PROBABILISTIC GRAPHICAL MODELS

Unit # 10

ACKNOWLEDGEMENTS

Some of the material in this presentation is taken from Sucar's book "Probabilistic Graphical Models" (Chapter 4).

APPROXIMATE INFERENCE MECHANISMS

Simulation Based Schemes

- Logic Sampling
- Likelihood Weighting

Approximate Algorithms

- Loopy Belief Propagation

SAMPLING BASED INFERENCE

STOCHASTIC SIMULATION

Stochastic simulation algorithms consist in *simulating* the BN several times, where each simulation gives a sample value for all non-instantiated variables.

These values are chosen randomly according to the conditional probability of each variable.

This process is repeated *N* times, and the posterior probability of each variable is approximated in terms of the frequency of each value in the sample space.

This gives an estimate of the posterior probability which depends on the number of samples; however, the computational cost is not affected by the complexity of the network.

LOGIC SAMPLING

Logic sampling is a basic stochastic simulation algorithm that generates samples according to the following procedure:

1. Generate sample values for the root nodes of the BN according to their prior probabilities. That is, a random value is generated for each root variable X, following a distribution according to $P(X)$.
2. Generate samples for the next *layer*, that is the sons of the already sampled nodes, according to their conditional probabilities, $P(Y \mid Pa(Y))$, where $Pa(Y)$ are the parents of Y.
3. Repeat (2) until all the leaf nodes are reached.

LOGIC SAMPLING (CONT'D)

The direct application of the previous procedure gives an estimate of the marginal probabilities of all the variables when there is no evidence.

If there is evidence (some variables are instantiated), all samples that are not consistent with the evidence are discarded and the posterior probabilities are estimated from the remaining samples.

A disadvantage of logic sampling when evidence exists is that many samples have to be discarded; this implies that a larger number of samples are required to have a *good* estimate.

LIKELIHOOD WEIGHTING (LASKEY)

Logic sampling can be very inefficient because it throws out so many observations.

It can be modified to avoid throwing out observations.

To take one random draw (one observation on all variables):

- The evidence variables are fixed to their observed states.
- Generate random samples for the non-evidence variables.
- Weight the observation by a “sampling weight” that depends upon the evidence.

Estimate $P(Xt|Xe)$ as the weighted average.

COMPUTING WEIGHT

Likelihood weighting generates samples in the same way as logic sampling; however, when there is evidence the non consistent samples are not discarded.

Instead, each sample is given a weight according to the weight of the evidence for this sample.

Given a sample s and the evidence variables $\mathbf{E} = \{E1, \ldots, Em\}$, the weight of sample s is estimated as:

$$W(\mathbf{E} \mid s) = P(E1)P(E2) \ldots P(Em)$$

where $P(Ei)$ is the probability of the evidence variable Ei for that sample.

COMPUTING POSTERIOR PROBABILITY

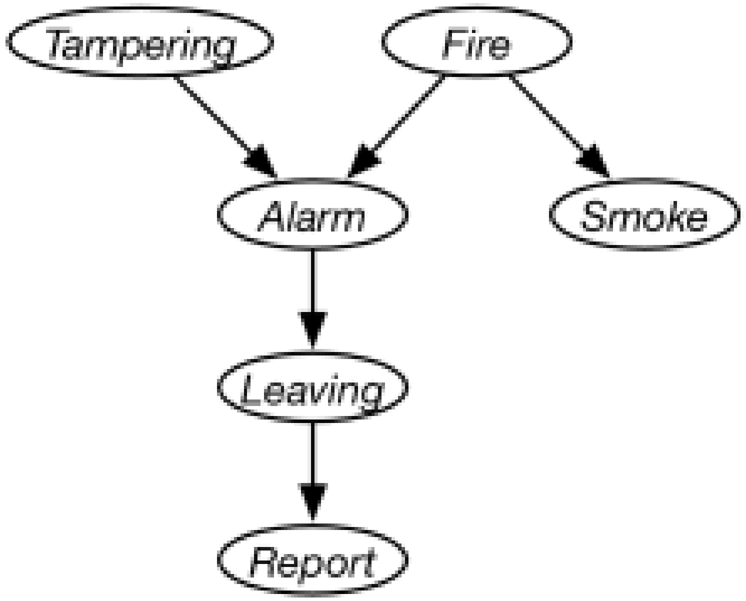
The posterior probability for each variable *X* taking value *xi* is estimated by dividing the sum of the weights *Wi (X = xi)* for each sample where *X = xi* by the total weight for all the samples:

$$P(X = x_i) \sim \sum_i W_i(X = xi) / \sum_i W_i$$

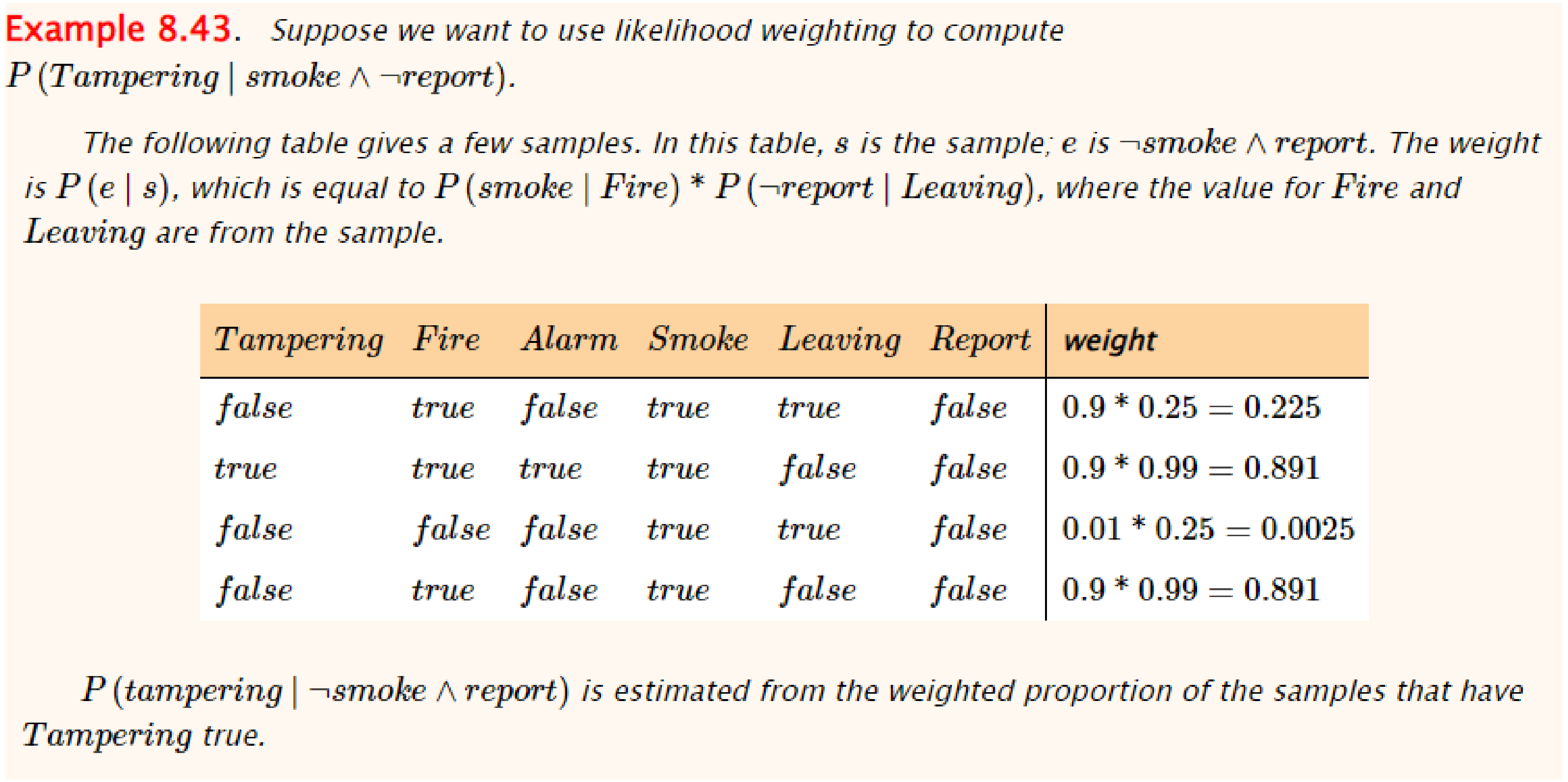
ALGORITHM (FROM JENSEN AND NIELSEN)

```
1. Let $(X_1, \ldots, X_n)$ be a topological ordering of the variables.
2. For $j = 1$ to $N$:
    a) w:=1.
    b) For $i = 1$ to $n$:
     - Let $\mathbf{x}'$ be the configuration of $(X_1, \ldots, X_{i-1})$ specified by $\mathbf{e}$ and the
       previous samples.
     - If $X_i \notin \mathcal{E}$, then:
         - Sample a state $x_i$ for $X_i$ using $P(X_i \mid \mathrm{pa}(X_i) = \pi)$, where
           $\mathrm{pa}(X_i) = \pi$ is consistent with $\mathbf{x}'$.
     - else
           $w := w \cdot P(X_i = e_i \mid \mathrm{pa}(X_i) = \pi)$, where $\mathrm{pa}(X_i) = \pi$ is consistent
           with $\mathbf{x}'$.
    c) $N(X_k = x_k) := N(X_k = x_k) + w$, where $x_k$ is the sampled state for
       $X_k$.
3. Return:
```

$$P(X_k = x_k \mid \mathbf{e}) \approx \frac{N(X_k = x_k)}{\sum_{x \in \mathrm{sp}(X_k)} N(X_k = x)}.$$



$P(tampering) = 0.02$
$P(fire) = 0.01$
$P(alarm \mid fire \wedge tampering) = 0.5$
$P(alarm \mid fire \wedge \neg tampering) = 0.99$
$P(alarm \mid \neg fire \wedge tampering) = 0.85$
$P(alarm \mid \neg fire \wedge \neg tampering) = 0.0001$
$P(smoke \mid fire) = 0.9$
$P(smoke \mid \neg fire) = 0.01$
$P(leaving \mid alarm) = 0.88$
$P(leaving \mid \neg alarm) = 0.001$
$P(report \mid leaving) = 0.75$
$P(report \mid \neg leaving) = 0.01$



Example 8.43. *Suppose we want to use likelihood weighting to compute* $P(Tampering \mid smoke \wedge \neg report)$.

The following table gives a few samples. In this table, s *is the sample;* e *is* $\neg smoke \wedge report$. *The weight is* $P(e \mid s)$, *which is equal to* $P(smoke \mid Fire) * P(\neg report \mid Leaving)$, *where the value for* $Fire$ *and* $Leaving$ *are from the sample.*

$Tampering$	$Fire$	$Alarm$	$Smoke$	$Leaving$	$Report$	***weight***
$false$	$true$	$false$	$true$	$true$	$false$	$0.9 * 0.25 = 0.225$
$true$	$true$	$true$	$true$	$false$	$false$	$0.9 * 0.99 = 0.891$
$false$	$false$	$false$	$true$	$true$	$false$	$0.01 * 0.25 = 0.0025$
$false$	$true$	$false$	$true$	$false$	$false$	$0.9 * 0.99 = 0.891$

$P(tampering \mid \neg smoke \wedge report)$ *is estimated from the weighted proportion of the samples that have* $Tampering$ *true.*

8.6.4 Likelihood Weighting▸ 8.6 Stochastic Simulation ▸ Chapter 8 Reasoning with Uncertainty ▸ Artificial Intelligence: Foundations of Computational Agents, 2nd Edition

SAMPLING BASED INFERENCE

LOOPY BELIEF PROPAGATION

This is simply the application of the probability propagation algorithm for multi-connected networks.

Although in this case the conditions for this algorithm are not satisfied, and it only provides an efficient approximate solution for the inference problem.

The propagation is repeated several times. The procedure is the following:

1. Initialize the λ and π values for all nodes to random values.
2. Do probability propagation according to the algorithm for singly connected networks.

ASSUMPTION

In loopy belief propagation, we make certain independence assumption (assuming that no loop exists when one exist).

The algorithm converges when the difference between the posterior probabilities for all variables of the current and previous iterations is below a certain threshold.

It has been found empirically that for certain structures this algorithm converges to the true posterior probabilities; however, for other structures it does not converge

CONCEPT BEHIND LOOPY BELIEF PROPAGATION

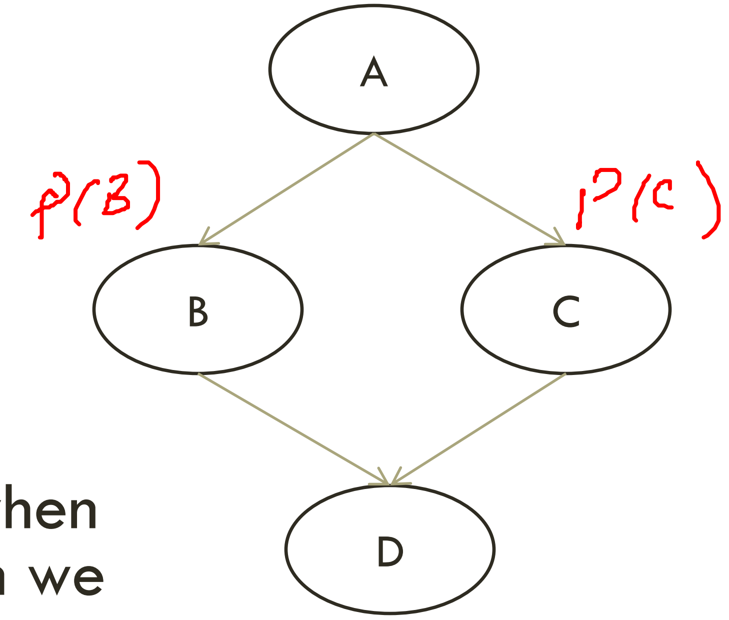
$P(B) = P(B|A)P(A)+P(B|{\sim}A)P({\sim}A)$

$P(C) = P(C|A)P(A)+P(C|{\sim}A)P({\sim}A)$

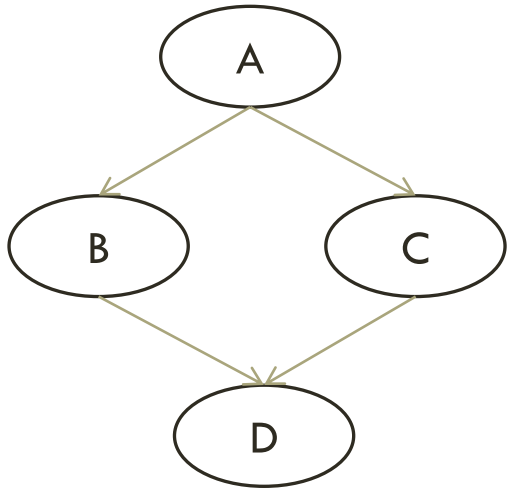
$P(D) = P(D|B,C)P(B,C) + P(D|B,{\sim}C)P(B,{\sim}C) + P(D|{\sim}B,C)P({\sim}B,C) + P(D|{\sim}B,{\sim}C)P({\sim}B,{\sim}C)$

If we assume that B and C are independent (when they are not as obvious from the structure) then we can compute the above equation as

$P(D) = P(D|B,C)\underline{P(B)P(C)} + P(D|B,{\sim}C)P(B)P({\sim}C) + P(D|{\sim}B,C)P({\sim}B)P(C) + P(D|{\sim}B,{\sim}C)P({\sim}B)P{\sim}C)$



EXAMPLES



SET I

P(A) = 0.3

P(B|A)=0.7, P(B|~A)=0.05

P(C|A)=0.2, P(C|~A)=0.9

P(D|B,C) =0.9, P(D|B,~C) =0.6

P(D|~B,C)=0.8, P(D|~B,~C)=0.02

SET II

P(A) = 0.3

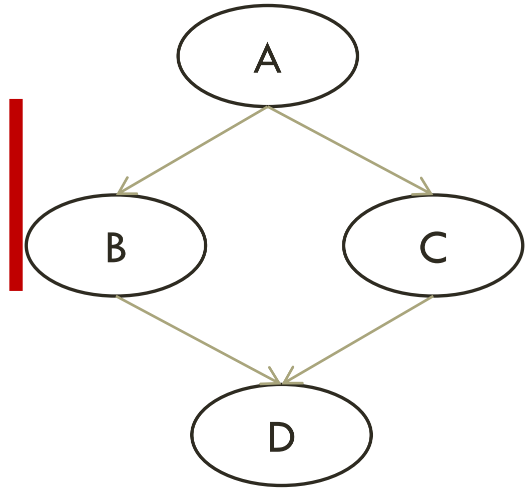
P(B|A)=0.9, P(B|~A)=0.01

P(C|A)=0.02, P(C|~A)=0.95

P(D|B,C) =0.99, P(D|B,~C) =0.95

P(D|~B,C)=0.90, P(D|~B,~C)=0.02

Using Set I and II, compute the probability of D, P(D), using loopy belief propagation and compare the results with the ones obtained through GeNIe (exact, logic sampling and likelihood weighting). What do you observe?



SET I

P(A) = 0.3
P(B|A)=0.7, P(B|~A)=0.05
P(C|A)=0.2, P(C|~A)=0.9
P(D|B,C) =0.9, P(D|B,~C) =0.6
P(D|~B,C)=0.8, P(D|~B,~C)=0.02

SET II

P(A) = 0.3
P(B|A)=0.9, P(B|~A)=0.01
P(C|A)=0.02, P(C|~A)=0.95
P(D|B,C) =0.99, P(D|B,~C) =0.95
P(D|~B,C)=0.90, P(D|~B,~C)=0.02

$P(B) = 0.7 \times 0.3 + 0.05 \times 0.7$
$= 0.245$

$P(C) = 0.2 \times 0.3 + 0.9 \times 0.7$
$= 0.69$

$P(D) = 0.9 \times 0.245 \times .69 + .6 \times .245 \times .31$
$+ 0.8 \times 0.76 \times .69 + .02 \times .76 \times .245$
$= 0.62$

$P(B) = .9 \times .3 + 0.01 \times .7$
$= .277$

$P(C) = 0.02 \times .3 + 0.95 \times .7$
$= .671$

$P(D) = .99 \times .277 \times .671$
$+ .95 \times .277 \times .329$
$+ .90 \times .723 \times .671$
$+ .02 \times .723 \times .329$
$= .707$

KULLBACK-LEIBLER DIVERGENCE

The performance of a particular approximate inference algorithm is assessed via Kullback-Leibler divergence.

$$KL(P, P') = \sum_i P(i) \log \frac{P(i)}{P'(i)}$$

KULLBACK-LEIBLER DIVERGENCE

The performance of a particular approximate inference algorithm is assessed via Kullback-Leibler divergence.

$$KL(P,P') = \sum_i P(i) \log \frac{P(i)}{P'(i)}$$

$$\left(0.3 \log_2 \frac{0.3}{0.31}\right) + \left(.25 \log_2 \frac{.25}{.25}\right)$$
$$+ \left(0.65 \log_2 \frac{0.65}{0.66}\right)$$
$$+ \left(0.67 \log_2 \frac{0.67}{0.65}\right) =$$

KL-DIVERGENCE

Entropy: In the context of information theory, entropy is a measure of the uncertainty in a random variable. In other words, it measures the "surprise" you expect to have on average when you learn the outcome of the variable. The entropy of a discrete random variable X with probability mass function p(x) is defined as:

$$(X) = -\sum[p(x)\log(p(x))]$$

The formula for **KL Divergence** for discrete probability distributions P and Q is defined as:

$$D_{\mathrm{KL}}(P||Q) = \sum P(x)\log\left(\frac{P(x)}{Q(x)}\right)$$

where the sum is over all possible outcomes.

KL divergence of Q from P is a measure of the information lost when Q is used to approximate P.

THANKS