* **Bayesian Network** - data structure that represents dependencies among variables
  + Can represent any full joint probability distribution
  + Directed graph in which each node is annotated with quantitative probability information
    - Each node corresponds to a random variable (discrete or continuous)
    - Directed links or arrows connect pairs of nodes
      * If there’s an arrow from X to Y, X is a parent of Y
      * Graph is directed and acyclic
      * Intuitive meaning of an arrow is X has a direct influence on Y suggesting that causes should be parents of effects
    - Each node Xi has associated probability information that quantifies effect of the parents on the node using finite number of parameters
  + Full joint distribution for all variables is defined by topology and local probability information
  + **Conditional probability table (CPT):** local probability information attached to each node
    - Each row in CPT contains conditional probability of each node value for a conditioning case
    - **Conditioning case**: possible combination of values for the parent nodes
      * Each row must sum to 1
      * For boolean variables the second value is often omitted (since it’s 1 - p)
    - Probabilities summarize potentially infinite set of circumstances
* Bayes nets defines each entry in the joint distribution as
  + *parents(Xi)* denotes values of *Parents(Xi)*
    - Each entry in the joint distribution is represented by the product of appropriate elements of the local conditional distribution in the Bayes net
  + Bayes net can be used to answer any query by summing relevant joint probability values, each calculated by multiplying probabilities from local conditional distributions
  + When one estimates values for local conditional distributions they need to be the actual conditional probabilities for the variable given its parents
* **How to construct a Bayesian network in such a way that resulting joint distribution is a good representation of given domain**
  + Conditional independence relationships can be used to guide construction
  + **Chain rule**: Reducing each joint probability to a conditional probability and a joint probability on a smaller set of variables
  + **Topological ordering**: any order consistent with the directed graph structure
  + Bayesian network is a correct representation of he domain if each node is conditionally independent of its other predecessors in the node ordering given its parents
    - First determine set of variable required to model the domain and order them
      * Any order will work but resulting network is more compact if variables are ordered such that causes precede effects
    - For *i =* 1 to *n*
      * Choose minimal set of parents for *Xi* from *X1* …*Xi - 1* such that is satisfied
      * For each parent insert link from parent to *Xi*
      * Write down CPT
  + Parents of node *Xi* should contain all nodes that directly influence it
    - Since nodes are connected only to earlier nodes, guarantees acyclic graph
    - Bayes nets contain no redundant probability values
      * No redundancy means little chance for inconsistency
      * Impossible to create a bayesian network that violates the axioms of probability
* **Locally structured (sparse) systems** - each subcomponent interacts directly with only a bounded number of other components regardless of the total number of components
  + Allows Bayesian networks by complete while being more compact than the full joint distribution
  + Reasonable to suppose that in most domains each random variable is directly influenced by at most *k* others
    - Assume *n* Boolean variables, need 2*k* numbers to specify each CPT
    - Complete network is 2*k* \* *n*
      * Full joint distribution would need 2*n* numbers
      * E.g. k = 5, n = 30
        + 960 numbers for Bayesian network vs. over a billion
  + Filling out CPTs needs same amount of information as joint distribution
    - So instead w leave out some links even if slight dependency exists
      * Slight gain in accuracy isn’t worth additional network complexity
* Choosing the wrong order can introduce unnecessary network complexity
  + Sticking to a causal model means specifying fewer numbers and the numbers will often be easier to come up with
  + Bad ordering means more probabilities need to be specified
  + However the networks are ordered, they can represent exactly the same joint distribution
* **Conditional independence properties**
  + Variable is conditionally independent of its other predecessors given its parents
  + Each variable is conditionally independent of its non-descendants given its parents
    - Instead of defining full joint distribution as the product of conditional distributions, the network defines a set of conditional independence properties
  + **Markov’s blanket:** variable that is conditionally independent of all other nodes in the network given its markov’s blanket (parents, children, and children’s parents)
  + **D-Separation**
    - Is a set of nodes **X** conditionally independent of another set **Y** given a third set **Z? I.e. Does Z d-separate X and Y?**
      * Consider ancestral subgraph consisting of X, Y, Z, and their ancestors
      * Add links between any unlinked pair of nodes that share a common child for the **moral graph**
      * Replace all directed links by undirected links
      * If Z blocks all paths between X and Y, then Z d-separates X and Y
        + X is conditionally independent of Y given Z
        + Otherwise original Bayes net doesn’t require conditional independence
    - Markov blanket property follows directly from d-separation property since variable’s Markov blanket d-separates it from all other variables
* **Representing Conditional Distributions -** even if *k* is small, it can still be a PITA to fill out all possible conditioning cases. How can we cut it down even further?
  + **Deterministic nodes -** has its values specified exactly by the values of its parents with no uncertainty
  + **Context-specific independence (CSI)** - conditional distribution exhibits CSI if a variable is conditionally independent of some of its parents given certain values of others
    - Bayes net systems often implement CSI using if-then-else syntax
  + **Noisy logical relationships** - uncertain relationships
    - **noisy-OR -** generalization of the logical OR
      * Allows for uncertainty about the ability of each parent to cause the child to be true
      * Causal relationship between parent and child might be inhibited
        + E.g. patient can have cold but not exhibit fever
    - Assumes all possible causes are listed
      * Can always add ‘leak node’ that covers misc.
    - Assumes inhibition of each parent is independent of inhibition of any other parents
    - Noisy logical relationships in which a variable depends on *k* parents can be described using O(*k*) parameters instead of O(2*k*)
* **Bayesian nets with continuous variables**
  + Can be handled using **discretization:** divide possible values into fixed intervals
    - Tradeoff between loss of accuracy and large CPTs
  + Can be handled by defining continuous variable using standard families of probability density functions
    - Gaussian/Normal distribution
  + **Nonparametric representation** - define conditional distribution implicitly with a collection of instances containing specific values of parent and child variables
* **Hybrid Bayesian network** - network with discrete and continuous variables
  + How to specify a hybrid network?
    - Need conditional distribution for a continuous variable given discrete or continuous parents
      * Can specify parameters of the distribution of the continuous variable as a function of its continuous parent
        + **Linear-gaussian conditional distribution:** child has a Gaussian distribution whose mean varies linearly with the value of the parent whose standard deviation is fixed

Network containing only continuous variables with linear-Gaussian distributions has a joint distribution that is a multivariate Gaussian distribution over all the variables

* + - * + When discrete variables are added as parents (not children) of continuous variables, the network defines a **conditional Gaussian (CG) distribution**

**CD distribution**: given any assignment to the discrete variables, the distribution over the continuous variables is a multivariate Gaussian

* + - Need conditional distribution for a discrete variable given continuous parents
      * Conditional distribution is like a soft threshold function
        + **Probit (probability unit) model -** Can use the integral of the standard normal distribution to make soft thresholds

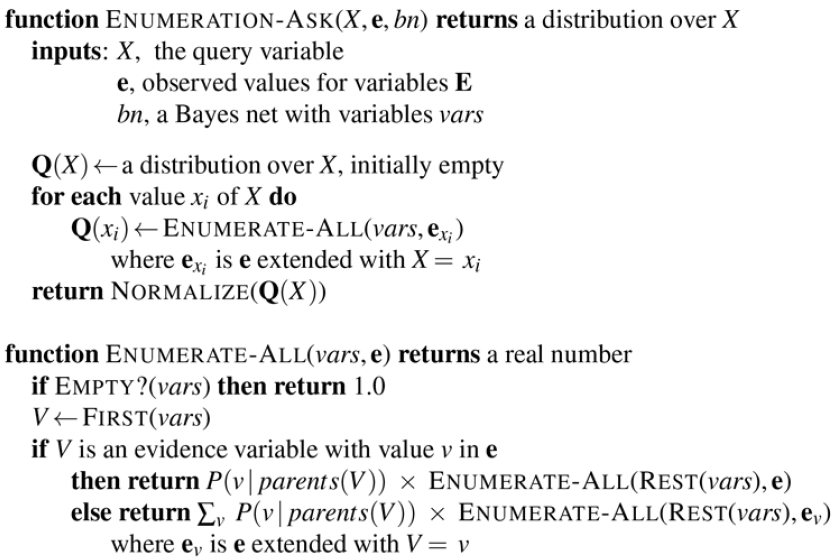
Better fit to real situations

* + - * + **Expit or Inverse logit model** - uses logistic function to produce soft threshold

Maps any *x* to a value between 0 and 1

Sometimes easier to deal with mathematically

* + - * + Both models can be generalized to handle multiple continuous parents by taking a linear combination of parent values
  + Discrete variable with many possible values makes exact inference more expensive **unless the variable’s value is always observed**
    - E.g. make/model of a car has thousands of possible values, but in practice it’s always observed so it doesn’t contribute to inference complexity
* **Hidden variables** - neither input or output variables but essential for structuring the network so that it’s reasonably sparse with a manageable number of parameters
* *A probabilistic inference system’s task is to compute posterior probability distribution for a set of query variables given some observed event (usually some assignment of values to a set of evidence variables)*
  + - * ***X* -** query variable
      * **E** - set of evidence variables E1…Em
      * **e** - particular observed event
      * **Y** - hidden variables Y1…Yl
  + **Inference by enumeration (ENUMERATION-ASK)**
    - Query can be answered using a Bayes net by computing sums of products of conditional probabilities from the network
      * Bayes net gives a complete representation of the full joint distribution
      * The joint distribution can be written as products of conditional probabilities from the network



* + - Doesn’t handle repeated subexpressions
      * Identical values are sometimes computed twice!
  + **Variable elimination algorithm** - eliminating repeated calculations from ENUMERATION-ASK
    - Do the calculation once, and save the results for later use
      * **Variable elimination** - works by evaluating expressions in right-to-left order (bottom up)
        + Intermediate results are stored
        + Summations over each variable are only done for those portions of the expression that depend on the variable
        + Evaluation process sums out variables from pointwise products of factors to produce new factors, eventually yielding a factor that constitutes the solution

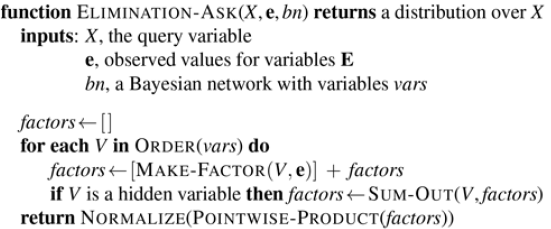
Pointwise product of two factors **f** and **g** yields a new factor **h** whose variables are the union of the variables in **f** and **g** whose elements are given by the product of the corresponding elements in the two factors

Factor resulting from a pointwise product can contain more variables than any of the factors being multiplied

Size of a factor is exponential in the number of variables

Summing out a variable from a product of factors is done by adding up the submatrices formed by fixing the variable to each of its values in turn

*Any factor that doesn’t depend on the variable to be summed out can be moved outside the summation*



* + - * + ORDER function chooses an ordering for the variables

Every choice of ordering yields valid algorithm but different orderings cause different intermediate factors to be generated during the calculation

Intractable to determine optimal ordering

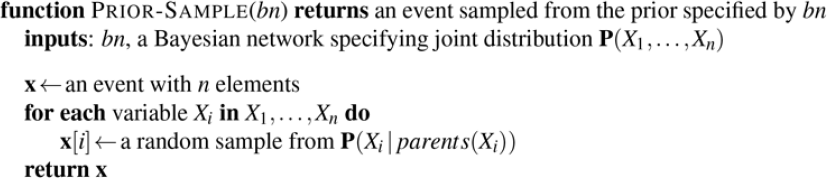
But there are good ordering heuristics

Greedy: eliminate whichever variable minimizes the size of the next factor to be constructed

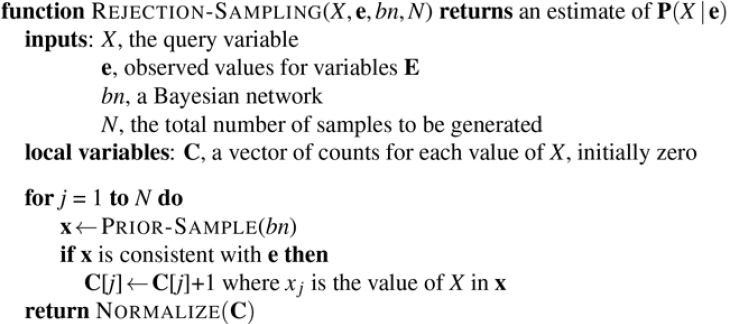
Can remove any leaf node that isn’t a query variable or evidence variable

*Every variable that is not an ancestor of a query variable or evidence variable is irrelevant to the query*

* *Complexity of exact inference in Bayes nets depends strongly on the structure of the network*
  + **Singly connected networks or polytrees:** family of networks in which there is at most one undirected path between any two nodes in the network
    - The time and space complexity of exact inference in polytrees is linear in the size of the network
      * Size is defined as number of CPT entries
      * If number of parents of each node is bounded by constant then complexity will also be linear in number of nodes
        + Holds for any ordering consistent with the topological ordering of the network
  + **Multiply connected networks**
    - Variable elimination can have exponential time and space complexity even when number of parents per node is bounded
  + Because it includes inference in propositional logic as a special case, inference in Bayes nets is NP-hard
    - Proved by reducing satisfiability problems to Bayes net inference
    - Can also reduce Bayes net inference to satisfiability problems
      * Allows us to take advantage of **weighted model counting (WMC)**
* **Clustering algorithms (join tree algorithms)** - join individual nodes of the network to form cluster nodes (**meganodes**) in such a way that the resulting network is a polytree
  + Once in polytree form, special-purpose inference algorithm is required
  + Ordinary inference methods can’t handle meganodes that share variables with each other
* **Approximate Inference for Bayesian Networks**
  + **Monte Carlo** - algorithms that provide approximate answers whose accuracy depends on the number of samples generated
    - Generate random events based on the probabilities in the Bayes net and count up different answers found in those events
    - Can get arbitrarily close to recovering true probability distribution
      * Given Bayes net has no deterministic conditional distributions
  + **Direct sampling methods**
    - Given a source of random numbers *r* uniformly distributed in the range [0,1] can sample any distribution on a single variable
      * Construct cumulative distribution for the variable and return first value whose cumulative probability exceeds *r*
    - Sample each variable in turn in topological order
      * Probability distribution from which value is sampled is conditioned on values already assigned to variable’s parents



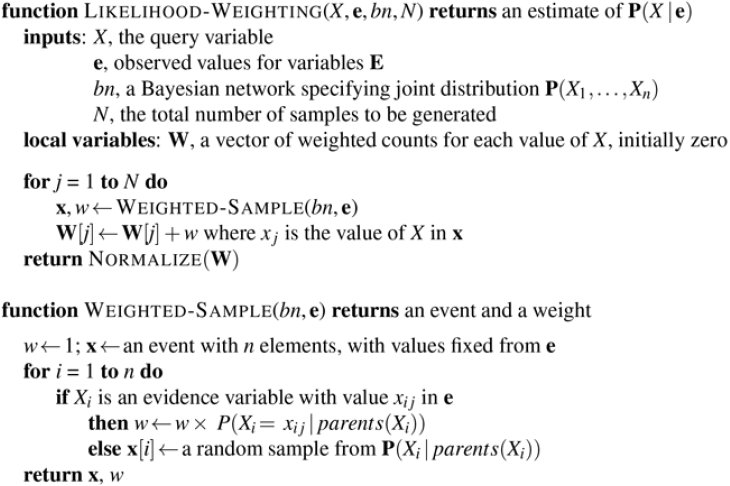
* + - * + Generates samples from prior joint distribution specified by network
    - In any sampling algorithm answers are computed by counting actual samples generated
      * As sample size increases, the estimated probability becomes more exact, it provides **consistent** estimate
    - *The probability of the event can be estimated as the fraction of all complete events generated by the sampling process that match the partially specified event*
    - **Rejection sampling -** general method for producing samples from a hard to sample distribution given an easy to sample distribution
      * First generates samples from the prior distribution specified by the network
      * Then rejects all those that don’t match the evidence
      * Estimate is obtained by counting how often *X* = *x* occurs in the remaining samples



* + - * + Produces a consistent estimate of the true probability

Converges to true distribution as number of samples increase

* + - * + Standard deviation of the error in each probability will be proportional to (*n* = # of samples used to estimate)
      * Complexity of rejection sampling depends on the fraction of samples that are accepted
        + For complex problems convergence can be really slow
        + difficulties with continuous-valued evidence variables
    - **Importance sampling** - aims to emulate the effect of sampling from a distribution *P* using samples from another distribution *Q*
      * Ensure answers are correct in the limit by applying a **correction factor** or **weight** *P*(*x*)/*Q*(*x*), to each sample *x* when counting samples
        + Correction factor compensates for over/under sampling
      * Sample from an easy distribution and apply the necessary corrections
      * Estimate converges to the correct value regardless of which sampling distribution *Q* is used

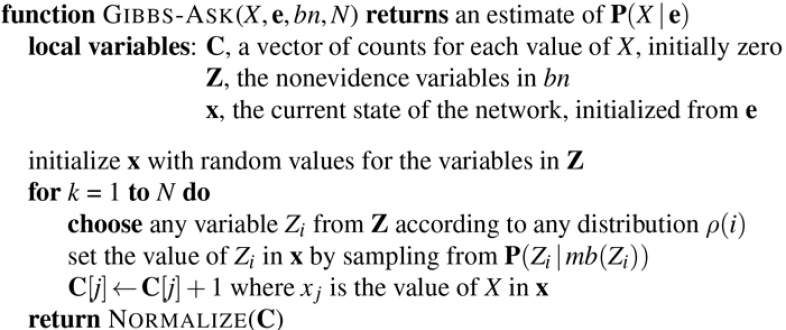


* + - * Weight is the product of the conditional probabilities for the evidence variables given their parents
      * Weight calculation is implemented incrementally in WEIGHTED-SAMPLE
        + Multiply by conditional probability each time evidence variable is encountered
        + Normalization is done before result returned
      * Likelihood weighting uses all samples generated so can be more efficient than rejection sampling
        + Suffers in performance as number of evidence variables increase

Most samples will have very low weights so weighted estimate will be dominated by tiny fraction of samples

Even worse if evidence variables are late in variable ordering

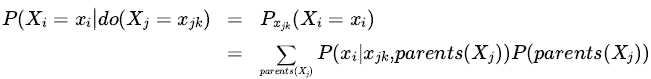
* **Markov chain Monte Carlo (MCMC) -** generate a sample by making a random change to the preceding sample
  + Being in a current state that specifies a value for every variable and generate next state by making random changes to current state
  + **Markov chain** - random process that generates state sequence
  + **Gibbs sampling** - start with an arbitrary state and generate a next state by randomly sampling a value for one of the nonevidence variables ***Xi***
    - ***Xi*** is independent of all other variables given its Markov blanket
    - Gibbs sampling for ***Xi*** means sampling conditioned on the current values of the variables in its Markov blanket
    - Algorithm wanders randomly around the state space of possible complete assignments, flips one variable at a time but keeps evidence variable fixed



* + - How to calculate Markov blanket distribution:



* + - The stationary distribution of the Gibbs sampling process is exactly the posterior distribution for the nonevidence variables conditioned on the evidence
    - The work required to generate each sample is independent of the size of the network
    - **Mixing rate**: the rate of convergence for Gibbs sampling of the markov chain
      * Depends strongly on quantitative properties of conditional distributions in the network
    - **Block sampling**: sampling multiple variables simultaneously
  + **Metropolis-Hastings (MH) Sampling** - designed to generate sample according to target probabilities
    - Sample a new state from a proposal distribution given the current state
    - Probabilistically accept or reject new state according to acceptance probability
    - If proposal is rejected, remain at current state
    - Proposal distribution is responsible for proposing next state
      * For example
        + With probability 0.95 perform Gibbs sampling step to generate new state
        + Otherwise generate new state by running WEIGHTED-SAMPLE algorithm
      * Make sure new states being proposed are reasonably likely
      * Make sure that the chain mixes well
        + Sometimes propose large moves to distant parts of the state space
    - *Converge to the correct stationary distribution is guaranteed for any proposal distribution*
    - Need to maximize: If a next state is proposed that is more likely than the current state it will definitely be accepted
      * If the proposed state is less likely than the current state its probability of being accepted drops proportionally
* Problem: **all the operations required to access the data structure of Bayes net are repeated a lot as the sampling algorithm runs and all these computations are unnecessary**
  + Network’s structure and conditional probabilities remain fixed through computation
    - Compile the network into model-specific inference code that carries out just the inference computations needed for that specific network
* **Causal Networks -** restricted class of Bayesian networks that forbids all buy causally compatible orderings
  + Devised so we can represent causal asymmetries and leverage the asymmetries towards reasoning with causal information
  + Decide on arrow directionality by considerations that go beyond probabilistic dependence and invoke different type of judgement
  + The value *xi* of each variable *Xi* is determined by an equation *xi* = *fi* (OtherVariables) and an arrow *Xj* *Xi* is drawn only if *Xj* is one of the arguments of *fi*
    - **Structural equation:** *xi* = *fi* 
      * Stable mechanism which remains invariant to measurements and local changes in environment
      * Unlike probabilities that quantify Bayesian network
  + Any local reconfiguration of the mechanisms in the environment can be translated into an isomorphic reconfiguration of the network topology
  + **Unmodeled variables (error terms or disturbances):** variables that perturb the functional relationship between each variable and its parents
    - If all U-variables are mutually independent random variables with suitably chosen priors, the joint distribution can be represented exactly by the structural equations
  + A system of stochastic relationships can be captured by a system of deterministic relationships each of which is affected by an exogenous disturbance
  + System of structural equations allows us to predict how interventions will affect the operation of the system and the observable consequences of those interventions
  + **Do-calculus**: action/operator that represents intervention to impose the condition
    - E.g. *do*(Sprinkler = true)
      * Different from observing Sprinkler = true
  + **Adjustment formula**: probability-weighted average of the influence of *Xj* on *Xi* where teh weights are the priors on the parent values



* + **Back-door criterion**: allows to write an adjustment formula that conditions on any set of variables that closes the back door
    - Provides a way to argue against dogma asserting that only randomized controlled trial can provide causal information