Bayesian Network and Explain Inference by Enumeration.

**Bayes's Rule.**

The product rule can be written in two forms:

\[
P(a \land b) = P(a | b) P(b)
\]
\[
P(a \land b) = P(b | a) P(a)
\]

Equating two right hands & divided by \( P(a) \)

\[
\frac{P(b | a)}{P(a)} = \frac{P(a \land b) P(b)}{P(a)}
\]

This equation is known as Bayes’s rule or

**Bayes’s Theorem.**

**Bayesian Network.** (Used to solve large no of variable)

A data structure called Bayesian Network to represent the dependencies among variable. It can represent essentially any full joint probability distribution & in many cases can do very concisely.

It is directed graph in which each node is annotated with quantitative probability information. The full specifications are:

1. Each node corresponds to a random variable, which may be discrete (or) continuous.
2. A set of directed links (or) arrows connects pair of nodes. If there is an arrow from node \( x \) to node \( y \), \( x \) is said to be parent of \( y \). The graph has no directed cycles. (DAG)
3. Each node \( x_i \) has a conditional probability distribution \( P(x_i | \text{Parents}(x_i)) \) that quantifies the effect of the parents on the node.
Inference by enumeration.

It is one of the exact inference in Bayesian Network.

Any conditional probability can be computed by summing terms from the full joint distribution.

$$P(x/e) = \alpha P(x,e) = \alpha \sum_y P(x,e,y)$$

$x$ denotes the query variable.
$e$ denotes the set of evidence variable.
$y$ denotes the hidden variable.
$\alpha$ is for Normalization.

A query can be answered using a Bayesian network by computing sums of products of conditional probabilities from the network.

Consider the query $P(\text{Burglary} | \text{JohnCalls} = \text{true}, \text{MaryCalls} = \text{true})$.

The hidden variables in this query are Earthquake & Alarm.
Using Initial letters to shorten the expression.

\[ P(B|j,m) = \alpha \sum \sum P(B, j, m) = \alpha \sum \sum P(B, j, m, e, a) \]

\[ e \quad a \]

Gives us an expression in terms of CPT entries.

=> we do this just for burglary to true.

\[ P(b|j, m) = \alpha \sum \sum p(b) P(e) P(a/b, e) P(j|a) P(m|a) \]

To compute this expression, we have to add four terms, each computed by multiplying 5 nos.

Complexity of the algorithm for a network with 'n' Boolean Variables is \( O(n^2) \)

From observation, P(b) term is a constant so we move that outside the summations over a and e.

P(e) moved outside the summation over a.

\[ P(b|j, m) = \alpha P(b) \sum P(e) \sum P(a/b, e) P(j|a) P(m|a) \]

To evaluate this by looping & multiplying CPT entries.

The structure will be

Algorithm.

function ENUMERATION-ASK(X, e, bn) returns a distribution over X

inputs: X, the query variable

e, observed values for variables E

bn, a Bayes net with variables \{X, Y, E, V\}

@ (x) \leftarrow a distribution over X, initially empty

www.studentsfocus.com
for each value $x_i$ of $X$ do

$$Q(x_i) \leftarrow \text{ENUMERATE-ALL} (bn, \text{VARS}, e_{xt})$$

where $e_{xt}$ is $e$ extended with $x = x_i$

return $\text{NORMALIZE}(Q(x_i))$

function $\text{ENUMERATE-ALL}(\text{VARS}, e)$ returns a real number

if $\text{EMPTY}(\text{VARS})$ then return 1.0

$y \leftarrow \text{FIRST}(\text{VARS})$

if $Y$ has value $y$ in $e$

then return $P(y/\text{parents}(y)) \times \text{ENUMERATE-ALL}(\text{REST(VARS)}, e)$

else return $\sum_y P(y/\text{parents}(y)) \times \text{ENUMERATE-ALL}(\text{REST(VARS)}, e_y)$

where $e_y$ is $e$ extended with $Y = y$. 

www.studentsfocus.com
Bayesian Network:

A Bayesian network is a directed graph in which each node is annotated with quantitative probability information. The full specification is as follows:

* Each node corresponds to a random variable, which may be discrete or continuous.
* A set of directed links (or) arrows connects pair of nodes.
* Each node $x_i$, has a conditional probability distribution.

Full joint distribution:

$P(\mathbf{x} = \mathbf{a}, \ldots, \mathbf{x}_n = \mathbf{a}_n)$. A Bayesian network is a directed acyclic graph with some numeric parameters attached to each node. A generic entry in the joint distribution is the probability of a conjunction of particular assignments to each variable, such as
\[ P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | \text{parents}(x_i)) \]

where \text{parents}(x_i) denotes the values of parents \(x_i\) that appear in \(x_1, \ldots, x_n\). Thus each entry in the joint distribution is represented by the product of the appropriate elements of the conditional probability table (CPT) in the Bayesian network.

\[ P(s, m, a, t_b, t_e) = P(s|a) P(m|a) P(a|t_b, t_e) P(t_b) P(t_e) \]

\[ = 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998 \]

\[ = 0.0000628 \]

The full joint distribution can be used to answer any query about the domain.

A method for constructing Bayesian networks:

\[ P(x_1, \ldots, x_n) = P(x_n | x_{n-1}, \ldots, x_1) \cdot P(x_{n-1}, \ldots, x_1) \]

Then we repeat the process, reducing each conjunctive probability to a conditional probability and a smaller conjunction. We end up with big product:

\[ P(x_1, \ldots, x_n) = P(x_n | x_{n-1}, \ldots, x_1) \cdot P(x_{n-1} | x_{n-2}, \ldots, x_1) \]

\[ \ldots \cdot P(x_2 | x_1) \cdot P(x_1) \]
This identity is called the chain rule.

\[ P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | x_{i-1}, \ldots, x_1). \]

The conditional distributions can be used for discrete variables, other representations including those suitable for continuous variables, conditional probability tables to representing Bayesian network is a correct representation of the domain only if each node is conditionally independent of its other predecessors in the ordering.

+ Nodes
+ Links
Inference using full joint distribution:

we introduce a method for probabilistic inference ie) a computation of posterior probabilities for query positions given observed evidence.

we use the full joint distribution and it is involved by computing conditional probability.

eg:

A domain consists

+ Toothache
+ Cavity
+ Catch

<table>
<thead>
<tr>
<th></th>
<th>tooth ache</th>
<th>7toothache</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Catch</td>
<td>TCatch</td>
</tr>
<tr>
<td>Cavity</td>
<td>0.108</td>
<td>0.012</td>
</tr>
<tr>
<td>7Cavity</td>
<td>0.016</td>
<td>0.064</td>
</tr>
</tbody>
</table>

There are 6 possible worlds Cavity v toothache

\[
P(Cavity \lor toothache) = 0.108 + 0.012 + 0.072 + 0.008 + 0.016 + 0.064
\]

\[
= 0.28
\]
One particularly common task is to extract the distribution over some subset of a variable or single variable. 

Eq:

Marginal probability (or) unconditional probability

\[ P(\text{Cavity}) = 0.108 + 0.102 + 0.072 + 0.008 = 0.3 \]

This is called marginalization. It is also

can be written in the form of

\[ P(Y) = \sum_{Z \in Z} P(Y, Z) \]

\[ P(\text{Cavity}) = \sum_{Z \in \{\text{Cough, Toothache}\}} P(\text{Cavity}, Z) \]

A variant of this rule involves conditional probability instead of joint probabilities, using the product rule

\[ P(Y) = \sum_{Z} P(Y | Z) P(Z) \]

This is called conditioning and is provided a sort of all kinds of probabilities.
\[ P(\text{cavity} \mid \text{toothache}) = \frac{P(\text{cavity} \land \text{toothache})}{P(\text{toothache})} \]

\[ = \frac{0.108 + 0.012}{0.108 + 0.012 + 0.016 + 0.064} \]

\[ = 0.6 \]

we can also compute there is no cavity given a toothache.

\[ P(\neg \text{cavity} \mid \text{toothache}) = \frac{P(\neg \text{cavity} \land \text{toothache})}{P(\text{toothache})} \]

\[ = \frac{0.106 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} \]

\[ = 0.4 \]
Bayesian networks:

To represent the dependencies among variables, a data structure called Bayesian networks.

1. A Bayesian network is a directed graph in which each node is annotated with quantitative probability information.
2. Each node corresponds to a random variable, which may be discrete or continuous.
3. A set of directed links or arrows connects pairs of nodes. If there is an arrow from node $X$ to node $Y$, $X$ is said to be a parent of $Y$. The graph that has no directed cycle is a directed acyclic graph or DAG.
4. Each node $X_i$ has a conditional probability distribution $P(X_i | \text{parents}(X_i))$ that quantifies the effect of the parents on the node.

Example for a simple Bayesian network:

![Diagram of a simple Bayesian network]

Consider a simple network consisting of variables: toothache, cavity, catch, and weather. Weather is independent of the other three variables. Toothache and catch are conditional independence that could be observed by there is no link between them.
Conditional independence of toothache and catch.

Given cavity $2$, indicated by the absence of a link between toothache and catch.

P: for typical Bayesian network:

```
  Burglary  Earthquake
  P(B) .001  P(E) .001

  John calls
  P(A)   
  t .90
  f .05

  Mary calls
  P(M)    
  t .70
  f .30
```

John nearly always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then, too. Mary, on the other hand, likes rather loud music and often misses the alarm altogether.

Given the evidence of who has or has not called, we have to estimate the probability of a burglary.

Conditional probability or CPT:

This form of table can be used for discrete variables, other representations, including those suitable for continuous variables.

Discrete random variables can be represented in the form of table known as conditional probability table.
Representing the full joint distribution:

A generic entry in the joint distribution is the probability of a conjunction of particular assignments to each variable, such as \( P(x_1 = x_1, \ldots, x_n = x_n) \). The formula for this entry is

\[
P(x_1, \ldots, x_n) = \prod_{i=1}^{n} \phi(x_i \mid \text{parents}(x_i)) \rightarrow 0
\]

where \( \text{parents}(x_i) \) denotes the value of \( \text{parents}(x_i) \) that appear in \( x_1, \ldots, x_n \). Thus each entry in the joint distribution is represented by the product of the appropriate element of the conditional probability tables (CPTs) in Bayesian network.

It is easy to prove that the parameters \( \phi(x_i \mid \text{parents}(x_i)) \) are exactly the conditional probabilities \( P(x_i \mid \text{parents}(x_i)) \) implied by joint distribution:

\[
P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid \text{parents}(x_i)). \rightarrow 0.
\]

Calculate the probability that the alarm has sounded, but neither a burglary nor an earthquake has occurred, and both John and Mary call.

\[
P(j, m, a, Tb, Te) = P(j \mid a) \cdot P(m \mid a) \cdot P(a \mid Tb \land Te) \cdot P(Tb) \cdot P(Te)
\]

\[
= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998.
\]

= 0.000628.
If a Bayesian network is a representation of the joint distribution, then it can be used to answer any query by summing all the relevant joint entries.

A method for constructing Bayesian networks:

\[ P(x_1, \ldots, x_n) = P(x_n \mid x_{n-1}, \ldots, x_1) P(x_{n-1}, \ldots, x_1) \]

Repeating the process, reducing each conjunctive probability and a smaller conjunction.

\[ P(x_1, \ldots, x_n) = P(x_n \mid x_{n-1}, \ldots, x_1) P(x_{n-1} \mid x_{n-2}, x_1) \ldots \]

\[ = \prod_{i=1}^{n} P(x_i \mid x_{i-1}, \ldots, x_1) \]

This identity is called the chain rule. It holds for any set of random variables, comparing it with eqn (8), that the specification of the joint distribution is equivalent to the general assertion that, for every variable \( x_i \) in network,

\[ P(x_i \mid x_{i-1}, \ldots, x_1) = P(x_i \mid \text{Parents}(x_i)) \rightarrow (8) \]

Condition:

1. **Nodes**: determining the set of variables that are required to model the domain.

2. **Links**: there will be a conditional probability between successors and predecessor nodes.
Bayesian Network:

A data structure to represent the dependencies among variables is called Bayesian Network.

A Bayesian Network is a directed graph in which each node is annotated with quantitative probability information. The full specification are as follows:

1. Each node corresponds to a random variable, may be discrete or continuous.
2. A set of directed links or arrows connect pair of nodes. If there is an arrow from \( x \) to node \( y \) then it is said to be \( x \) as a parent of \( y \). The graph has no directed cycles called directed acyclic graph (DAG).
3. Each node \( x_i \) has a conditional probability distribution \( P(x_i | \text{parent}(x_i)) \) what quantifies the effect of the parents on the node.

Approximate Inference in Bayesian Networks:

- Approximate Inference in Bayesian networks describes randomized sampling algorithms called Monte Carlo algorithm.

- Algorithm provides approximate answers whose accuracy depends on the number of samples generated.

- Two families of Algorithms are direct sampling and Markov chain planning.
Likelihood weighting:

Likelihood weighting avoids the inefficiency of a rejection sampling by generating only events that are consistent with the evidence e.

Function LIKELIHOOD-WEIGHTING (X, e, bn, N) returns an estimate of P(x|e).

Inputs: X, the query variable
e, observed values for variables E
bn, a Bayesian network specifying joint distribution P(x₁, ..., xₙ)
N, the total number of samples to be generated.

Local variables: W, a vector of weighted counts for each value of x, initially zero

for j = 1 to N do
  x, w ← WEIGHTED-SAMPLE (bn, e)
  W[x] ← W[x] + w where x is the value of x in x
return NORMALIZE (W)

Function WEIGHTED-SAMPLE (bn, e) returns an event and a weight
w ← 1; x ← an event with n elements initialized from e
for each variable xᵢ in x, ..., xₙ do
  if xᵢ is an evidence variable with xᵢ in e
  then w ← w × P(xᵢ = xᵢ | parents(xᵢ))
  else x[i] ← a random sample from P(xᵢ | parents(xᵰ))
return x, w
Likelihood-weighting fixes the value for the evidence variables E and samples only the nonevidence variables.

Let us apply the algorithm to the network with the query

\[ P(\text{Rain} \mid \text{Cloudy} = \text{true}, \text{WetGrass} = \text{true}) \]

and the ordering cloudy, sprinkler, rain, wetGrass.

First, the weight \( w \) is set to 1.0. The following are the events:
1. cloudy is an evidence variable with value true.

   Then
   \[ w' \leftarrow w \times P(\text{cloudy} = \text{true}) = 0.5 \]

2. sprinkler is not an evidence variable, so sample from \( P(\text{sprinkler} | \text{cloudy} = \text{true}) = <0.0, 0.9> \); suppose this returns false.

3. Similarly, sample from \( P(\text{rain} | \text{cloudy} = \text{true}) = <0.8, 0.2> \); suppose this returns true.

4. wetGrass is an evidence variable with value true.

   \[ w' \leftarrow w \times P(\text{wetGrass} = \text{true} | \text{sprinkler} = \text{false}, \text{rain} = \text{true}) = 0.45 \]

5. Similarly, sample from \( P(\text{rain} | \text{cloudy} = \text{true}) \); here the cloudy is an evidence variable = \( <0.2, 0.2> \); suppose this returns true.
Bayesian Network:

A Bayesian network is a directed graph in which each node is annotated with quantitative probability information. The full specification is as follows:

* Each node corresponds to a random variable which may be discrete or continuous.
* A set of directed links or arrows connects pairs of nodes.
* Each node $X_i$ has a conditional probability distribution.

Variable elimination algorithm:

The elimination algorithm can be improved substantially by eliminating repeated calculation. This idea is simple to do to calculation once and save the result and this is a form of dynamic programming.

The variable elimination algorithm is the simplest elimination works by evaluating expression from right to left such as the equation:

$$P(b|j,m) = \frac{P(c) \cdot P(e) \cdot P(a|b,c) \cdot P(j|a) \cdot P(m|a)}{P(c)}$$
Example:

\[
\begin{array}{c|c|c|c|c}
 & \text{Burglary} & \text{Earthquake} & \text{Alarm} & \text{Mary calls} \\
\hline
P(B) & 0.001 & & & \\
\hline
P(E) & & 0.002 & & \\
\hline
P(A|B,E) & t & 0.95 & 0.70 & \\
 & f & 0.24 & 0.01 & \\
\hline
P(A|B,\neg E) & t & 0.29 & & \\
 & f & 0.01 & & \\
\hline
A P(I) & t & 0.90 & & \\
 & f & 0.08 & & \\
\end{array}
\]

A Bayesian network for this domain is appears above. The network structure shows the burglary and earthquake directly affect the probability of the alarm's going off. But whether John and Mary call depends only on the alarm.

The network that represent our assumptions that they do not perceive burglaries directly; they do not notice minor earthquakes and they do not warn before calling.

The network structure depends on order of introduction; in each network we have introduced nodes in top-to-bottom order.
Let us illustrate this process for the burglary network. Then we evaluate the expression as:

\[ P(B_{j,m}) = \frac{\alpha \cdot P(B) \cdot \frac{\sum_{e} P(e) \cdot \frac{P(a|B,e) \cdot P(j|a) \cdot P(m|a)}{f_2}}{P(j|a)} \cdot f_3 \cdot f_4 \cdot f_5}}{f_1} \]

Consider \( f_4 \) and \( f_5 \):

\[ f_4(A) = \frac{P(j/a)}{P(j|a)} = \left( \frac{0.90}{0.99} \right) \]

\[ f_5(A) = \frac{P(m/a)}{P(m|a)} = \left( \frac{0.70}{0.01} \right) \]

\[ P(a/B,e) = 0.95 \]

And the above expression written as:

\[ P(B_{j,m}) = \alpha \cdot f_1(B) \cdot \frac{\sum_{e} f_2(E) \cdot \sum_{e} f_3(A,B,E) \cdot f_4(A) \cdot f_5(A)}{f_1} \]

First we sum out \( A \) from the product \( f_3, f_4, \) and \( f_5 \). This gives new factor \( f_6(B,E) \) ranges over \( B \) and \( E \):

\[ f_6(B,E) = \sum_{a} f_3(A,B,E) \cdot f_4(A) \cdot f_5(A) \]

Now we are left with the expression:

\[ P(B_{j,m}) = \alpha \cdot f_1(B) \cdot \frac{\sum_{e} f_2(E) \cdot f_6(B,E)}{f_1} \]

Now we sum \( f_2 \) and \( f_6 \) we get \( f_7 \):

\[ f_7(B) = \sum_{e} f_2(E) \cdot f_6(B,E) \]

Thus:

\[ P(B_{j,m}) = \alpha \cdot f_1(B) \cdot f_7(B) \]
Examining this sequence, we see the two basic computational operations are required: they are pointwise product of a pair of factors and summing out a variable from a product of factors.

Algorithm:

Variable Elimination algorithm for inference in Bayesian network.

\textbf{function}\ \textsc{Elimination-Ask}\ (x, e, bn) \textbf{return} \textit{a distribution over} x.

\textbf{inputs:} x, the query variable

e, observed values for variable E

bn, a bayesian network specifying joint distribution \( P(x_1, \ldots, x_n) \)

\textbf{factors} \leftarrow \emptyset

\textbf{for each} \textsc{var} \textbf{in} \textbf{order (bn, vars)} \textbf{do}

\textbf{factors} \leftarrow [\textsc{make-factor} (\textsc{var}, e) | \textbf{factors}]

\textbf{if} \textsc{var} \textbf{is a hidden variable then}

\textbf{factors} \leftarrow \textsc{sum-out} (\textsc{var}, \textbf{factors})

\textbf{return} \textsc{normalize} (\textsc{pointwise-product} (\textbf{factors}))
1) Approximate Inference in Bayesian Networks:

* Bayesian Network is a data structure that represents the dependencies among the variables.

* A Bayesian network is a directed graph in which each node is annotated with quantitative probability information.

* In a Bayesian network,

1) Each node corresponds to a random variable, which may be discrete or continuous.

2) The arrow connects the pair of nodes. If there is an arrow from node X to node Y, then X is said to be the parent of Y. The graph has no directed cycles.

3) Each node X_i has a conditional probability distribution P(X_i | Parents(X_i)) that quantifies the effect of the parents of X_i on the node.

* Approximate inference provides approximate answers whose accuracy depends on the number of samples generated.

* Four mechanisms used for approximate inference are:

  1) Direct sampling
  2) Rejection sampling
  3) Markov chain sampling
  4) Likelihood weighting

Direct sampling methods:

* The primitive element in any sampling algorithm is the generation of samples from a known
probability distribution.

* The simplest kind of random sampling process for Bayesian networks generates events from a network that has no evidence associated with it.

* The idea is to sample each variable in the topological order.

* For example, consider the following ordering

[Cloudy, Sprinkler, Rain, WetGrass]

The given samples are,

i) Sample from \( P(\text{Cloudy}) = <0.5, 0.5> \), value is true.

ii) Sample from \( P(\text{Sprinkler} | \text{Cloudy} = \text{true}) = <0.1, 0.9> \), value is false.

iii) Sample from \( P(\text{Rain} | \text{Cloudy} = \text{true}) = <0.8, 0.2> \), value is false.

iv) Sample from \( P(\text{WetGrass} | \text{Sprinkler} = \text{false}, \text{Rain} = \text{true}) = <0.9, 0.1> \), value is true.

\[
P(C) = 0.5
\]
\[ P(C) = 0.5 \]

Cloudy

Spr + Rain

Wet grass

<table>
<thead>
<tr>
<th>5 + R</th>
<th>P(W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T T</td>
<td>0.99</td>
</tr>
<tr>
<td>T F</td>
<td>0.90</td>
</tr>
<tr>
<td>F T</td>
<td>0.90</td>
</tr>
<tr>
<td>F F</td>
<td>0.00</td>
</tr>
</tbody>
</table>

\[ P(5+R = x) \]

<table>
<thead>
<tr>
<th>C</th>
<th>P(5+R = x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T T</td>
<td>0.80 0.02 0.12 0.18</td>
</tr>
<tr>
<td>T F</td>
<td>0.10 0.40 0.10 0.40</td>
</tr>
</tbody>
</table>

* To find the \( P(\text{Wet grass} | \text{Rain, Sprinkler}) \).

Here, the evidence are Rain and Sprinkler.

The hidden variable is Cloudy.

* In this case, prior sample returns the event [True, False, True, True].

**Sampling Algorithm:**

function **prior-sample** (bn) returns an event sampled from the prior specified by bn.

inputs : bn, a Bayesian network specifying joint distribution \( P(x_1, \ldots, x_n) \)

\( x \) ← an event with \( n \) elements.

for each variable \( x_i \) in \( x_1, \ldots, x_n \) do

\( x[i] \) ← a random sample from \( P(x_i | \text{Parents}(x_i)) \)

return \( x \)
Rejection Sampling:

It is a general method for producing samples from a hard-to-sample distribution given an easy-to-sample distribution.

Let $P(x|e)$ be the estimated distribution from that we have

$$P(x|e) = \alpha \frac{N_{ps}(x|e)}{N_{ps}(e)}$$

$\alpha$ - evident true.

$$\Rightarrow P(x|e) \approx \frac{P(x, e)}{P(e)}$$

Let us assume that we have 100 samples.

To find $P(Rain | Sprinkler = true)$

Suppose out of 100 samples 73 have Sprinkler = False are rejected. 27 have Sprinkler = true.

Out of 27, 8 have Rain = true & 19 have Rain = false.

$$\therefore P(Rain | Sprinkler = true) \approx \text{NORMALIZE}(8, 19) \approx \frac{8}{27} = \langle 0.296, 0.704 \rangle$$

Algorithm:

function REJECTION-SAMPLING$(x, e, bn, N)$ returns

$P_{No.\ 54.2}$
1. Why we move to clustering algorithms in exact inference in Bayesian networks?

**Bayesian Networks:**

Bayesian networks is a static structure to represent the dependencies among variables. Bayesian networks can represent essentially any full joint probability distribution.

A Bayesian network is a directed graph, it contains the following information:

- Each node correspond to a random variable which may be discrete or continuous.
- A set of directed link or arrows connected pairs of nodes.
- Each node $X_i$ has a conditional probability distribution $P(X_i | \text{parent}(X_i))$ that quantifies the effect of the parent on the node.

**Exact Inference in Bayesian Networks:**

The basic task for any probabilistic inference system is to compute the posterior probability distribution for a set of query variables, event $E$ and some assignment of values to a set of evidence variables. It take one queries at a time.

$X$ denotes the query variable, $E$ denotes the set of evidence variables $E_1, \ldots, E_m$ and $O$ is a particular observed event, $Y$ will denote the non evidence, non query variables $Y_1, \ldots, Y_l$ called as hidden variables.

Exact algorithm for computing posterior probabilities and will consider the complexity of this task.

8 covers methods for approximate inference. Here we discuss one of the method that is clustering method.
Clustering algorithm:

Drawback of the complexity of exact inference:

It avoids the repeated computations as well as dropping irrelevant variables. The time and space requirements of variable elimination are dominated by the size of the largest factor constructed during the operation of the algorithm.

Eqn: There is at most one undirected path between any two nodes in the network. It is called as singly connected networks (SCNs) polytrees. The particular property of this problem is

- The time and space complexity of exact inference in polytrees is linear on the size of the network. Here, the size is defined as the number of CPD entries. The complexity will also be linear in the number of nodes.

For multiply connected networks, variable elimination can have exponential time and space complexity in the worst case.

So we move to the clustering algorithm.

Clustering algorithm: Multiple path to single path by combining nodes.

The variable elimination algorithm is simple and efficient for answering individual queries.

If we want to compute posterior probabilities for all the variables in a network, it can be left inefficient. For example, in a polytree network, one would need to issue $O(n)$ queries testing each, for a total of $O(n^2)$ time.

Using clustering algorithm it also known as

Jeffreys tree algorithm.
The time can be reduced to O(n), the algorithms are widely used in commercial Bayesian networks.

The basic idea of clustering is to join individual nodes of the network to form cluster nodes, that the resulting network is a polytree network.

For example, multiple unconnected network.

```
<table>
<thead>
<tr>
<th></th>
<th>P(C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>0.5</td>
</tr>
<tr>
<td>t</td>
<td>0.1</td>
</tr>
<tr>
<td>f</td>
<td>0.8</td>
</tr>
</tbody>
</table>
```

```
cloudy

<table>
<thead>
<tr>
<th></th>
<th>P(C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>0.5</td>
</tr>
</tbody>
</table>
```

```
sprinkler
```

```
<table>
<thead>
<tr>
<th></th>
<th>P(C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>0.2</td>
</tr>
<tr>
<td>t</td>
<td>0.8</td>
</tr>
<tr>
<td>f</td>
<td>0.2</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>P(C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>0.99</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>0.90</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>0.90</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>0.90</td>
</tr>
</tbody>
</table>
```

It is a multiply connected network with conditional probability tables where two variables are used as a Boolean variables to check no conditions if it is true or false.

Then we convert the conditional probability tables to the clustering equivalent of the multiply connected network by its converted into a polytree by combining the sprinkler and rain node into a cluster node called as sprinkler+rain.

Sprinkler and the rain are called as two Boolean nodes, the two Boolean nodes are replaced by one mega node called as sprinkler+rain.
It takes on four values as Tt, TF, PT and PF. The clustered equivalent of the multiply connected network is given below.

\[
\begin{array}{c}
\text{Cloudy} \\
\downarrow \\
\text{Spr + Rain} \\
\downarrow \\
\text{Wed Ones}
\end{array}
\]

The node (Spr + Rain) has only one parent, the boolean variable Cloudy, so there are two conditioning causes.

Once the network is in polytree form, a special-purpose inference algorithm is applied. Essentially, the algorithm is a form of constraint propagation, where the constraints ensure that neighboring clusters agree on the posterior probability of any variables that they have a common.

The algorithm is able to compute posterior probabilities for all the non-evidence nodes in the network in time \(O(n)\), where \(n\) is the size of the modified network.

The no-hardness of the problem has not disappeared, if a network requires exponential time and space with variable elimination. The clustering network require exponential time and space to construct.