

Bayesian Networks II: BN Inference

For Self-learning Purposes

Jiashu Chen

February 8, 2024

Contents

1	Recall: BN Representation	1
1.1	What is Bayesian Networks	1
2	Probabilistic Inference	3
2.1	Inference	3
2.2	Types of inferences	3
3	Inference by Enumeration	5
3.1	General Case	5
3.2	Procedures	5
3.3	Problem with Enumeration	5
4	Inference by Variable Elimination	6
4.1	Inference by Enumeration Versus Variable Elimination	6
4.2	Factors	6
4.3	Traffic Example	8
4.4	Variable Elimination: Marginalize Early!	10
4.5	Evidence	11
4.6	General Variable Elimination	11
4.7	Variable Elimination Ordering	12
4.8	Variable Elimination Complexity	12
5	Sampling	13
5.1	Approximate Inference: Sampling	13
5.2	Prior Sampling	14
5.3	Rejection Sampling	15
5.4	Likelihood Sampling	15
5.5	Example	17
5.6	Gibbs Sampling	19

Chapter 1

Recall: BN Representation

1.1 What is Bayesian Networks

- BN is a directed, acyclic graph, one node per random variable.
- A conditional probability table (CPT) for each node. A collection of distributions over X , one for each combination of parents' values.

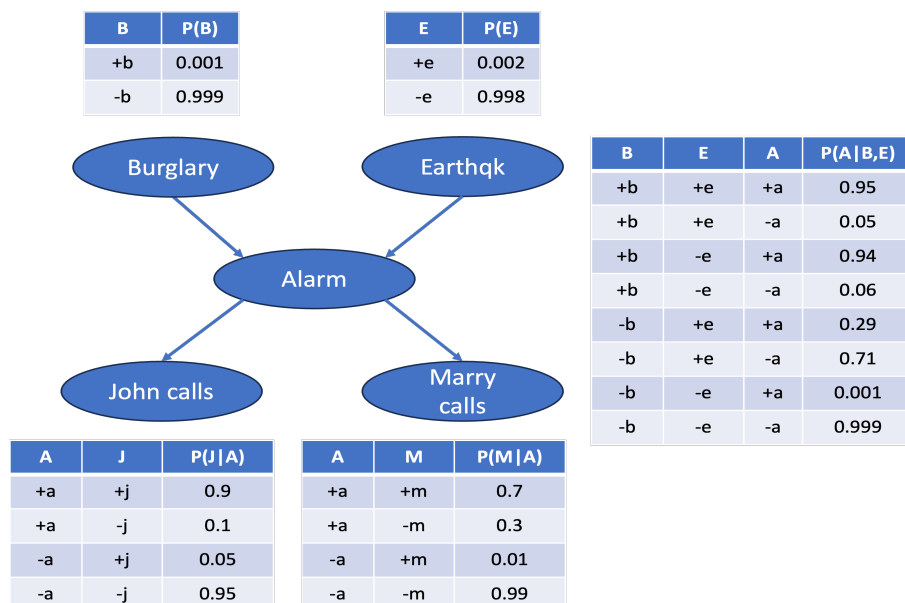
$$P(X \mid a_1, \dots, a_n)$$

- Bayesian networks implicitly encode joint probability distributions.
 - As a product of local conditional distributions
 - To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together.

$$P(x_1, x_2, \dots, x_n) = \prod_{i=1}^n P(X_i \mid \text{parents}(X_i))$$

- If a node has no parent, then it is independent probability distribution. On the other hand, if nodes have many parents, then the BN will become complicated. In general, we want all variables connected but a small number of parents for each node so that each $P(X_i \mid \text{parents}(X_i))$ term will be simple.

- Example



Suppose we want to compute the $P(+b, -e, +a, -j, +m)$

$$P(+b, -e, +a, -j, +m) = P(+b)P(-e)P(+a \mid +b, -e)P(-j \mid +a)P(+m \mid +a) = \\ 0.001 * 0.998 * 0.94 * 0.1 * 0.7$$

Chapter 2

Probabilistic Inference

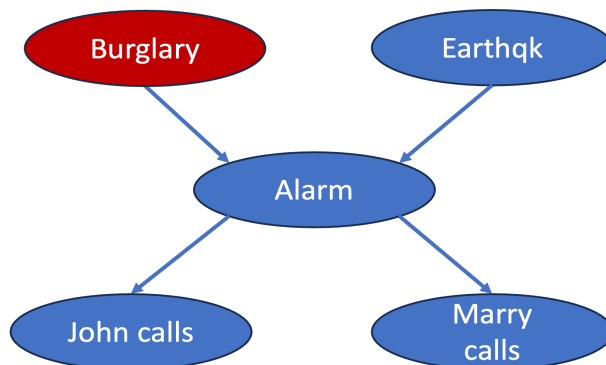
2.1 Inference

- Inference: calculating some useful quantity from a joint probability distribution. For instance, posterior probability $P(Q \mid E_1 = e_1, \dots, E_k = e_k)$, most likely explanation $\operatorname{argmax}_q P(Q = q \mid E_1 = e_1, \dots, E_k = e_k)$

2.2 Types of inferences

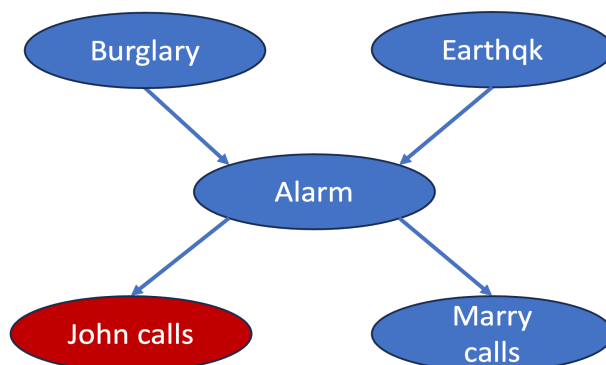
- Causal Reasoning

If there is a burglary, what is the probability that John calls, $P(j|b)$?



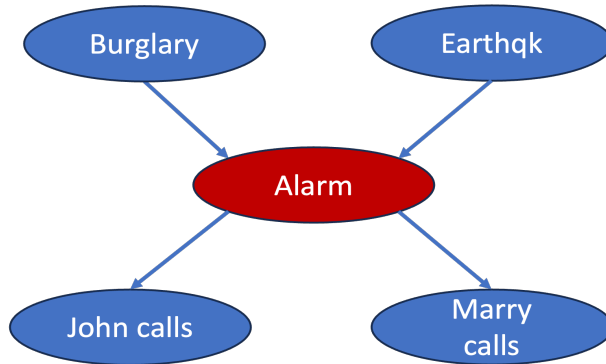
- Evidential Reasoning

If there is John calls, what is the probability that there is a Burglary, $P(b|j)$?



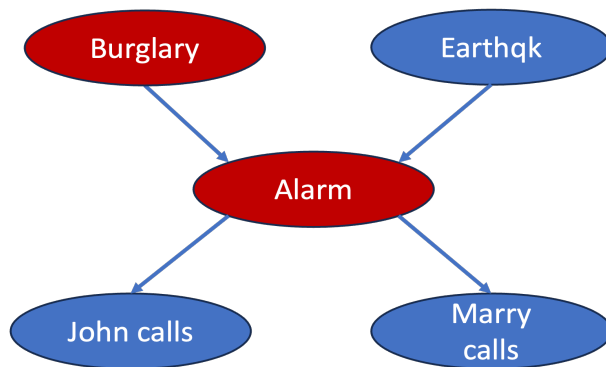
- Intercausal Reasoning

If there is alarm, the probability of earthquake versus the probability of burglary $P(b|a)$ versus $P(e|a)$?



- Intercausal Reasoning

If there is alarm and burglary, what is the probability of earthquake? $P(e|a,b)$?



Chapter 3

Inference by Enumeration

3.1 General Case

All variables can be divided into three categories:

- Evidential variables: $E_1, \dots, E_k = e_1, \dots, e_k$
- Query variable: Q
- Hidden variables: H_1, \dots, H_r

3.2 Procedures

- Step1: select the entries consistent with evidence.
- Step2: Sum out H (marginalize hidden variables) to get joint probability of query and evidence. Results becomes one-dimmmensional array where each row differs by the value of Q .

$$P(Q, e_1, \dots, e_k) = \prod_{h_1, \dots, h_r} P(Q, h_1, \dots, h_r, e_1, \dots, e_k)$$

- Step3: Normalize

$$Z = \sum q P(Q, e_1, \dots, e_k)$$
$$P(Q \mid e_1, \dots, e_k) = \frac{1}{Z} P(Q, e_1, \dots, e_k)$$

3.3 Problem with Enumeration

- Inference by enumeration is extremely slow (exponential time complexity), because we multiply all pieces of local conditional probabilities to form a exponential large joint probability. And then we collapse all variables we do not care about (hidden variables) to make it small again. We inflate and then deflate.
- Better idea: interleave joining and marginalizing (variable elimination).

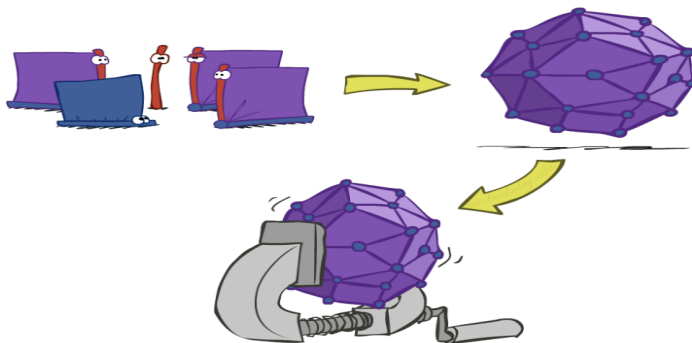
Chapter 4

Inference by Variable Elimination

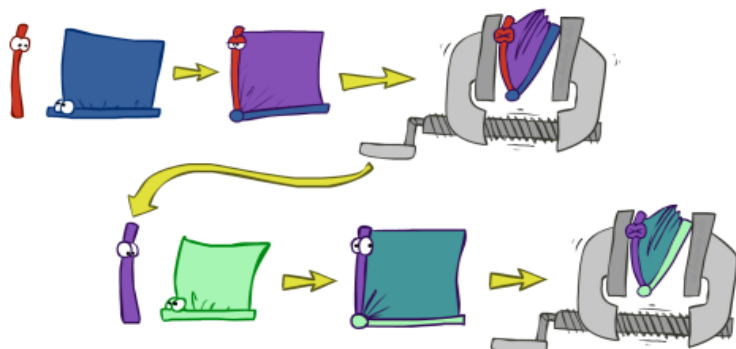
4.1 Inference by Enumeration Versus Variable Elimination

In enumeration, we join up the whole joint distribution before sum out hidden variables. In variable elimination, we join some variables and immediately sum out, and join next variables and sum out. We sum out hidden variables as soon as we can. Thus, we could control the growth speed. Variable elimination is still NP-hard, but it is usually much faster than inference by enumeration.

- Inference by Enumeration



- Variable Elimination

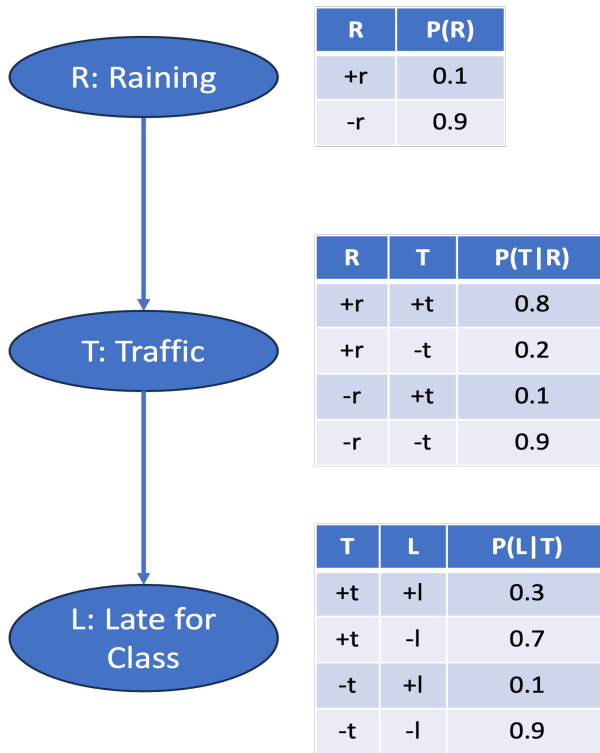


4.2 Factors

- Joint distribution: $P(X, Y)$

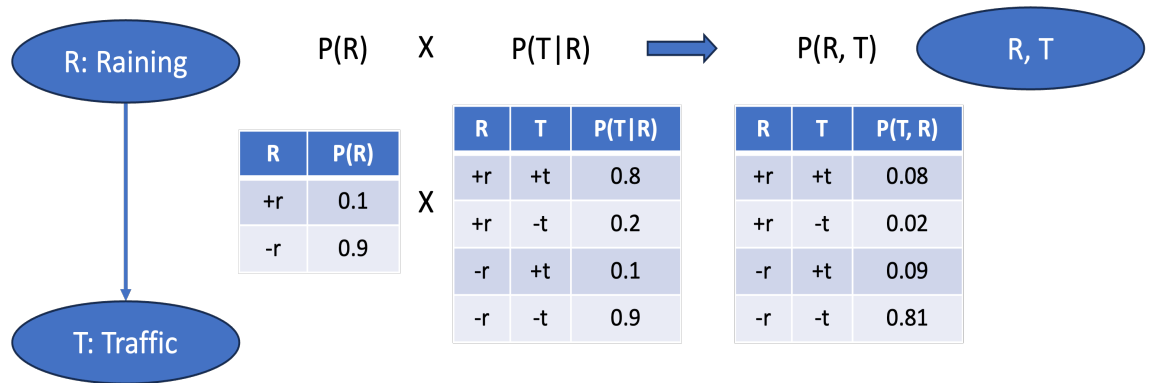
- entries $P(x, y)$ for all x, y
 - sums to 1
 - example: $P(T, W)$ (2d array: the number of capital letters is the number of dimensionality.)
- Selected join: $P(x, Y)$
 - a slice of the joint distribution
 - entries $P(x, y)$ for fixed x , all y
 - sums to $P(x)$
 - example: $P(\text{cold}, W)$ (1d array)
- Single conditional: $P(Y | x)$
 - entries $P(y | x)$ for fixed x , all y
 - sums to 1
 - example: $P(W, \text{cold})$ (1d array)
- Family of conditionals: $P(Y | X)$
 - multiple conditionals
 - entries $P(y | x)$ for all x, y
 - sums to $|X|$
 - example: $P(W | T)$ (2d array)
- Specified family: $P(y | X)$
 - entries $P(y | x)$ for fixed y but for all x
 - example: $P(\text{rain} | T)$ (1d array)
- Summary
 - $P(Y_1, \dots, Y_n | X_1, \dots, X_M)$
 - It is a factor, a multi-dimensional array
 - Its values are $P(y_1, \dots, y_n | x_1, \dots, x_M)$
 - Any assigned (lower case) X or Y is a dimension missing (or selected) from the array.

4.3 Traffic Example

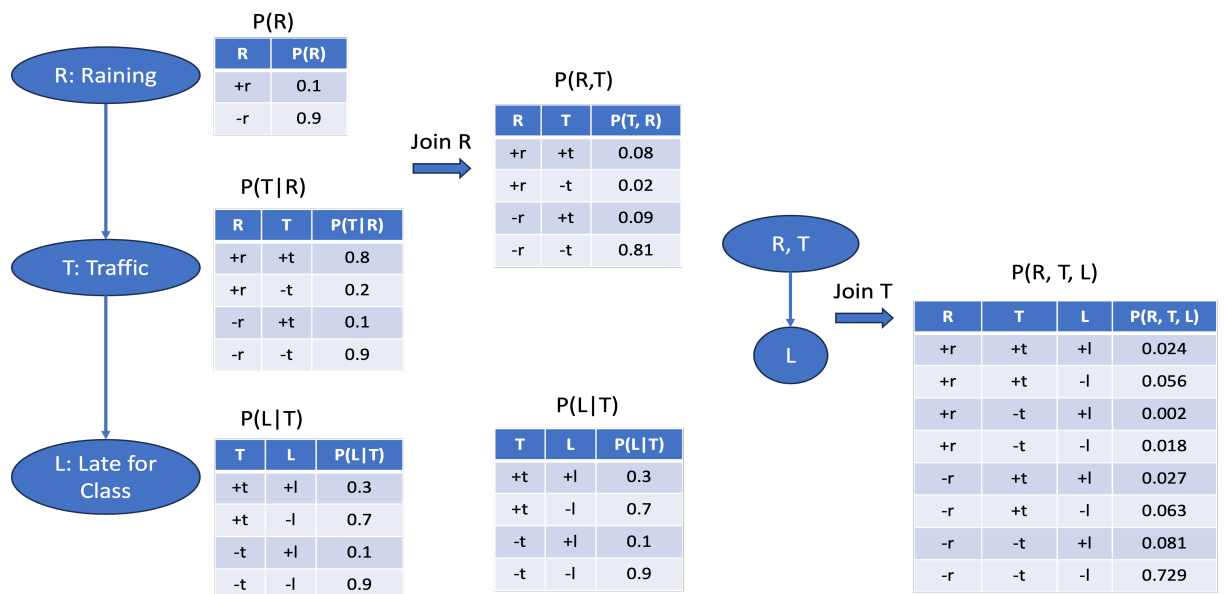


$$\begin{aligned}
 P(L) &=? \\
 &= \sum_{r,t} P(r, t, L) \\
 &= \sum_{r,t} P(r)P(t | r)P(L | t)
 \end{aligned} \tag{4.1}$$

- Tracked objects called factors
- Initial factors are local CPTs (one per node)
local CPTs are $P(R)$, $P(T|R)$, $P(L|T)$
- Any known values are selected
suppose we know $L = +l$, the initial factors are $P(R)$, $P(T|R)$, $P(+l|T)$
- **Operation1: Join Factors (on X)**
 - Combine factors just like a database join
 - Get all factors over the joining variable (get all factors that include variable X)
 - Build a new factor over the union of the variables involved.
 - Example1: join on R
It is like merging R and T into a mega variable (R, T)



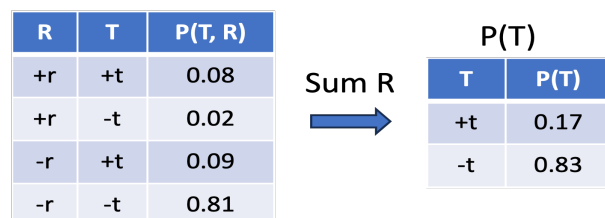
– Example2: Multiple join



• Operation2: Elimination (Marginalization)

- Take a factor and sum out a variable
- Shrinks a factor to a smaller one
- A projection operation
- Example1: Marginalization on R
Collapsing 2d data to 1d

$P(R, T)$



– Example2: Multiple elimination

R, T, L				T, L			L	
P(R, T, L)				P(L,T)			P(L)	
R	T	L	P(R, T, L)	T	L	P(L,T)	L	P(L)
+r	+t	+l	0.024	+t	+l	0.051	+l	0.134
+r	+t	-l	0.056	+t	-l	0.119	-l	0.886
+r	-t	+l	0.002	-t	+l	0.083		
+r	-t	-l	0.018	-t	-l	0.747		
-r	+t	+l	0.027					
-r	+t	-l	0.063					
-r	-t	+l	0.081					
-r	-t	-l	0.729					

4.4 Variable Elimination: Marginalize Early!

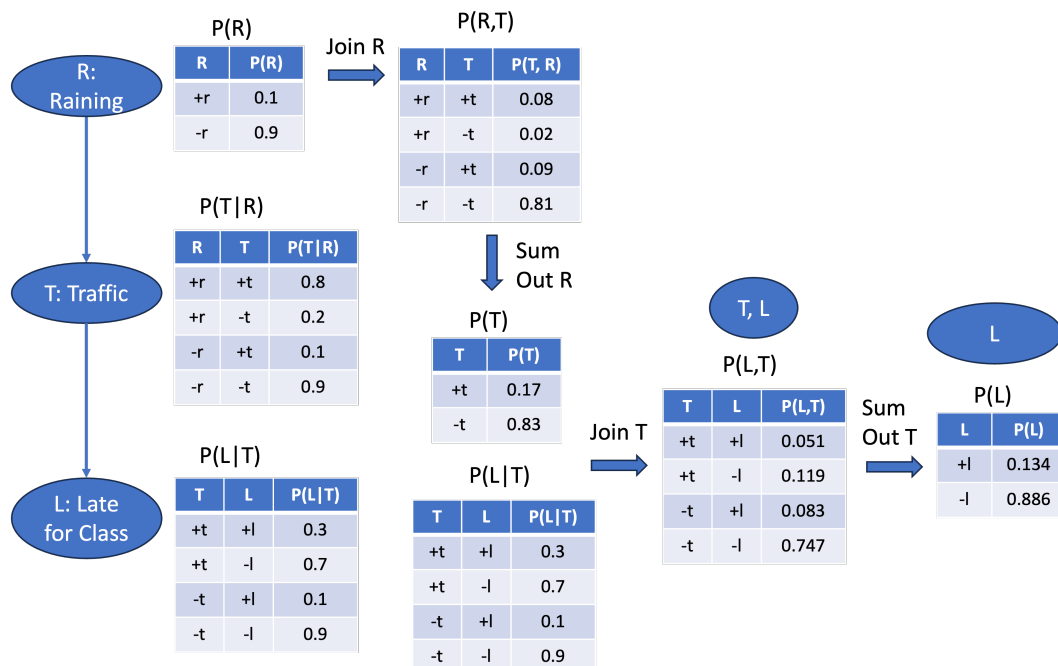
You cannot marginalize a variable until you join all factors that include that variable. Except for query variable, which is to leave to the end. For evidential variables, we select them at the first step.

Inference by Enumeration

$$P(L) = \sum_t \sum_r P(L|t) \underbrace{P(r)P(t|r)}_{\text{Join on } r} \underbrace{}_{\text{Join on } t} \underbrace{}_{\text{Eliminate on } r} \underbrace{}_{\text{Eliminate on } t}$$

Variable Elimination

$$P(L) = \sum_t P(L|t) \sum_r \underbrace{P(r)P(t|r)}_{\text{Join on } r} \underbrace{}_{\text{Eliminate on } r} \underbrace{}_{\text{Join on } t} \underbrace{}_{\text{Eliminate on } t}$$



4.5 Evidence

- If evidence, start with factors that select that evidence
 - If there is no evidence, start with following initial factors


P(R)		P(T R)			P(L T)		
R	P(R)	R	T	P(T R)	T	L	P(L T)
+r	0.1	+r	+t	0.8	+t	+l	0.3
-r	0.9	+r	-t	0.2	+t	-l	0.7
		-r	+t	0.1	-t	+l	0.1
		-r	-t	0.9	-t	-l	0.9

- If there is evidence ($R = +r$), initial factors become

P(+r)		P(T +r)			P(L T)		
R	P(R)	R	T	P(T R)	T	L	P(L T)
+r	0.1	+r	+t	0.8	+t	+l	0.3
		+r	-t	0.2	+t	-l	0.7
					-t	+l	0.1
					-t	-l	0.9

- We eliminate all vars other than query + evidence
 - Results will be a selected join of query and evidence, to get our answer, normalize it.
- For $P(L, +r)$, we would end up with:

P(+r, L)			P(L +r)	
R	L	P(+r, L)		
+r	+l	0.026		
+r	-l	0.074		

Normalize 

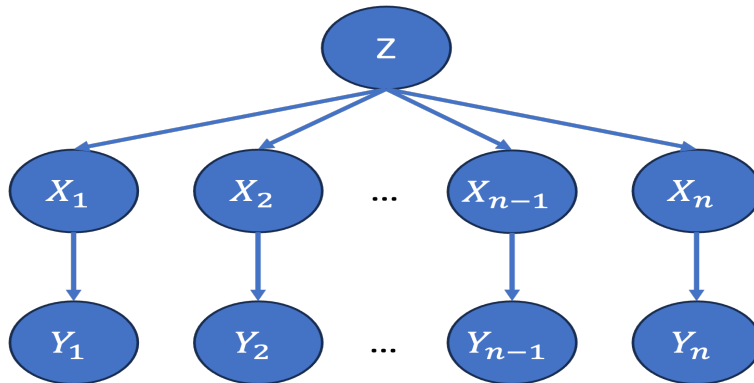
L	P(L +r)
+l	0.26
-l	0.74

4.6 General Variable Elimination

- Query: $P(Q \mid E_1 = e_1, \dots, E_k = e_k)$
- Start with initial factors: local CPTs initiated by evidence
- While there are still hidden variables:
 - Pick a hidden variable H
 - Join all factors mentioning H
 - Eliminate (sum out) H
- Join all remaining factors and normalize

4.7 Variable Elimination Ordering

For the query $P(X_n \mid y_1, y_2, \dots, y_n)$ work through the following two different orderings: $Z, X_1, X_2, \dots, X_{n-1}$ and $X_1, X_2, \dots, X_{n-1}, Z$



- If we start with X_1 , we need to join $P(Y_1 \mid X_1), P(X_1 \mid Z)$.
- If we start with Z , we need to join $P(Z), P(X_1 \mid Z), P(X_2 \mid Z), \dots, P(X_{n-1} \mid Z)$ (a giant factor).
- To find the best ordering: NP-hard

4.8 Variable Elimination Complexity

- The computational and space complexity of variable elimination is determined by the largest factor.
- The elimination ordering could greatly affect the size of the largest factor. Eg. in previous example, 2^n vs 2
- Factor size: d^k entries calculated for a factor over k variables with domain size d .
- Does there always exist an ordering that only results in small factors? NO.
- Worst case: running time exponential in the size of the BN.
- Inference in BN is NP-hard. No known efficient probabilistic inference in general.
- Good situation: Polytree
 - A polytree is a directed graph with no undirected cycles.
 - For polytrees, you can always find an ordering that is efficient.
 - Choose set of variables that if removed only a polytree remains.

Chapter 5

Sampling

5.1 Approximate Inference: Sampling

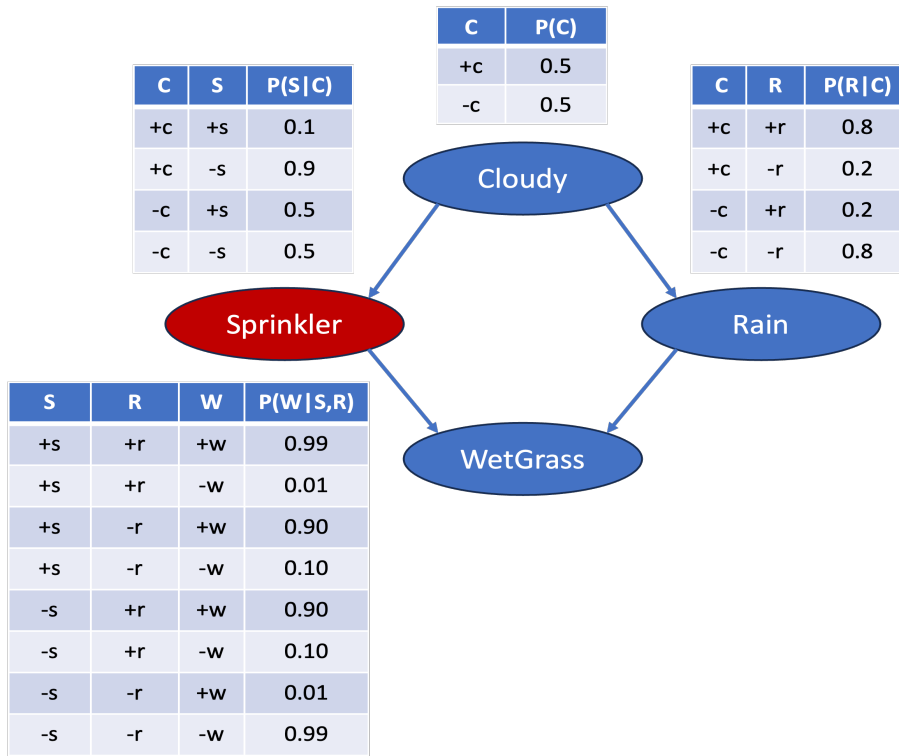
Trade off: Computation versus Accuracy.

- Sampling is a lot like repeated simulation. Sampling could be used for learning (we do not know the real probability distribution, but we could interact with the world to learn). However, here we use sampling for inference. We know probabilities and we sample for faster computation.
- Why sample?
 - Learning: get samples from a distribution you do not know.
 - Inference: get a sample is faster than computing the right answer.
- Basic Idea
 - Draw N samples from a sampling distribution S
 - Compute an approximate posterior probability
 - Show this converges to the true probability P .
- Sampling from Given Distribution
 - Step1: given sample u from uniform distribution over $[0, 1)$. eg: `random()` in python
 - Step2: convert this sample u into an outcome for the given distribution by having each target outcome associated with a sub-interval of $[0, 1)$ with sub-interval size equal to probability of the outcome.

C	P(C)
red	0.6
green	0.1
blue	0.3

$0 \leq u < 0.6$, $C = \text{red}$
 $0.6 \leq u < 0.7$, $C = \text{green}$
 $0.7 \leq u < 1$, $C = \text{blue}$

5.2 Prior Sampling



- For $i = 1, 2, \dots, n$,
sample x_i from $P(X_i \mid \text{Parents}(X_i))$.

Return (x_1, x_2, \dots, x_n)

- This process generates samples with probability

$$S_{PS}(x_1, x_2, \dots, x_n) = \prod_{i=1}^n P(x_i \mid \text{Parents}(X_i)) = P(x_1, x_2, \dots, x_n)$$

- Let the number of samples of an event be $N_{PS}(x_1, x_2, \dots, x_n)$
- Then,

$$\begin{aligned} \lim_{x \rightarrow \infty} P(x_1, \dots, x_n) &= \lim_{x \rightarrow \infty} \frac{N_{PS}(x_1, x_2, \dots, x_n)}{N} \\ &= S_{PS}(x_1, x_2, \dots, x_n) \\ &= P(x_1, x_2, \dots, x_n) \end{aligned} \quad (5.1)$$

- The sampling procedure is consistent (the sampling distribution approximate the actual distribution).
- Example

- We will get a bunch of samples from the BN
 - +c, -s, +r, +w
 - +c, +s, +r, +w
 - c, +s, +r, -w

+C, -S, +I, +W
-C, -S, -I, +W

- If we want to know $P(W)$
 - * We have counts $\langle +w:4, -w:1 \rangle$
 - * Normalize to get $P(W) = \langle +w:0.8, -w:0.2 \rangle$
 - * This will get closer to the true distribution with more samples.
 - * What about $P(C|-I,-W)$? we don't have the sample, therefore cannot answer the question.
 - * Fast: can use fewer samples if less time

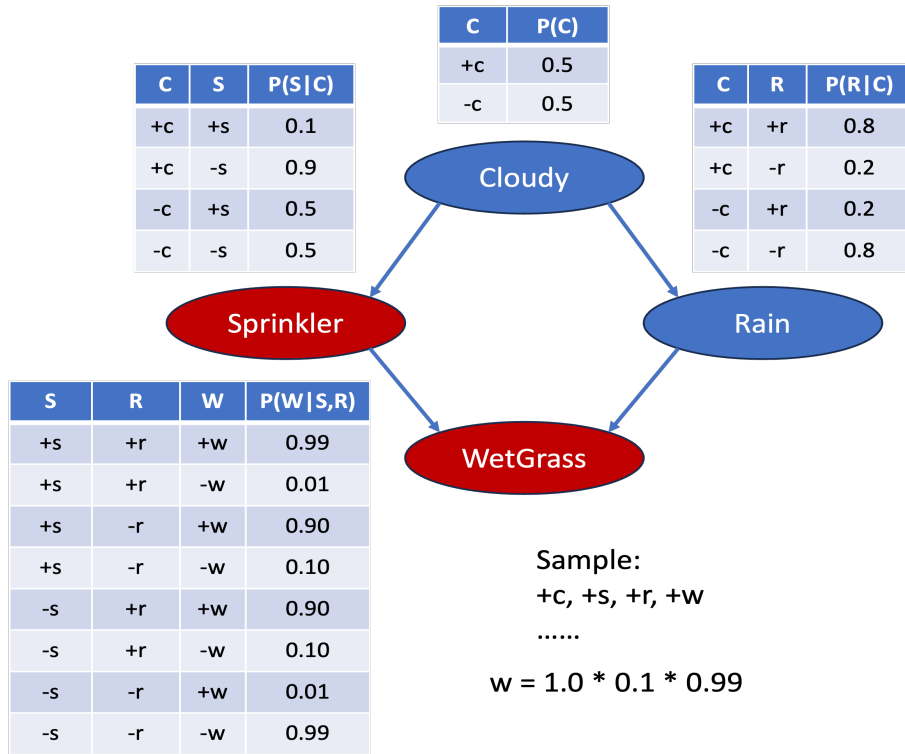
5.3 Rejection Sampling

- We get a bunch of samples from the BN
 - +C, -S, +I, +W
 - +C, +S, +I, +W
 - C, +S, +I, -W
 - +C, -S, +I, +W
 - C, -S, -I, +W
- Lets say we want $P(C|+S)$
- Tally C outcomes, but reject samples which do not have $S = +s$. This is called rejection sampling
- It is also consistent for conditional probabilities
- For $i = 1, 2, \dots, n$,
 - sample x_i from $P(X_i \mid \text{Parents}(X_i))$.
 - if x_i not consistent with evidence,
reject: return no sample generated in this cycle.

Return (x_1, x_2, \dots, x_n)

5.4 Likelihood Sampling

- Problems with rejection sampling
 - If evidence is unlikely, reject lots of samples
 - Evidence is not exploited
- Idea: fix evidence variables and sample the rest
 - Problem: sample distribution not consistent (eg, we assume that all sampling objects are blue. The probability of $P(\text{blue})$ changed to 1.)
 - Solution: weight by probability of evidence given parents.



- Algorithm

- input: evidence initiation
- $w = 1.0$
- for $i = 1, 2, \dots, n$,
 - * if X_i is an evidence variable
 - x_i = observation x_i for X_i
 - Set $w = w * P(x_i | Parents(X_i))$
 - * else:
 - sample x_i from $P(X_i | Parents(X_i))$.
- Return $(x_1, x_2, \dots, x_n), W$

- Sampling distribution if z sampled and e fixed evidence

$$S_{WS}(z, e) = \prod_{i=1}^l P(Z_i | Parents(Z_i))$$

- Now, samples have weights

$$w(z, e) = \prod_{i=1}^m P(e_i | Parents(E_i))$$

- Together, weighted sampling distribution is consistent

$$S_{WS}(z, e) * w(z, e) = \prod_{i=1}^l P(Z_i | Parents(Z_i)) * \prod_{i=1}^m P(e_i | Parents(E_i))$$

$$S_{WS}(z, e) * w(z, e) = P(z, e)$$

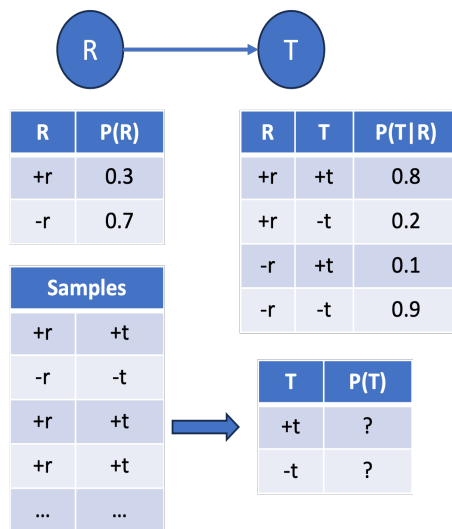
- Likelihood weighting is good

- We have taken evidence into account as we generate the sample

- Most of our samples will reflect the state of the world suggested by evidence.
- Likelihood weighting does not solve all problems. Likelihood works well when evidence is at the top. Evidence influences the choices of downstream variables, but not upstream ones. However, if the evidence is at the bottom, for instance, suppose the evidence is we know john called. Likelihood weighting is good at simulating consequences of evidence, but not very good at simulating the causes the evidence.

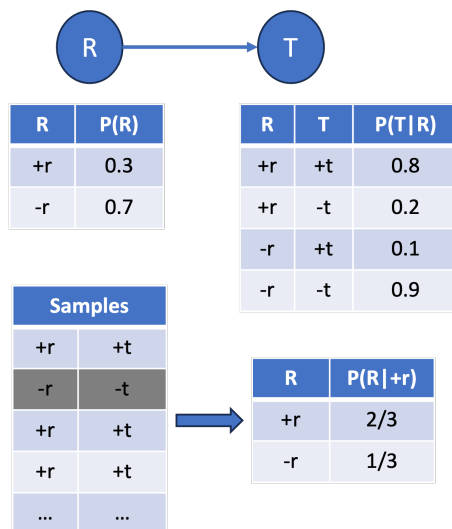
5.5 Example

- Prior Sampling



In the sample, $p(+t) = \frac{3}{4}$, it might not be equal to true $p(+t)$. However, if we get 4000 samples, these two numbers will be very close.

- Rejection Sampling



In the sample, $p(+t) = \frac{3}{4}$, We reject one sample $T=-t$.

- Likelihood Weighting

5.6 Gibbs Sampling

- We do not sample from top to the bottom. Instead, we start with a complete assignment. We take into account evidence both upstream and downstream.
- Procedure: Keep track of a full instantiation $x_1, x_2, x_3, \dots, x_n$. Start with an arbitrary instantiation consistent with the evidence. Sample one variable at a time, conditioned on all the rest, but keep evidence fixed. Keep repeating this for a long time.
- Rationale: both upstream and downstream variables condition on evidence.
- In contrast: likelihood weighting only conditions on upstream evidence, and hence weights obtained in likelihood weighting can sometimes be very small.

