G = P(G|F)$ , finding:  $G=1$

$\text{bucket}_D = P(D|A, B)$

$\text{bucket}_F = P(F|B, C)$

$\text{bucket}_B = P(B|A)$

$\text{bucket}_C = P(C|A)$

$\text{bucket}_A = P(A)$

### 2. Process bucket G:

$$\lambda_G(f) = P(G = 1|f)$$

*Note: this operation (trivially) sums over all values of G for which finding is true*

#### Remaining buckets:

$\text{bucket}_D = P(D|A, B)$

$\text{bucket}_F = P(F|B, C) \lambda_G(F)$

$\text{bucket}_B = P(B|A)$

$\text{bucket}_C = P(C|A)$

$\text{bucket}_A = P(A)$

### 3. Process bucket D:

$$\lambda_D(b, a) = \sum_d P(d|b, a)$$

*Note: this sum is equal to 1*

#### Remaining buckets:

$\text{bucket}_F = P(F|B, C) \lambda_G(F)$

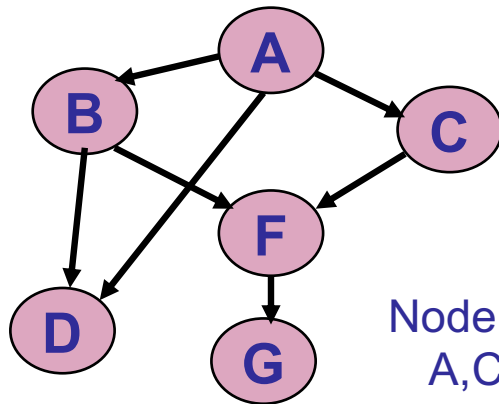
$\text{bucket}_B = P(B|A) \lambda_D(B, A)$

$\text{bucket}_C = P(C|A)$

$\text{bucket}_A = P(A)$

Example taken from undated presentation notes by Dechter and Lavee

## Bucket Elimination Example



Node Ordering:  
A, C, B, F, D, G

Goal: Find  $P(A|G=1)$

### 4. Process bucket F:

$$\lambda_F(b, c) = \sum_f P(f|b, c) \lambda_G(f)$$

Remaining buckets:

$$\text{bucket}_B = P(B|A) \lambda_D(B, A) \lambda_F(B, C)$$

$$\text{bucket}_C = P(C|A)$$

$$\text{bucket}_A = P(A)$$

### 5. Process bucket B:

$$\lambda_B(c, a) = \sum_b P(b|a) \lambda_D(b, a) \lambda_F(b, c)$$

Remaining buckets:

$$\text{bucket}_C = P(C|A) \lambda_B(C, A)$$

$$\text{bucket}_A = P(A)$$

### 6. Process bucket C:

$$\lambda_C(a) = \sum_c P(c|a) \lambda_B(c, a)$$

Remaining bucket:

$$\text{bucket}_A = P(A) \lambda_C(A)$$

### 6. Process bucket A:

$$Z = \sum_a P(a) \lambda_C(a)$$

$$P(a|G=1) = \frac{1}{Z} P(a) \lambda_C(a)$$

Example taken from undated presentation notes by Dechter and Lavee

## Modifications

- Most probable explanation (MPE)
  - Elimination operation: replace sum with max
  - Keep track of maximizing value at each stage
  - “Forward” step to determine maximizing value at each stage
- Maximum a Posteriori Hypothesis(MAP)
  - Sum over non-hypothesis variables
  - Max over hypothesis variable(s)
  - Forward phase over hypothesis variables only
- Also can be modified for constraint satisfaction, propositional satisfiability, and maximum expected utility action

Software for bucket elimination (and other algorithms)  
can be found at Rina Dechter's software page  
<https://www.ics.uci.edu/~dechter/software.html>

## Exact Inference with Continuous Variables

- So far we have focused on Bayesian networks with discrete variables
- Graph-based and factorization-based exact algorithms can be modified to handle certain networks with continuous random variables
  - Linear Gaussian networks
  - Conditional linear Gaussian networks
  - Mixtures of conditional linear Gaussian networks
  - Mixtures of truncated exponentials
- Can be extended to “almost exact” inference for networks with discrete children of Gaussian parents (Lerner, et al., 2001)

## Junction Tree Algorithm for CLG Networks

- Junction tree algorithm can be adapted to exact inference for:
  - Continuous graphical models with Gaussian potentials
  - Conditional linear Gaussian (CLG) Bayesian networks
    - » Hybrid discrete / continuous
    - » No continuous RV can have a discrete child
    - » Continuous RVs are Gaussian with mean a linear function of continuous parents given discrete parents
  - CLG mixtures
    - » Continuous nodes either linear Gaussian or a mixture of linear Gaussians
- CLG junction tree algorithm propagates
  - Means and covariances for Gaussian networks
  - Mixtures of means and covariances for mixtures
- Inference is tractable for graphical models for which the treewidth (size of largest clique) is not too large
  - Even simple networks can lead to intractably large cliques
  - Approximate inference must be used when cliques are too large
- JT for CLG networks is available in MATLAB BNT package
  - BNT is widely used, available on GitHub, but no longer under active development

## Unit 6 Outline

- Exact Algorithms
-  • Monte Carlo Approximation Algorithms
- Deterministic Approximation Algorithms
- Inference in Relational Models



## Approximate Inference: Monte Carlo Methods

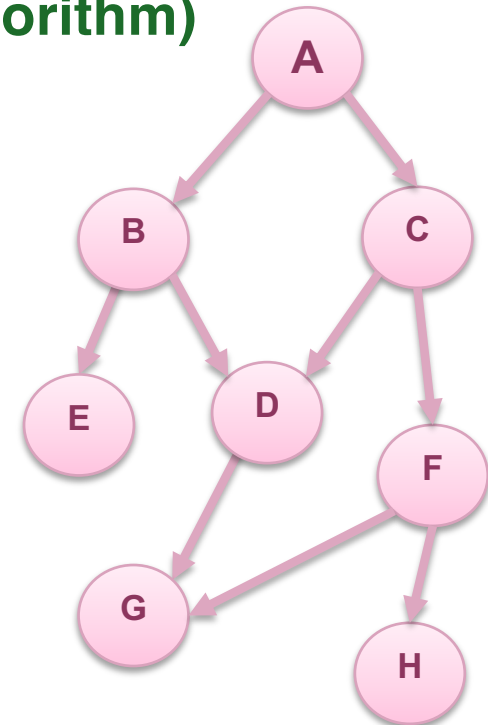
- Objective: Compute a computationally challenging integral or sum
- Assumptions:
  - We can represent the sum or integral as the expected value of a random variable
  - The random variable is cheap to simulate relative to computing the sum or integral
- The procedure:
  - Simulate many realizations of the random variable and compute the average
- The theory:
  - As the number of samples becomes large:
    - » The sample average converges to the sum or integral we want to compute
    - » The variance of the estimate becomes small
  - There are intelligent ways to draw the sample and form the estimate to keep the variance low

## Monte Carlo Methods for Inference in Graphical Models

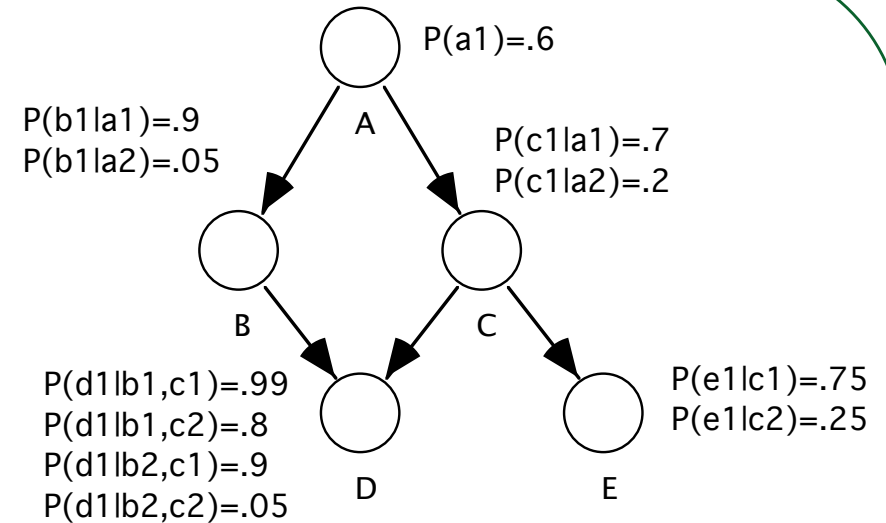
- Forward sampling (works only on Bayesian networks)
  - Logic sampling
  - Likelihood weighting
  - Particle filtering (a variant of likelihood weighting)
- Markov blanket sampling (works on directed or undirected graphical models)
  - Gibbs sampling
  - Metropolis-Hastings sampling

## Logic Sampling (The Simplest Forward Sampling Algorithm)

- Logic sampling takes random draws from the joint distribution of all variables in the network
- Each node keeps a running count of how many times each of its states is drawn
- To take one random draw (one observation on all variables):
  - Sample the nodes in order (so parents of a node are sampled before the node)
  - For root node(s), select state with probability  $P(\text{state})$
  - For non-root node(s) select state with probability  $P(\text{state}|\text{sampled states of parents})$
  - After all nodes have been sampled
    - » Check whether sampled values of evidence values match observed states
    - » If they don't match, throw away the draw
    - » If they match, increment the count for sampled state of each non-evidence node
- Estimate  $P(X_t|X_e)$  by observed frequency among the retained samples



## Example of Logic Sampling



- Goal: find  $P(B|D=d2, E=e2)$
- Steps in drawing one sample:
  - Select  $A=a1$  with probability 0.6 and  $a2$  with probability .4
  - Select value of B with probability  $P(B|A)$ 
    - » If  $A=a1$  select  $B=b1$  with probability 0.9 &  $B=b2$  with probability 0.1
    - » If  $A=a2$  select  $B=b1$  with probability 0.05 &  $B=b2$  with probability 0.95
  - Select value of C with probability  $P(C|A)$ 
    - » If  $A=a1$  select  $C=c1$  with probability .7 &  $C=c2$  with probability .3
    - » If  $A=a2$  select  $C=c1$  with probability .2 &  $C=c2$  with probability .8
  - Select value of D with probability  $P(D | B, C)$ 
    - » If  $D=d1$  throw sample out and start over
    - » If  $D=d2$  continue
  - Select value of E with probability based  $P(E | C)$ 
    - » If  $E=e1$  throw sample out and start over
    - » If  $E=e2$  continue
  - Increment counts for observed states of A, B, and C
- Estimate  $P(B|D=d2, E=e2)$  by the frequency of times each value of B occurs in the retained sample

## Theory Behind Logic Sampling

- On each sample draw,
  - We get  $(A,B,C,D,E) = (a,b,c,d,e)$  with probability  $P(a)P(b|a)P(c|a)P(d|b,c)P(e|c)$ , which is equal to  $P(a,b,c,d,e)$
  - We get  $(B,D,E) = (b,d,e)$  with probability  $P(b,d,e)$
  - We get  $(D,E) = (d,e)$  with probability  $P(d,e)$
  - We keep the sample with probability  $P(d_2,e_2)$
  - We increment the count for state  $b$  of  $B=b$  with probability  $P(b,d_2,e_2)$
  - Conditional on keeping the sample we increment the count for state  $B$  with probability  $P(b|d_2,e_2)$
- For samples we keep, the observed value of  $B$  is a random variable with possible values  $b_1$  and  $b_2$ , with probabilities  $P(b_i|d_2,e_2)$
- The estimate is:  $\hat{p}_b = \frac{\text{count for state } b}{\text{total count for all states}}$ 
  - Expected value:  $P(b|d_2,e_2)$
  - Variance:  $\frac{1}{\text{\#samples kept}} P(b|d_2,e_2)(1 - P(b|d_2,e_2))$
- We keep sampling until we have kept enough observations that the variance of the estimate is acceptably low
- If  $P(b|d_2,e_2) \neq 0$ , then the estimate satisfies the Central Limit Theorem. As count gets large, estimate will be approximately normally distributed with mean and variance as given above

## Aside: Random Numbers

- Monte Carlo algorithms depend on sequences of *random numbers*
- These sequences can be produced by:
  - Hardware random number generator: uses a physical device to produce sequences of numbers that are random according to physical theory (e.g., thermal noise or quantum phenomena)
  - Pseudo-random number generator: uses an algorithm to produce a deterministic sequence of numbers that passes statistical randomness tests
    - » The sequence is determined by an initial value called the *seed*
  - Combination: use physical device to set the seed for a pseudo-random number generator with a faster data rate
    - » "Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin." – John von Neumann
- Any pseudo-random number generator is ultimately periodic and can produce artifacts such as autocorrelation and values that fall into a low-dimensional subspace. These artifacts can provide poor performance on Monte Carlo simulations
- Reproducibility of pseudo-random sequences is useful for debugging
  - Results of a Monte Carlo algorithm will be the same each time if the seed is set to a given value at initiation of the algorithm

## Modification: Likelihood Weighting

- Logic sampling can be very inefficient due to discarded observations
  - In our example,  $P(d2, e2) = .26$
  - This means we are throwing out 74% of the observations!
  - For some problems the vast majority of observations are discarded
- Logic sampling can be modified to avoid throwing out observations
  - Suppose we had sampled  $(a2, b2, c1)$
  - Then  $P(d2|a2, b2, c1) = .1$  and  $P(e2|a2, b2, c1) = .25$
  - To sample D:
    - » If we sampled many times, 90% of the time we would throw out observations  $(a2, b2, c1)$
    - » In other words, 10% of the time we would increment the counts for  $a2$ ,  $b2$ , and  $c1$
    - » Instead of throwing out the sample, we assign a weight of .1 to the sampled value due to D
  - To sample E:
    - » The weight due to E for this observation is .25 (probability of getting  $e2$  with logic sampling)
  - Multiply the weights together to get a weight for the observation of .025
  - Instead of throwing out the observation, we increment the counts for  $a2$ ,  $b2$  and  $c1$  by the observation weight (0.025 in this case)
- Observation weights in our example:
  - If  $B=b1$  and  $C=c1$ , observation weight is  $.0025 = \Pr(d2|b1, c1)\Pr(e2|c1)$
  - If  $B=b1$  and  $C=c2$ , observation weight is  $.15 = \Pr(d2|b1, c2)\Pr(e2|c2)$
  - If  $B=b2$  and  $C=c1$ , observation weight is .025
  - If  $B=b2$  and  $C=c2$ , observation weight is .7125

## The Likelihood Weighting Estimate

- Likelihood weighting estimate of  $P(b|d_2, e_2)$  is given by:

$$\tilde{p}_b = \frac{\sum \text{weights for } b}{\sum \text{all weights}} = \frac{\sum \text{weights for } b}{\sum \text{weights for } b_1 + \sum \text{weights for } b_2}$$

- If  $P(b|d_2, e_2) \neq 0$ , then the likelihood weighting estimate also has a limiting normal distribution



## Comparison: LW and Logic Sampling

- General rule in Monte Carlo: replacing sampling by exact computation reduces variance
- Both logic sampling and likelihood weighting can be viewed as following these steps:
  - Sample A, B, C according to  $P(A,B,C)$
  - Fix  $D=d_2$ ,  $E=e_2$
  - Weight the observation by a "sampling weight" that depends on B and C
    - » In likelihood weighting the weight is  $P(d_2,e_2|B,C)$
    - » In logic sampling the weight is drawn randomly. It is equal to 1 with probability  $P(d_2,e_2|B,C)$  and 0 with probability  $1-P(d_2,e_2|B,C)$  [kept observations have weight 1, thrown out observations have weight 0]
- In both algorithms:
  - Given B,C the sampling weights have expected value  $P(d_2,e_2|B,C)$
  - Variance of sampling weight is smaller for likelihood weighting:
$$V(W) = E[V[W | B,C]] + V[E[W | B,C]]$$
  - First term in this sum is zero for likelihood weighting, nonzero for logic sampling. Second term is same for both algorithms
  - This is an example of a "Rao-Blackwell" method
    - » Rao-Blackwell Theorem: variance of estimator is reduced by replacing a sampling step by a deterministic step that preserves expected value

## Importance Sampling

- If the observed evidence was a priori very unlikely then
  - Most weights will be very low
  - Logic sampling and likelihood weighting can be very inefficient
- Key idea behind importance sampling is to try to focus sampling on probable cases
  - Draw observations not from  $P(X)$  but from another distribution  $Q(X)$  called an importance sampling distribution.
  - Weight the observations to compensate for drawing from a different distribution from the one we want to estimate
- Example: estimate  $P(B=b1|D=d2,E=e2)$  in the Bayesian network of the earlier example
  - Draw observations  $(A,B,C)$  from the distribution  $Q$
  - Let  $Z = \begin{cases} \frac{P(A,b1,C,d2,e2)}{Q(A,b1,C)} & \text{if } B = b1 \\ 0 & \text{if } B \neq b1 \end{cases}$  and  $W = \frac{P(A,B,C,d2,e2)}{Q(A,B,C)}$
- Draw  $N$  samples and calculate  $Z_i$  and  $W_i$ ,  $i=1,\dots,n$
- Estimate  $P(B=b1|D=d2,E=e2)$  as weight for  $b1$  / total weight:

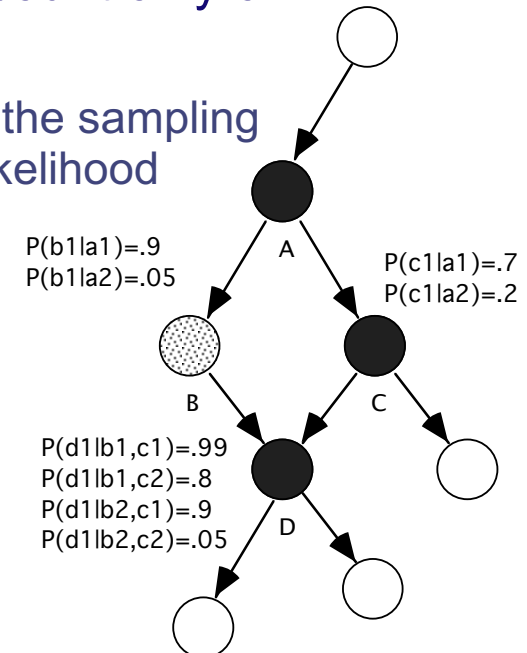
$$\hat{p}_{b1} = \frac{\sum_{i=1}^N Z_i}{\sum_{i=1}^N W_i}$$

## Why Importance Sampling Works

- $E[W] = \sum_{a,b,c} \frac{P(a,b,c,d2,e2)}{Q(a,b,c)} Q(a,b,c) = P(d2,e2)$
- $E[Z] = \sum_{a,c} \left( \frac{P(a,b1,c,d2,e2)}{Q(a,b1,c)} Q(a,b1,c) + 0 \cdot Q(a,b2,c) \right) = P(b1,d2,e2)$
- By the Law of Large Numbers,  $\frac{\sum_{i=1}^N Z_i}{\sum_{i=1}^N W_i} \rightarrow P(b1 | d2,e2)$
- If  $P(b1|d2,e2), P(b2|d2,e2) \neq 0$  then  $(\sum Z_i, \sum W_i)$  satisfies a central limit theorem.
  - Variance will be smallest when distribution  $Q$  is near  $P(A,B,C|d2,e2)$
  - Variance may be very high if  $Q$  places too small a probability on values of  $A,B,C$  relative to  $P(A,B,C|d2,e2)$
- A good importance distribution is crucial
- Adaptive importance sampling
  - Adapt importance distribution as sampling progresses
  - Has shown good results in experimental tests

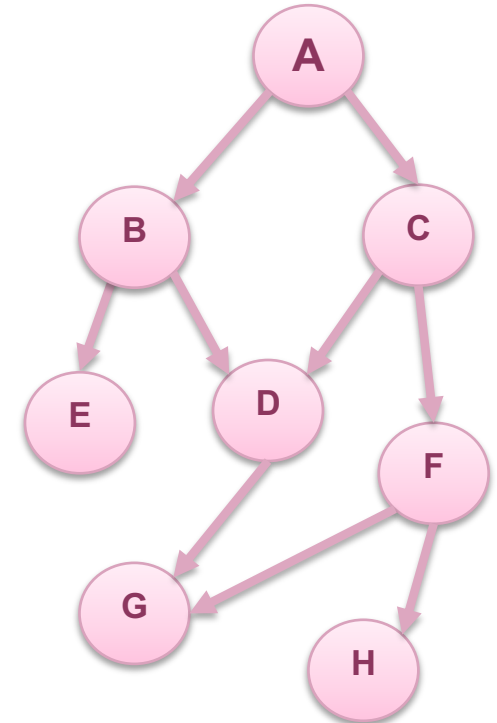
# Markov Blanket Scoring

- Logic sampling and likelihood weighting increment the count only of the sampled node
  - If we sample  $B=b2$  then we increment the count for  $b2$  by the sampling weight (1 for logic sampling and  $P(\text{evidence}|\text{sample})$  for likelihood weighting)
- Markov blanket scoring (see Pearl, 1987):
  - Compute scores
    - »  $s_{bi} = P(b_i | \text{sampled values of Markov blanket})$
  - Increment the count for each value  $b_i$  of  $B$  by its score
  - Another Rao-Blackwell method
- Markov blanket scoring is another example of replacing a sample by an expectation
- Shachter and Peot (1989) performed an experimental comparison of algorithms on some hard networks. Their results indicated that Markov blanket scoring was worth the computational overhead in computing the scores. This is an empirical evaluation and not a theoretical proof of greater efficiency.



## Summary: Forward Sampling

- Forward sampling draws samples in the order of arcs in a directed graphical model
  - Parents of  $X$  are always sampled before  $X$
- Logic sampling is the simplest forward sampling algorithm
  - $X$  is sampled with  $P(X|pa(X))$
  - Samples are discarded if inconsistent with evidence
  - Samples from correct distribution but inefficient
- Many modifications have been proposed, e.g.
  - Likelihood weighting
  - Importance sampling / adaptive importance sampling
  - Markov blanket scoring
- The R package `bnlearn` does inference using likelihood weighting
  - For continuous hybrid networks, `bnlearn` finds only MAP value, even for discrete nodes



## Dealing with Near-Deterministic Relationships

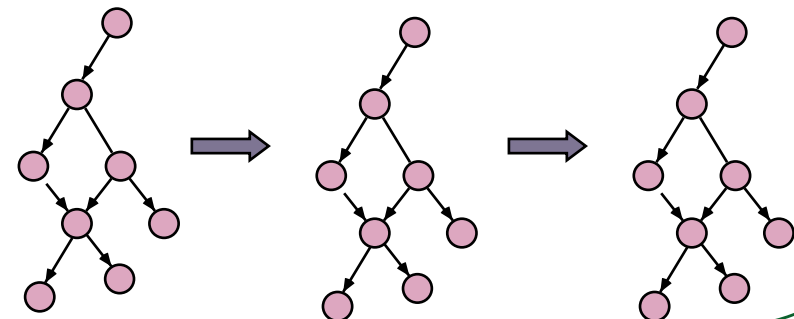
- Likelihood weighting and importance sampling can give very poor results when there are deterministic relationships
  - Many zero-weight samples are rejected
  - Sampling is highly inefficient
- Many applications involve mixed deterministic and probabilistic relationships
- SampleSearch (Gogate and Dechter, 2011) augments sampling with systematic constraint-based backtracking search
  - Combining sampling with search results in bias
  - SampleSearch adjusts for the bias using weights
    - » Exactly or approximately (when exact weights are too complex to compute)
  - Experimental results demonstrate performance improvement in models with significant amount of determinism

## Markov Blanket Sampling

- In forward sampling, variables are sampled in the order imposed by the directed graph
  - Requires a directed graphical model
  - Each random variable is sampled with a distribution depending on values of its parents
  - Consecutive realizations of the network are independent and identically distributed (except for weak dependence with adaptive importance sampling)
- Markov blanket sampling samples random variables in arbitrary order
  - Applies to directed or undirected graph
  - Each random variable is sampled with a distribution depending on values of its Markov blanket
  - Consecutive realizations of the network are dependent
- Markov blanket sampling is a kind of Markov chain Monte Carlo (MCMC) method

# Markov Chain Monte Carlo

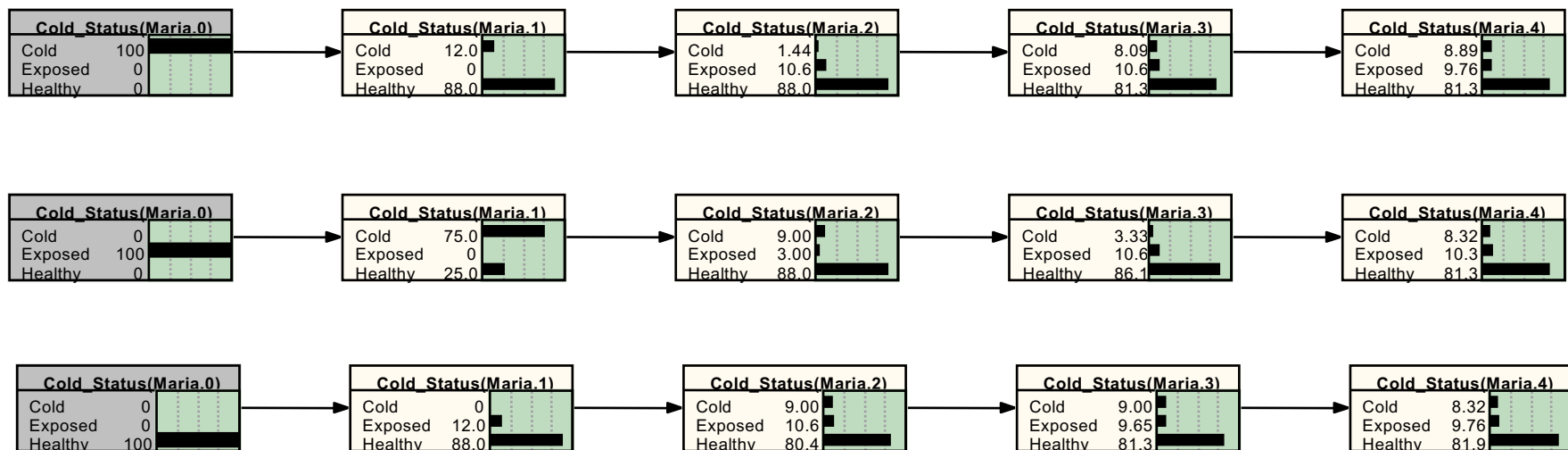
- MCMC originated in statistical physics and has become popular in statistics
- A MCMC algorithm draws samples from a Markov Chain designed to have the target distribution as its stationary distribution
  - A Markov chain is a sequence  $X_1, X_2, \dots$  of random variables with the “memoryless property”
    - »  $\Pr(X_i | X_1, \dots, X_{i-1}) = \Pr(X_i | X_{i-1})$  order 1 Markov chain
  - A stationary distribution  $Q$  is one that persists once it is established: if  $X_i$  has distribution  $Q$ , so does  $X_{i+1}$
- How MCMC works
  - Design a Markov chain having  $P_{target}$  as its stationary distribution
  - Draw a large number of samples from the Markov chain
  - Estimate  $P_{target}$  by the sample frequency
    - » Discarding initial “burn in” period
    - » Optionally “thin” sample to reduce autocorrelation
- For inference in graphical models the Markov chain state is the vector of realizations of all random variables





## Review: Markov Chain

- Example: model evolution of colds with a Markov chain
  - States: Cold, Exposed, Healthy
  - Allowable transitions:
    - » Cold → Healthy
    - » Healthy → Exposed
    - » Exposed → Cold
- This Markov chain has a unique stationary distribution
  - If a Markov chain has a finite state space and all states are reachable from any state, then it has a unique stationary distribution and all initial distributions evolve to this stationary distribution

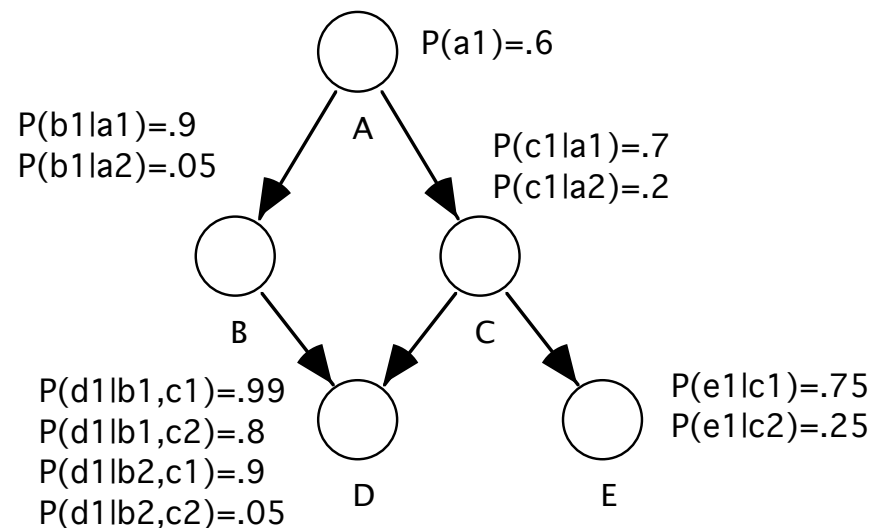


## The Simplest MCMC Algorithm: Gibbs Sampling

- Begin with an initial assignment of values to nodes
  - Evidence nodes  $X_{ev}$  are set to their observed values
  - Other nodes can be set to any value
- Go through the non-evidence nodes one at a time, following these steps:
  - Sample the node with probability:
$$P(\text{node}|\text{rest of nodes}) = P(\text{node}|\text{Markov blanket})$$
  - Replace node's state with newly sampled state
- Repeat until a large sample of observations is obtained
- Estimate  $P(\text{node}=\text{value})$  by sample frequency
  - Usually we discard samples from a “burn in” period before collecting statistics for computing sample frequency
  - To adjust for correlation of consecutive samples we often only count every  $k^{\text{th}}$  observation, where  $k$  is called the *thinning interval*

## Example of Gibbs Sampling

- Goal is to find  $P(B|D=d2,E=e2)$
- To sample node A:
  - Markov blanket is B,C
  - $P(A|B,C) \propto P(A) P(B|A) P(C|A)$
  - Example probabilities for configurations of (B,C):
    - »  $P(a1|b1,c1) = .6 \times .9 \times .7 / (.6 \times .9 \times .7 + .4 \times .05 \times .2) = .99$
    - »  $P(a1|b2,c1) = .6 \times .1 \times .7 / (.6 \times .1 \times .7 + .4 \times .95 \times .2) = .36$
- To sample node B:
  - Markov blanket is A,C,D
  - $P(B|A,C,D) \propto P(B|A)P(D|B,C)$
  - Example configuration for (A,C,D) = (a1,c2,d2):
    - »  $P(b1|a1,c2,d2) = .9 \times .2 / (.9 \times .2 + .95 \times .95) = .166$
- To sample node C:
  - Markov blanket is A,B,D,E
  - $P(C|A,B,D,E) \propto P(C|A)P(D|B,C)P(E|C)$
- D and E are not sampled



- Sample values  $(a_i, b_i, c_i)$  form a Markov chain
- The probability distribution for  $(a_1, b_1, c_1)$  is independent of past samples given the immediately preceding sample  $(a_{i-1}, b_{i-1}, c_{i-1})$
- The unique stationary distribution for this Markov chain is  $P(A,B,C|D=d2,E=e2)$

## Comments on Gibbs Sampling

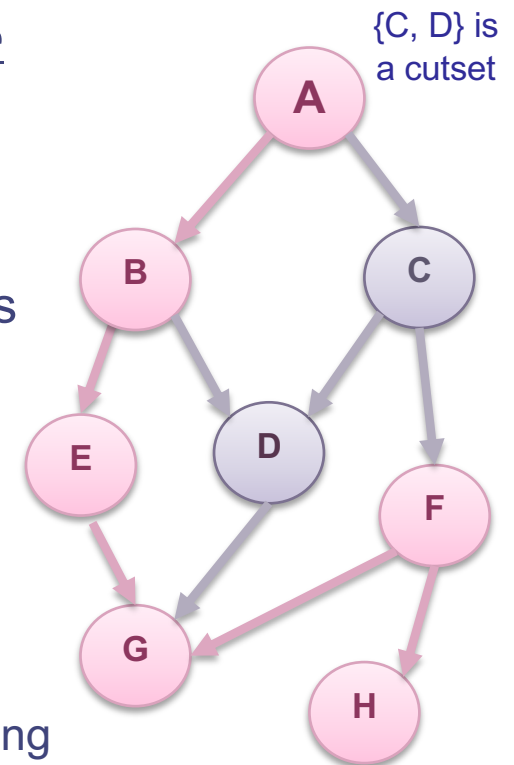
- Successive states of network form a Markov chain
  - Each time through the cycle gives a new set of values for all nodes in network
  - Probability distribution for new set of values depends only on most recent set of values, not on rest of past samples
- Joint distribution  $P(X|X_{ev})$  is a stationary distribution of this chain
  - Suppose distribution of Markov blanket of  $X$  is  $P(\text{Markov blanket of } X | X_{ev})$
  - Now sample  $X$  according to  $P(X|\text{Markov blanket of } X)$
  - Joint distribution of  $(X, \text{Markov blanket of } X)$  is
$$P(X|\text{Markov blanket of } X) P(\text{Markov blanket of } X | X_{ev})$$
- If all distributions are strictly positive  $P(X|X_{ev})$  is a unique stationary distribution and the chain converges to it
- Practical issues
  - Typically we discard “burn-in” period, thin the chain to reduce correlation between consecutive estimates, and use convergence diagnostics to decide when we have enough observations
  - When some probabilities are near zero the chain can get “stuck” in certain regions for a long time
  - Convergence diagnostics can be misleading when chain gets stuck (“poor mixing”)

## Soft Evidence and Gibbs Sampling

- Hard evidence is incorporated by fixing evidence variable at observed value
- If variable  $X$  has soft evidence  $\lambda(X)$  then sampling distribution of  $X$  | Markov blanket of  $X$  is proportional to the product of  $\lambda(X)$  and all belief tables that mention  $X$

# Cutset Conditioning & Cutset Sampling

- Loop cutset (Becker and Geiger):
  - A *sink* for a loop is a vertex with two edges on the loop directed into it
  - A vertex is *allowed* for a loop if it is not a sink for that loop
  - A *loop cutset* is a set of vertices that has at least one allowed vertex for each loop
- Cutset Conditioning (Pearl)
  - Calculate result for all combinations of cutset variables and weight by probability of cutset variables
  - This is an exact algorithm which can be very costly when there are many loops
- Cutset sampling (Bidyuk and Dechter)
  - Find loop cutset
  - Do repeatedly for large number of samples
    - » Sample values for cutset variables using Gibbs sampling
    - » Propagate using algorithm for singly connected networks
  - Has shown good results in experimental tests



## Component-Wise Metropolis-Hastings Sampling

- For many inference problems:
  - $P(X_{\text{new}} \mid \text{Markov blanket of } X)$  is expensive or impossible to sample
  - It is cheap to calculate
$$P(X_{\text{new}} \mid \text{Markov blanket of } X) / P(X_{\text{old}} \mid \text{Markov blanket of } X)$$
- Metropolis-Hastings algorithm works exactly like Gibbs sampling except:
  - We can use any distribution for sampling a proposed value for  $X_{\text{new}}$  given Markov blanket of  $X$
  - To ensure the correct stationary distribution, we add a rule for deciding stochastically whether to accept or reject the proposed value of  $X_{\text{new}}$

## Metropolis-Hastings Steps

- Begin with an initial assignment of values to nodes
  - Evidence nodes  $X_{ev}$  assigned to their observed values
  - Other nodes can have any value
- Go through the non-evidence nodes one at a time, following these steps:
  - Sample new value  $X_{new}$  with  $Q(X_{new} | X_{old}, MB)$  where  $X_{old}$  is the current value of  $X$  and  $MB$  is the current value of the Markov blanket of  $X$
  - Apply the following acceptance rule:
    - » Compute the acceptance probability
$$A(x_{new} | x_{old}, MB) = \min \left\{ 1, \frac{Q(x_{old} | x_{new}, MB)P(x_{new} | MB)}{Q(x_{new} | x_{old}, MB)P(x_{old} | MB)} \right\}$$
    - » Accept  $x_{new}$  as the new state for  $X$  with probability  $A(x_{new}|x_{old}, MB)$
    - » Reject  $x_{new}$  and keep  $x_{old}$  as the state of  $X$  with probability  $1 - A(x_{new}|x_{old}, MB)$
- Repeat until a large sample of observations is obtained
- Estimate  $P(\text{node}=\text{value})$  by sample frequency



## Some Comments

- The Metropolis-Hastings algorithm has the same stationary distribution as Gibbs sampling.
- Acceptance probability is higher when:
  - Posterior probability of new state is higher
  - Transition back to old state is more probable
- Under regularity conditions (e.g., sampling distribution  $Q(\cdot)$  and  $A(\cdot)$  do not change as sampling progresses and all values of each  $X$  are sampled with positive probability for all values of MB) the stationary distribution is unique and satisfies a central limit theorem
- Metropolis-Hastings sampling = Gibbs sampling when
$$Q(X_{\text{new}} | X_{\text{old}}, \text{MB}_{\text{old}}) = P(X_{\text{new}} | \text{Markov blanket of } X)$$
- Both Gibbs and component-wise MH suffer from the problem of getting stuck in local basins when exiting the basin requires multiple simultaneous changes

## Comparison: Markov Blanket and Forward Sampling

- Forward sampling with importance weights
  - Sample unobserved nodes in direction of node ordering ( $X_i$  is sampled after  $X_{pa(i)}$ )
  - Sampled cases  $\underline{X}_n$  are independent draws from sampling distribution
  - Achieve correct expectation by giving each draw an importance weight proportional to ratio of target probability and sampling likelihood
- Markov blanket sampling
  - Begin by assigning arbitrary values to the nodes
  - Visit nodes in an arbitrary order
    - » Propose new value for node using distribution that depends on current value of node's Markov blanket
    - » Accept new state with probability depending on relative probabilities of new and old states, and relative proposal likelihoods of new and old states
  - Sampling distribution is a Markov chain with unique stationary distribution equal to target distribution

## Comments on Markov Blanket Sampling

- Markov blanket sampling is popular because:
  - It is general-purpose
  - It is easy to implement
  - It works for both directed and undirected graphs
- BUGS (Bayesian inference Using Gibbs Sampling) is a free software package for statistical modeling
  - Modeling language allows specification of complex graphical probability models
  - Approximate inference uses Gibbs sampling
- Markov blanket sampling can be very slow to converge
  - Can remain stuck for long periods in local basin of attraction (this is called “poor mixing”)
  - Difficult to escape when multiple simultaneous changes are needed
  - May be very difficult to diagnose when sampler mixes poorly

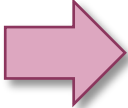
## Summary of Key Ideas in Monte Carlo Sampling

- Rejection sampling
  - Sample observations from a distribution
  - Reject observations according to criterion
  - Estimate target quantity from remaining observations
  - Resulting estimate approximates target quantity
  - Examples: Logic sampling and Metropolis-Hastings
- Importance sampling
  - Sample observations from a distribution
  - Compute sampling weight
  - Estimate target quantity as ratio of weighted sums
  - Resulting estimate approximates target quantity
  - Likelihood weighting can be viewed as importance sampling with the importance distribution equal to the prior distribution
- “Rao Blackwellization”
  - Replace the sampled quantity with its expectation when the expectation can be computed cheaply
  - Examples: likelihood weighting and Markov blanket scoring and cutset sampling
- Markov Chain sampling
  - Sometimes a distribution is difficult to compute or sample directly but is the stationary distribution of a Markov chain that is cheap to sample
  - Sampling from the Markov chain approximates target distribution
  - Examples: Gibbs sampling and Metropolis-Hastings sampling

## Some General Comments

- Markov Chain Monte Carlo (MCMC)
  - Construct Markov chain with stationary distribution equal to distribution being estimated
  - If we run  $N$  parallel chains, discard first  $B$  observations of each chain (burn-in sample), we can treat  $B+1^{\text{st}}$  observation from each chain as a sample of  $N$  independent observations from target distribution
  - Correlations between draws of Markov chain induces inefficiency
- Independent sampling with importance weights
  - Examples are logic sampling, adaptive importance sampling; SIR (sampling / importance resampling)
  - Draw independent samples with distribution that approximates target distribution
  - Use importance weights to achieve correct expected values
  - Skewed weights induce inefficiency

## Unit 6 Outline

- Exact Factorization Based Algorithms
- Monte Carlo Approximation Algorithms
-  • Deterministic Approximation Algorithms
- Inference Inference in Relational Models

## Loopy Belief Propagation

- Apply method designed for singly connected networks in a multiply connected network
  - Works by “peer pressure” – node’s neighbors tell it how to update its beliefs
  - Evidence enters at observed nodes and propagates via links in network
- May not converge
  - May oscillate indefinitely between belief states
  - Precise conditions for convergence are not well understood
  - Sufficient conditions for convergence have been identified
- Not guaranteed to be accurate even if it converges
- Works surprisingly well on some hard problems
- Sometimes used as starting point for approximation algorithms
- Iterative Join-Graph Propagation (IJGP) generalizes loopy BP to a *join graph* (like a junction tree but with loops)

## Combining Sampling with Deterministic Approximation

- There are algorithms that combine deterministic approximation with sampling
- IJGPSampleSearch is one such algorithm
  - SampleSearch combines importance sampling with search in networks that have deterministic links
    - » These networks are very hard for Monte Carlo algorithms
    - » SampleSearch combines uses constraint-based backtracking search to find and avoid sampling zero-probability paths
    - » Weights compensate for the bias introduced by avoiding zero-probability paths
  - Iterative Join-Graph Propagation (IJGP)
    - » Replaces junction (join) tree with join graph
    - » Generalization of loopy BP
    - » Anytime algorithm



## Variational Approximations

- Basic idea:
  - Transform intractable network by removing some links to leave "tractable backbone"
  - Adjust parameters in distributions in "tractable backbone" to approximate desired distribution
  - Common approximation approach: minimize information theoretic measure of distance
- Important area of research: Upper and lower bounds on approximation error
  - Results exist for special case network and local distribution structures

## Mean Field Approximation

- Remove all links
- Assign each node an initial probability
  - Evidence nodes have probability equal to 1 on observed value
  - Others have arbitrary probability
- Re-estimate each probability as:

$$P_{est}(X) = \sum_{ConfigMarkovBlanket} P(X \mid MarkovBlanket) P_{est}(MarkovBlanket)$$

- Iterate until convergence
- This estimate minimizes an information theoretic measure of distance
- Converges to local maximum of likelihood constrained to tractable structures
  - Can use restarts from randomly chosen initial probabilities to find better local optimum

## Unit 6 Outline

- Exact Factorization Based Algorithms
- Monte Carlo Approximation Algorithms
- Deterministic Approximation Algorithms
-  • Inference in Relational Models

## Learning as Inference in a Relational Model

- We have shown how parameter learning can be viewed as a problem of inference in graphical models
  - Plate model containing repeated structure is a simple kind of relational model
- Structure learning treats the arcs and/or within-node structural assumptions as uncertain
- Markov chain Monte Carlo methods can be applied to both parameter and structure learning
  - BUGS uses Gibbs sampling for parameter learning in graphical models
  - Reversible-jump MCMC allows sampling from spaces of different dimensionality and can be applied to structure learning

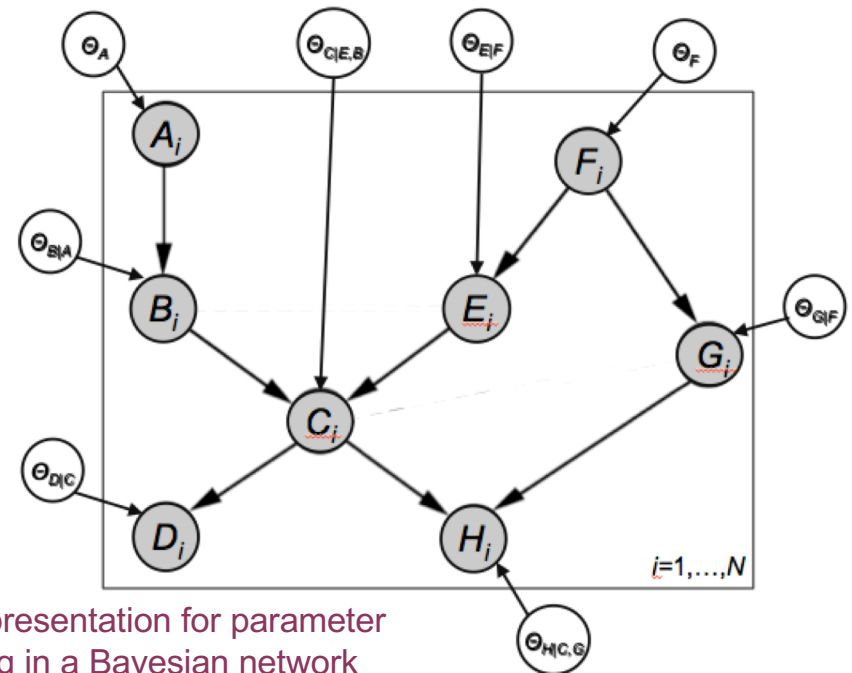
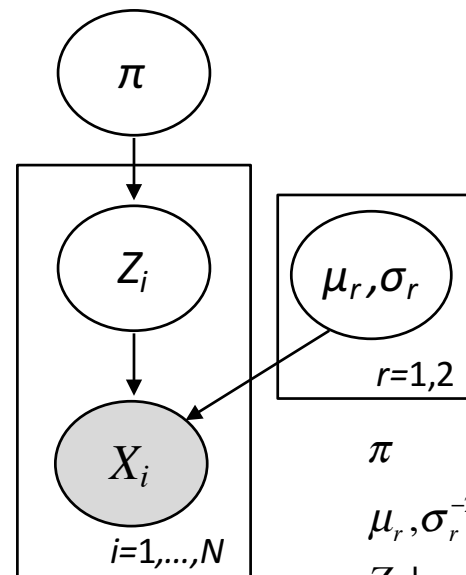
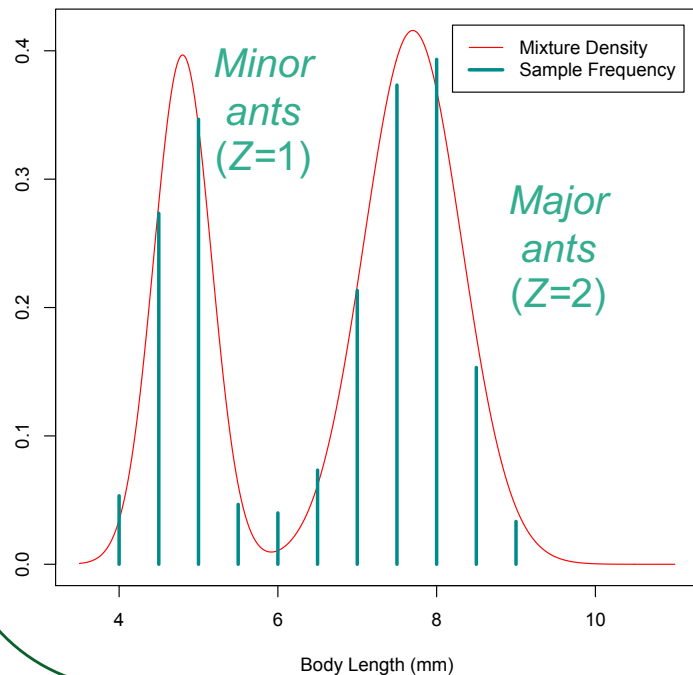


Plate representation for parameter learning in a Bayesian network

# Mixture Model

- A mixture model represents dependence of observations on hidden (latent) variable representing sub-populations
  - Observations are drawn from parametric family
  - Parameter depends on sub-population
- Mixture models can be applied to discover sub-populations and/or classify observations into sub-populations

*Example: Body Length of Weaver Ants (Weber, 1946)*



*Although exact inference for this model is intractable, the parameters can be approximated with MCMC or EM*

$$\begin{aligned} \pi &\sim \text{Beta}(\xi, \zeta) \\ \mu_r, \sigma_r^{-2} &\sim \text{Normal} / \text{Gamma}(m, k, \alpha, \beta) \\ Z_i | \pi &\sim \text{Bernoulli}(\pi) \\ X_i | Z_i, \mu_{Z_i}, \sigma_{Z_i} &\sim \text{Normal}(\mu_{Z_i}, \sigma_{Z_i}) \end{aligned}$$

## Mixed Membership Model

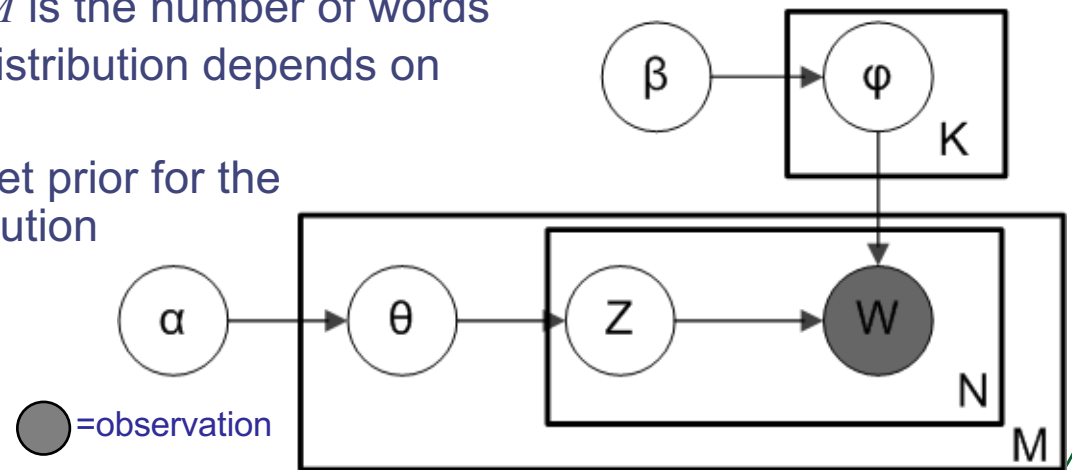
- A *mixed membership model* (aka topic model) is a model in which:
  - Observations belong to groups
  - Each group is modeled as a mixture
  - Mixture components (topics) are shared across all the groups
  - Mixture proportions vary between groups
- This is a powerful modeling technique
  - Groups share information via common set of topics
  - Groups differ in the emphasis placed on the topics
- Learning methods allow us to discover topics and group membership simultaneously
- Many applications, e.g.:
  - Text mining
  - Recommender systems
  - Social networks

Reference:

[http://www.people.fas.harvard.edu/~airoldi/pub/books/b02.AiroldiBleiEroshevaFienberg2014HandbookMMM/Ch1\\_MMM2014.pdf](http://www.people.fas.harvard.edu/~airoldi/pub/books/b02.AiroldiBleiEroshevaFienberg2014HandbookMMM/Ch1_MMM2014.pdf)

## Latent Dirichlet Allocation (LDA) Model

- Popular mixed membership model commonly applied to natural language understanding and text retrieval
  - There are  $M$  documents
  - Each document has  $N$  words
  - The  $n^{\text{th}}$  word in the  $m^{\text{th}}$  document is denoted by  $W_{mn}$
  - Each word  $W_{mn}$  has an associated “topic”  $Z_{mn}$
  - The topics  $Z_{mn}$  are independent draws from a  $K$ -dimensional multinomial distribution, where  $K$  is the number of topics.
  - The parameter  $\theta_m$  of the topic distribution depends on the document.
  - $\theta_m$  is drawn from a Dirichlet distribution with parameter  $\alpha$
  - The words  $W_{mn}$  are independent draws from a  $M$ -dimensional multinomial distribution, where  $M$  is the number of words
  - The parameter  $\varphi_k$  of the word distribution depends on the topic  $k$
  - $\beta$  is the parameter of the Dirichlet prior for the parameter  $\varphi_k$  of the word distribution
- The words are observed; the topics are discovered from the document corpus.



Source: [http://en.wikipedia.org/wiki/Latent\\_Dirichlet\\_allocation](http://en.wikipedia.org/wiki/Latent_Dirichlet_allocation)

## Comments on Mixed Membership Models

- In a standard mixture model each observation would be associated with exactly one of the mixture components
  - E.g., each ant is either a major or a minor ant
- In a mixed membership model like LDA, each document can be associated with multiple components (multiple topics)
  - The topics cluster words that tend to co-occur
  - A document can be “about” more than one topic (i.e., it can contain more than one co-occurring cluster of words)
  - This allows co-occurrence of word patterns to be shared among documents, and also allows for document heterogeneity
- The Dirichlet distribution’s tendency for sparsity tends to keep the number of topics per document small
- Inference problems for mixed membership model
  - Training: learn parameters of mixed membership model from a sample of observations
  - Application: inference on new observations
  - Online learning: refine model as new observations come in



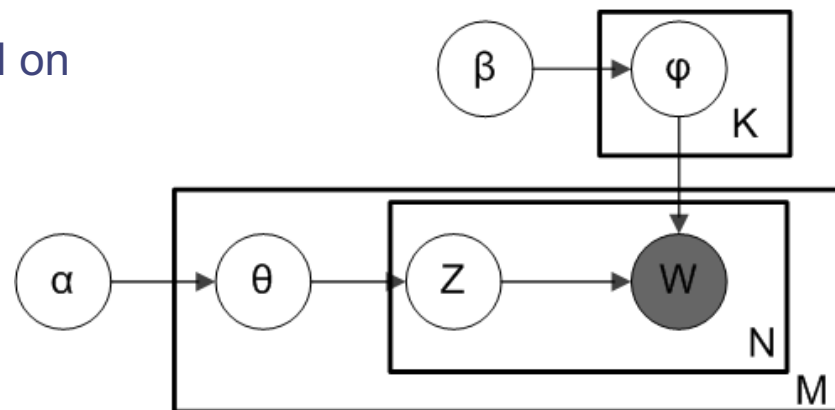
# Enron Email Dataset LDA Topics

source: William Darling, <http://www.uoguelph.ca/~wdarling/tm/enron.html#fn>

commission pge customer cpuc comment decision sce filing parties filed	agreement michelle legal employees letter attached plan document review company	gas capacity pipeline paso natural_gas california social storage market demand	davis power energy stock consultant governor public calpine company july	think thing going problem look put lot believe big job	company firm fund partner ventures technology round capital services investor	ticket houston room number reservation confirmed building hotel street seat	wine date seller received cameron view fax pictures suite nov	var area random shaperect color normal error engine parentrandom diamond	click free offer online price receive link web list save
final schedule hour hourahead found preferred iep sc_id mkt_type trans_date	emission environmental air permit plant facility unit epa water station	team group process review sally operation business global office meeting	going think weekend hope ill night guy thing friday sure	page court employees law labor worker union federal employer act	company stock financial dynegey investor shares billion credit rating analyst	power project india government company dpc indian dabhol electricity mseb	energy company power market trading business natural_gas companies electricity corp	market stock earning report schwab economic index growth analyst nasdaq	power edison utility billion utilities pge california states electricity bankruptcy
friend god gift holiday club love children family food kid	meeting scheduled conference thursday friday monday attend office number tuesday	lynn gas contract nng point month daily meter shipper storage	company skilling lay business stock market profit companies ceo world	contract corp review material offer party andor message affiliates basis	attached report file document comment status draft review version list	bill dwr puc access direct assembly customer committee senate bond	energy california power electricity plan percent davis bush blackout bill	travel hotel roundtrip fares special offer city visit miles deal	database message error received notify prohibited communication immediately delete operation
game yard defense allowed fantasy point passing rank against team	iso market load hour bid nyiso prices power energy pjm	robert michael david mike steve mark scott paul richard james	oil oil_gas industry energy offshore drilling field company production houston	american bush world attack country government president war campaign member	product business management experience company sales marketing services manager team	project request approval resource application construction manager transaction site date	service services fcc bill carrier telecom internet commission local network	texas longhorn team game play brown season top true_orange football	think meeting forward discuss issues group jeff help sure going
student program business school haas university mba berkeley class interview	cost customer risk contract rates pay amount credit period based	updated against play free injury agent season start expected fantasy	user data access password center server problem address sap outage	deal trading risk position transaction book power product credit gas	california ferc committee refund power senate price document generator billion	energy conference program policy event session member research group training	account order buy online free fund investment cash receive stock	paper pulp market houston ena prices mill wind canada story	market ferc california power generation order transmission utilities price price_cap

## Inference in Mixed Membership Models

- Typical mixed membership models use a conjugate prior for the mixture components and the observations
  - If the latent memberships were observed we could do exact inference on the parameters
- Standard inference methods
  - Variational approximation
  - Gibbs sampling
  - Collapsed Gibbs sampling
    - » Conditional on the observations and all other latent membership variables, the distribution for an individual membership variable can be computed exactly
    - » Thus, we can sample  $Z_n$  conditional on  $W_{1:n}$  and  $Z_{-n}$
    - » This “collapsed” sampling helps to prevent the Gibbs sampler from getting stuck in local modes of the posterior distribution



## Lifted Inference

- In Unit 3 we used first-order languages to allow representation of repeated structure
  - We discussed inference by constructing a problem-specific BN called a SSBN or “ground network”
  - The ground network typically has repeated structure
  - Inference on the full ground network can be intractable
- Sometimes we can improve inference performance by exploiting the repeated structure
- “Lifted” inference does this
  - Perform inference directly on the first-order representation
  - Apply results to the ground model
- Performance improvements depend on:
  - Structure of the ground model
  - Specific patterns of evidence

## Summary and Synthesis

- Exact inference
  - There is a duality between factorization approach and graph-based approach
  - Efficiency improvements in one type of algorithm can often be transported to another type of algorithm
  - Hard problems are generally hard for all algorithms
  - Factoring approach seems a more natural way to handle context-specific independence
- Common types of approximation algorithm
  - Stochastic sampling
  - Deterministic approximation
- There are many problem-specific types of approximation algorithm

## References for Unit 6

### Factorization Methods

- Dechter, R. Bucket Elimination: A Unifying Framework For Probabilistic Inference. *Constraints: An International Journal*, No. 2, pp. 51-55, 1997.
- D'Ambrosio, B. Symbolic Probabilistic Inference, *International Journal of Approximate Reasoning*, 1990.

### Continuous and hybrid networks

- Cobb, B., Shenoy, P. and Rumi, R. Approximating Probability Density Functions with Mixtures of Truncated Exponentials. *Proceedings of the Tenth Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2004, pp. 429—436.
- Lerner, U., Segal, E. and Koller, D. U. Lerner and R. Parr. Exact Inference in Networks with Discrete Children of Continuous Parents. In *Proc. Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2001.

### Monte Carlo Methods

- Bidyuk, B. and Dechter, R. Cutset Sampling for Bayesian Networks. In *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, 2006.
- Cheng, J. and Druzdzel, M.. AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks. *Journal of Artificial Intelligence Research (JAIR)*, 13:155-188, 2000.
- Gilks, W. R., Richardson, S. and Spiegelhalter, D. *Markov Chain Monte Carlo in Practice*, Chapman and Hall, 1996.
- Gogate, V. and Dechter, R. SampleSearch: Importance sampling in presence of determinism, *Artificial Intelligence*, Volume 175, Issue 2, 2011, Pages 694-729, ISSN 0004-3702, <http://dx.doi.org/10.1016/j.artint.2010.10.009>
- Pearl, J. Evidential Reasoning using Stochastic Simulation, *Artificial Intelligence* 32, 245-257, 1987.
- Robert, C. and Casella, G., *Monte Carlo Statistical Methods*, Springer-Verlag, 2005.
- R D. Shachter and M. A. Peot. Simulation Approaches to General Probabilistic Inference on Belief Networks, *Proceedings of the Fifth Conference on Uncertainty in Artificial Intelligence*, 1989.
- The BUGS project: <http://www.mrc-bsu.cam.ac.uk/bugs/welcome.shtml>

### Variational Methods

- Tommi S. Jaakkola and Michael I. Jordan. *Computing Upper and Lower Bounds on Likelihoods in Intractable Networks*. ✓

### Loopy belief propagation

- Kevin Murphy, Yair Weiss, and Michael Jordan. Loopy-belief Propagation for Approximate Inference: An Empirical Study *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

### Lifted inference

- R. Braz, E. Amir, and D. Roth, "Lifted First-Order probabilistic inference," in *Introduction to Statistical Relational Learning*. MIT Press, 2007. Available: <http://l2r.cs.uiuc.edu/~danr/Papers/BrazAmRo07.pdf>.