CS4710: Artificial Intelligence
Reasoning Under Uncertainty

Part 1: Most interesting AI problems are probabilistic in nature. Let's review probability and apply it to some AI problems.

Topics

- Review of probability
  - Random Variables
  - Joint Probability Distributions
  - Etc.
- Bayesian Networks
  - What is a Bayesian Network?
  - How to use one to make probabilistic inferences?
  - Etc.

Introduction

Introduction

Suppose you are trying to determine if a patient has inhalational anthrax. You observe the following symptoms:

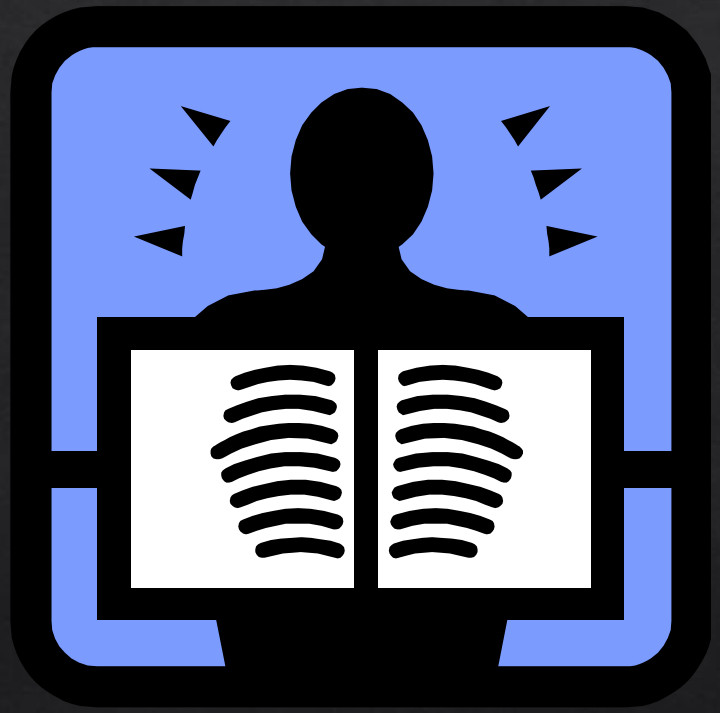
- The patient has a cough
- The patient has a fever
- The patient has difficulty breathing

Introduction

You would like to determine how likely the patient is infected with inhalational anthrax given that the patient has a cough, a fever, and difficulty breathing

We are not 100% certain that the patient has anthrax because of these symptoms. We are dealing with uncertainty!

Introduction



Now suppose you order an x-ray and observe that the patient has a wide mediastinum.

Your belief that that the patient is infected with inhalational anthrax is now much higher.

Introduction

- In the previous slides, what you observed affected your belief that the patient is infected with anthrax
- This is called **reasoning with uncertainty**
- Wouldn't it be nice if we had some methodology for reasoning with uncertainty? Why in fact, we do…

Probability Review

Probability Primer: Random Variables

- A **random variable** is the basic element of probability
- Refers to an event and there is some degree of uncertainty as to the outcome of the event
- For example, the random variable *A* could be the event of getting a heads on a coin flip

Boolean Random Variables

- We will start with the simplest type of random variables – Boolean ones
- Take the values *true* or *false*
- Think of the event as occurring or not occurring
- Examples (Let A be a Boolean random variable):

  A = Getting heads on a coin flip

  A = It will rain today

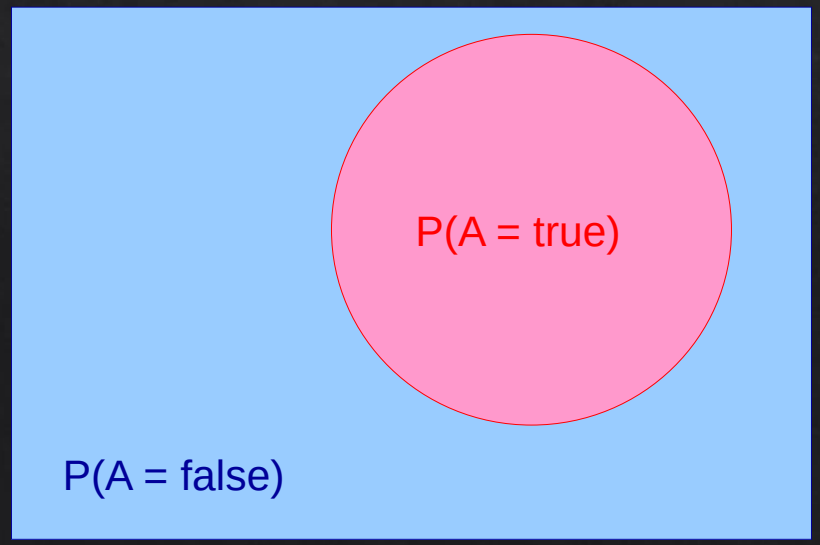
  A = The Cubs win the World Series

Probabilities

We will write either *P(A)* or *P(A = true)* to mean the probability that *A = true*.

What is probability? It is the relative frequency with which an outcome would be obtained if the process were repeated a large number of times under similar conditions*

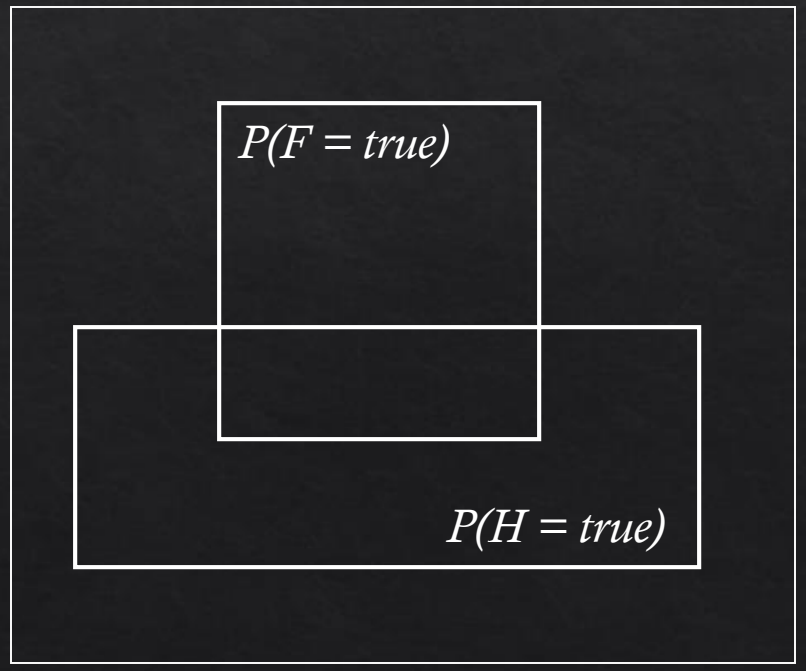
The sum of the red and blue areas is 1



*Ahem…there's also the Bayesian definition which says probability is your degree of belief in an outcome

Conditional Probability

- P(A = *true* | B = *true*) = Out of all the outcomes in which B is true, how many also have A equal to true
- Read this as: "Probability of A conditioned on B" or "Probability of A given B"



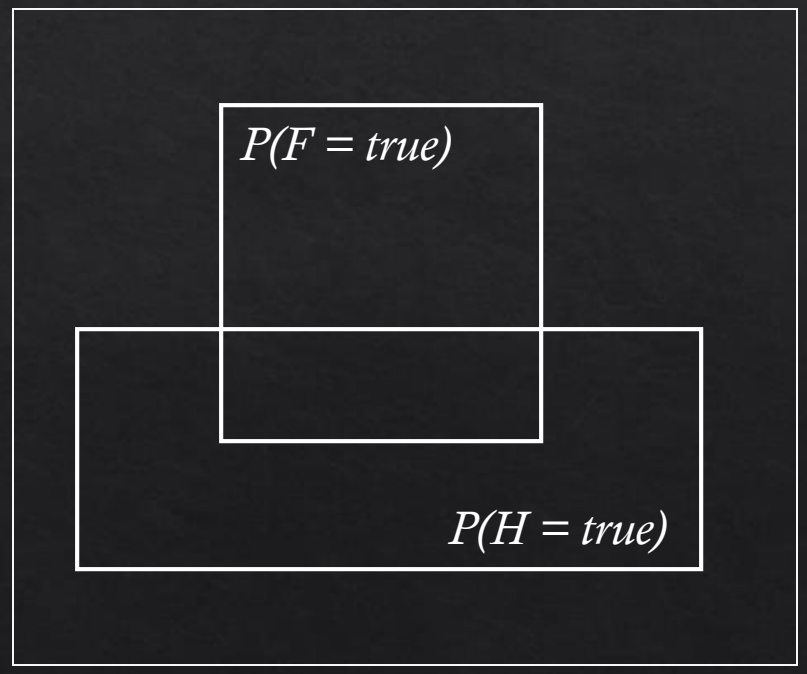
H = "Have a headache"
F = "Coming down with Flu"

P(H = *true*) = 1/10
P(F = *true*) = 1/40
P(H = *true* | F = *true*) = 1/2

"Headaches are rare and flu is rarer, but if you're coming down with flu there's a 50-50 chance you'll have a headache."

The Joint Probability Distribution

- We will write *P(A = true, B = true)* to mean "the probability of *A = true* **and** *B = true*"
- Notice that:



$$P(H=true \mid F=true)$$

$$= \frac{\text{Area of "H and F" region}}{\text{Area of "F" region}}$$

$$= \frac{P(H = true, F = true)}{P(F = true)}$$

In general, $P(X \mid Y) = P(X, Y) / P(Y)$

The Joint Probability Distribution

- Joint probabilities can be between any number of variables

  eg. *P(A = true, B = true, C = true)*
- For each combination of variables, we need to say how probable that combination is
- The probabilities of these combinations need to sum to 1

A	B	C	P(A,B,C)
false	false	false	0.1
false	false	true	0.2
false	true	false	0.05
false	true	true	0.05
true	false	false	0.3
true	false	true	0.1
true	true	false	0.05
true	true	true	0.15

Sums to 1

The Joint Probability Distribution

- Once you have the joint probability distribution, you can calculate any probability involving *A*, *B*, and *C*
- Note: May need to use marginalization and Bayes rule, (both of which are not discussed in these slides)

A	B	C	P(A,B,C)
false	false	false	0.1
false	false	true	0.2
false	true	false	0.05
false	true	true	0.05
true	false	false	0.3
true	false	true	0.1
true	true	false	0.05
true	true	true	0.15

Examples of things you can compute:

- *P(A=true)* = sum of *P(A,B,C)* in rows with *A=true*
- *P(A=true, B = true | C=true)* =

  P(A = true, B = true, C = true) / P(C = true)

The Problem with the Joint Distribution

- Lots of entries in the table to fill up!
- For k Boolean random variables, you need a table of size 2^k
- How do we use fewer numbers? Need the concept of independence

A	B	C	P(A,B,C)
false	false	false	0.1
false	false	true	0.2
false	true	false	0.05
false	true	true	0.05
true	false	false	0.3
true	false	true	0.1
true	true	false	0.05
true	true	true	0.15

Independence

Variables A and B are independent if any of the following hold:

- $P(A,B) = P(A)\,P(B)$
- $P(A \mid B) = P(A)$
- $P(B \mid A) = P(B)$

This says that knowing the outcome of A does not tell me anything new about the outcome of B.

Independence

How is independence useful?

- Suppose you have n coin flips and you want to calculate the joint distribution $P(C_1, \ldots, C_n)$
- If the coin flips are not independent, you need 2^n values in the table
- If the coin flips are independent, then

$$P(C_1,\ldots, C_n) = \prod_{i=1}^{n} P(C_i)$$

Each $P(C_i)$ table has 2 entries and there are n of them for a total of $2n$ values

Conditional Independence

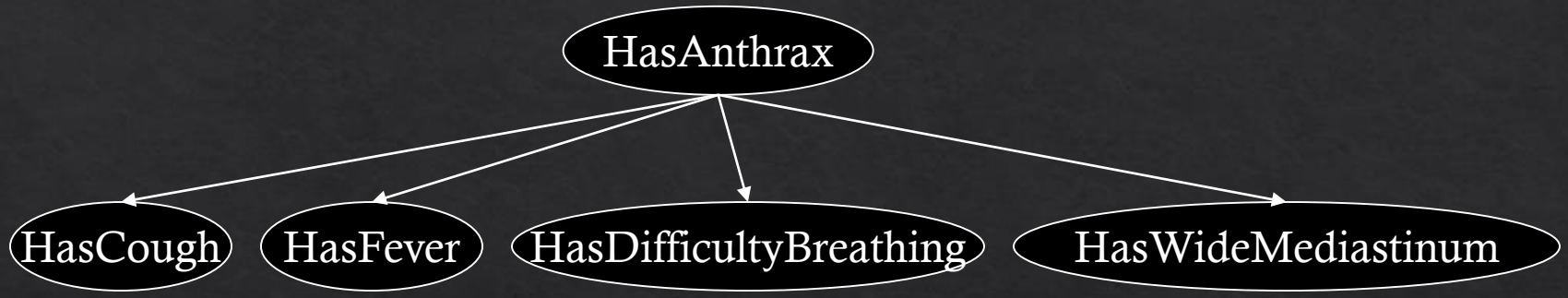
Variables A and B are conditionally independent given C if any of the following hold:

- $P(A, B \mid C) = P(A \mid C)\, P(B \mid C)$
- $P(A \mid B, C) = P(A \mid C)$
- $P(B \mid A, C) = P(B \mid C)$

Knowing C tells me everything about B. I don't gain anything by knowing A (either because A doesn't influence B or because knowing C provides all the information knowing A would give)

Bayesian Networks

Bayesian Networks

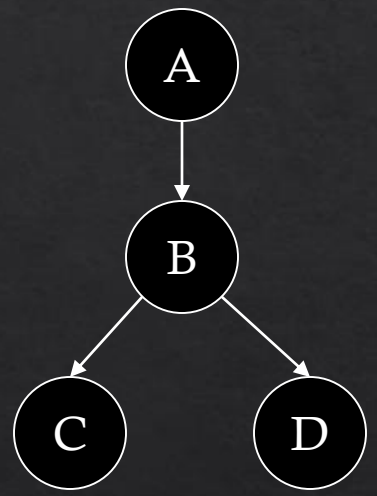


- In the opinion of many AI researchers, Bayesian networks are the most significant modern contribution in AI
- They are used in many applications eg. spam filtering, speech recognition, robotics, diagnostic systems and even syndromic surveillance

A Bayesian Network

A Bayesian network is made up of:

1. A Directed Acyclic Graph



2. A set of tables for each node in the graph

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

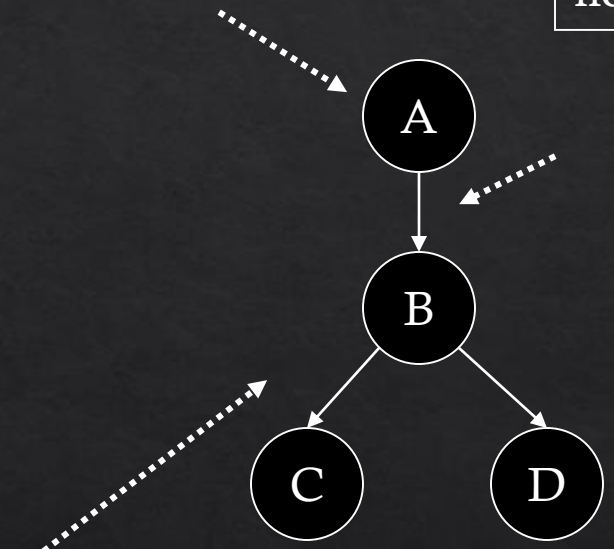
B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1

A Directed Acyclic Graph

Each node in the graph is a random variable

A node *X* is a parent of another node *Y* if there is an arrow from node *X* to node *Y* eg. *A* is a parent of *B*



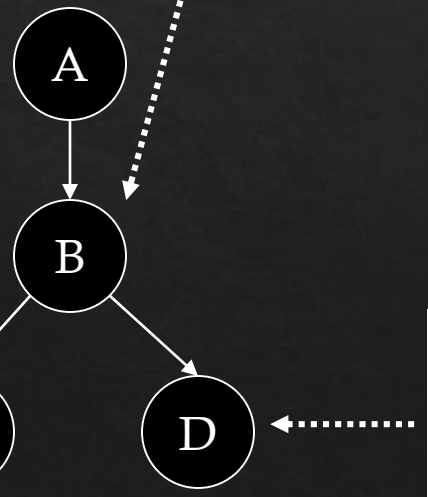
Informally, an arrow from node *X* to node *Y* means *X* has a direct influence on *Y*

A Set of Tables for Each Node

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Each node X_i has a conditional probability distribution $P(X_i \mid \text{Parents}(X_i))$ that quantifies the effect of the parents on the node

The parameters are the probabilities in these conditional probability tables (CPTs)

A Set of Tables for Each Node

Conditional Probability Distribution
for C given B

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1

For a given combination of values of the parents (B in this example), the entries for P(C=true | B) and P(C=false | B) must add up to 1
eg. P(C=true | B=false) + P(C=false |B=false)=1

If you have a Boolean variable with k Boolean parents, this table has 2^{k+1} probabilities (but only 2^k need to be stored)

Bayesian Networks

Two important properties:

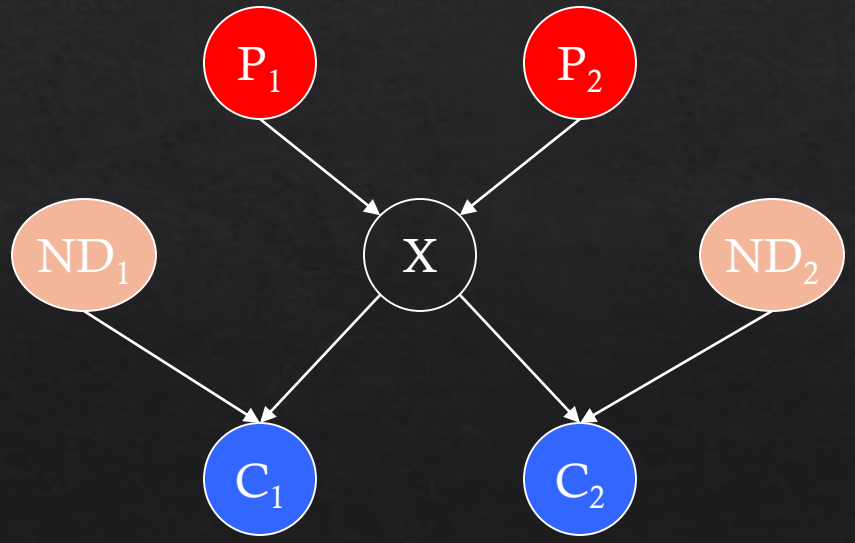
1. Encodes the conditional independence relationships between the variables in the graph structure

2. Is a compact representation of the joint probability distribution over the variables

Conditional Independence

The Markov condition: given its parents (P_1, P_2),

a node (X) is conditionally independent of its non-descendants (ND_1, ND_2)



The Joint Probability Distribution

Due to the Markov condition, we can compute the joint probability distribution over all the variables $X_1, \ldots, X_n$ in the Bayesian net using the formula:

$$P(X_1 = x_1, \ldots, X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i \mid Parents(X_i))$$

Where Parents(X_i) means the values of the Parents of the node X_i with respect to the graph

Using a Bayesian Network Example

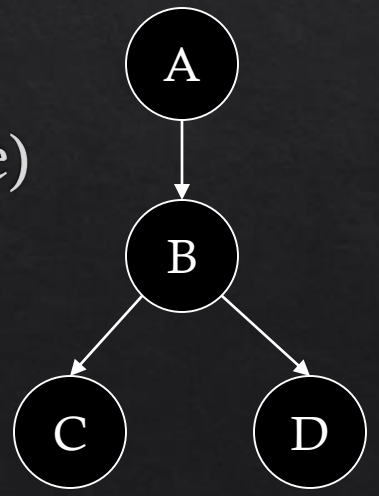
Using the network in the example, suppose you want to calculate:

P(A = true, B = true, C = true, D = true)

= P(A = true) * P(B = true | A = true) *

P(C = true | B = true) P(D = true | B = true)

= (0.4)*(0.3)*(0.1)*(0.95)



Using a Bayesian Network Example

Using the network in the example, suppose you want to calculate:

P(A = true, B = true, C = true, D = true)

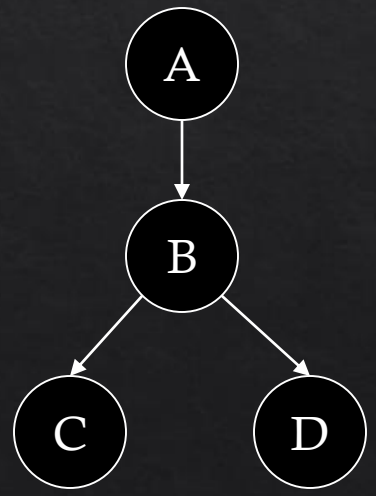
= P(A = true) * P(B = true | A = true) *

P(C = true | B = true) P(D = true | B = true)

= (0.4)*(0.3)*(0.1)*(0.95)

This is from the graph structure

These numbers are from the conditional probability tables

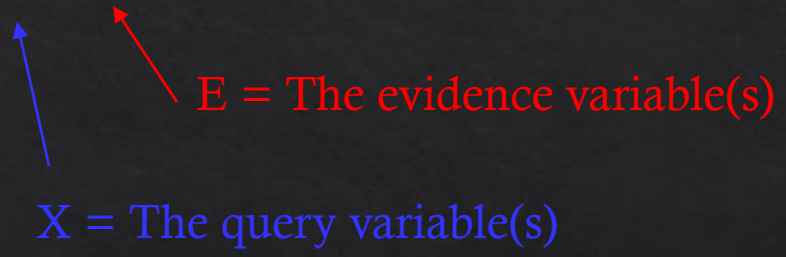


Inference

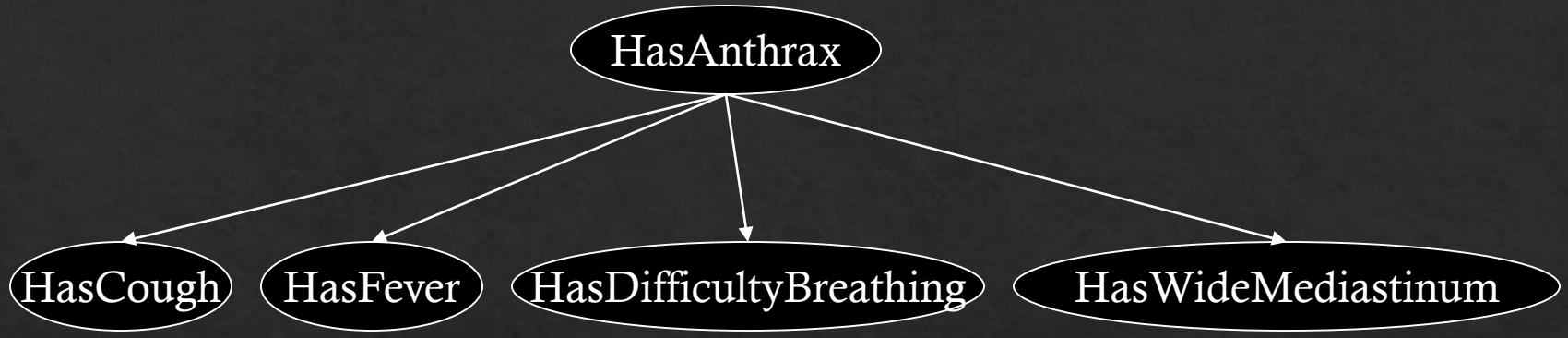
- Using a Bayesian network to compute probabilities is called inference

- In general, inference involves queries of the form:

  P(X | E)



Inference



- An example of a query would be:

  P(*HasAnthrax = true* | *HasFever = true*, *HasCough = true*)

- Note: Even though *HasDifficultyBreathing* and *HasWideMediastinum* are in the Bayesian network, they are not given values in the query (ie. they do not appear either as query variables or evidence variables)
- They are treated as unobserved variables

The Bad News

- Exact inference is feasible in small to medium-sized networks
- Exact inference in large networks takes a very long time
- We resort to approximate inference techniques which are much faster and give pretty good results

One last unresolved issue…

We still haven't said where we get the Bayesian network from. There are two options:

- Get an expert to design it
- Learn it from data

Bayesian Networks: Sampling

Approximate Inference

Sometimes, network is too big, so need to approximate query probabilities!

Simulation has a name: **Sampling**

Basic Idea:

- Draw N samples from a distribution S
- Computer approximate posterior probability based on observed values
- Show this converges to the true probability

Approximate Inference

Why sample?

Learning: get samples from a distribution you don't know

Inference: getting a sample is faster than computing the right answer

Approximate Inference

Simple Example:

You have a weighted coin and you want to know what the probabilities are.

Solution:

Flip the coin N times

count number of head and tails

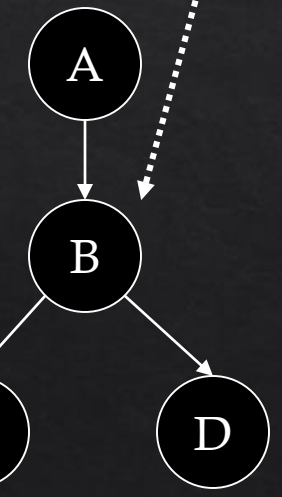
probabilities are (heads / N) and (tails / N)

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

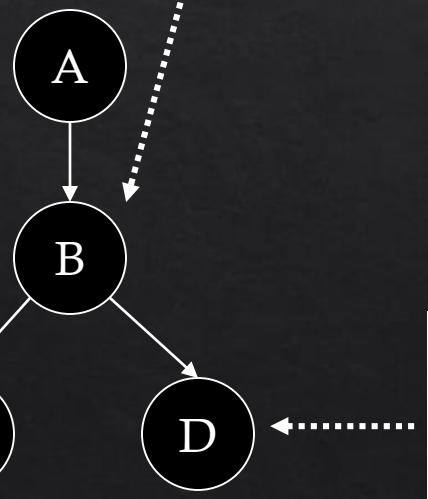
* Topological sort is useful here!

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

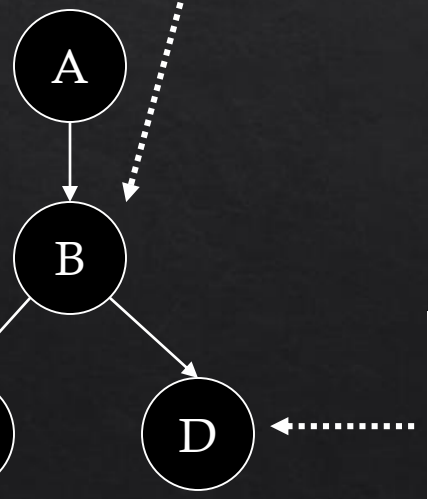
* Topological sort is useful here!

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

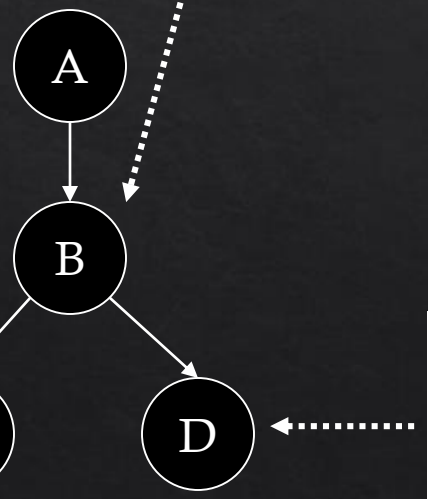
* Topological sort is useful here!

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

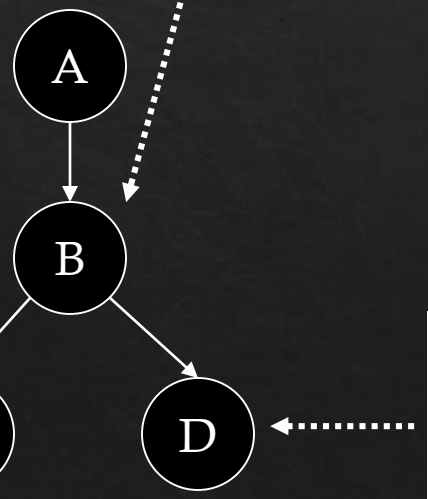
* Topological sort is useful here!

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

* Topological sort is useful here!

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1

B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

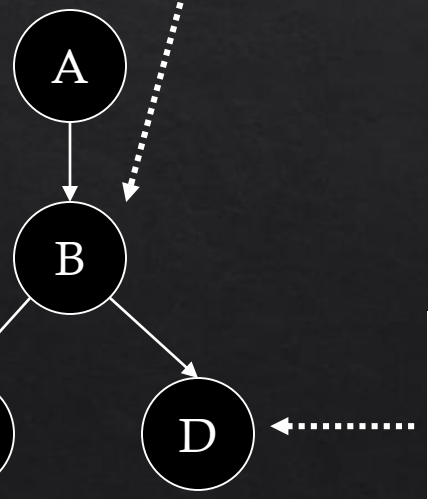
* Topological sort is useful here!

Prior Sampling

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Start at root nodes:

Flip a coin for A

Move to node B (flip a coin for B)

This depends on A's value

Repeat

* Topological sort is useful here!

Approximate Inference

So…we generate a bunch of samples from the network with:

$Sps(x1, \ldots, xn) = \prod_{i=1}^{n} P(zi \mid Parents(zi)) = P(x1, \ldots, xn)$

Let number of samples of a particular event be Nps(x1, … , xn)

Then:

$$\lim_{n\to\infty} P'(x1, \ldots, xn) = \lim_{n\to\infty} \frac{Nps(x1,\ldots,xn)}{N}$$

$$= \text{Sps(x1, … , xn)}$$

$$= \text{P(x1, … , xn)}$$

Approximate Inference

Example:

Suppose we want to know P(D) //probability the D node is true

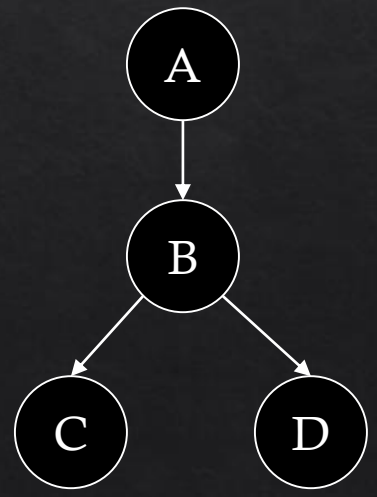
Process:

Sample the network N times

Count the number of times D ends up being true

divide this value by N

return



Approximate Inference

Another Example:

Suppose we want to know P(D | !A) //probability the D node is true given A is false

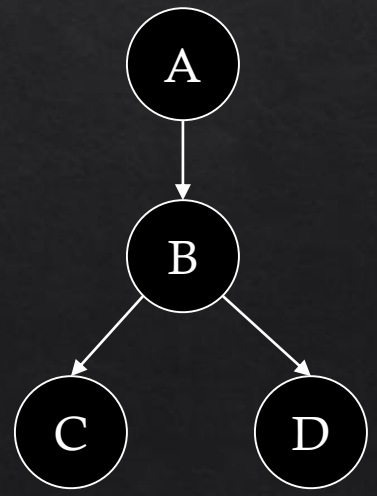
Process:

Sample the network N times

Count the number of times D ends up being true when A is false

divide this value by the total number of times A is false

return



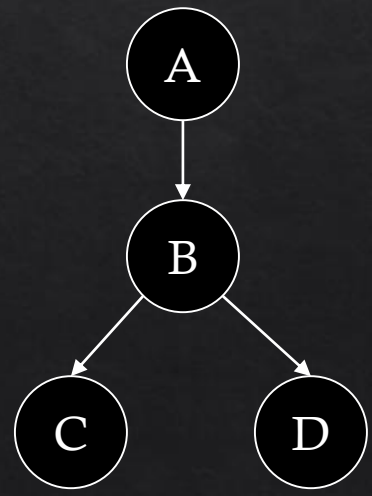
Approximate Inference

More samples we pull, the better our estimate!

Problem!

If we are computing the probability of a rare occurrence

We need LOTS of samples because that event is SO rare.



Bayesian Networks: Rejection Sampling

Rejection Sampling

Suppose we want to know P(D | !A) //probability the D node is true given A is false

Process:

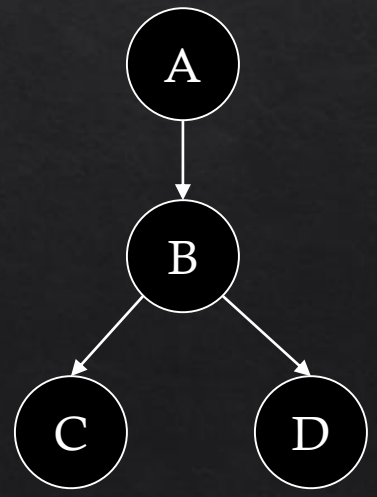
Sample the network N times

If A was not calculated to be false, throw out that sample (WHY!?)

Count the number of times D ends up being true when A is false

divide this value by N (number of times A was false)

return



This is called **Rejection Sampling**

Rejection Sampling

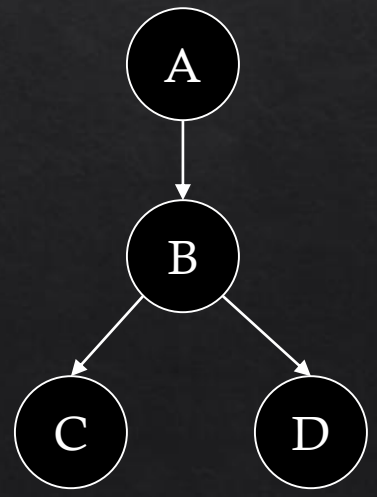
Problems:

If evidence is unlikely, you reject A LOT of samples

You don't exploit your evidence as you sample

We'll fix this in a minute!!

First, let's see an example!



Rejection Sampling

Suppose you have Two Cups:

Cup A: contains 1 penny and 2 quarters

Cup B: contains 2 pennies and 1 quarter

Suppose we flip a weighted coin (90,10) towards cup A to choose a cup

Then, we randomly pick a coin from that cup.

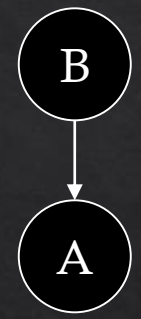
Suppose we want to calculate P(A | Q = false)

i.e., the probability Cup A is chosen given we didn't get a quarter

Bayesian Networks: Likelihood Weighting

Likelihood Weighting

Suppose we have a simple two node Bayesian network



B is burglary (P(B) = 0.01)

A is alarm went off

Consider calculating P(B | !A)

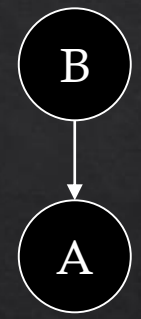
We are going to see A LOT of samples in which B is False

When B occasionally is true, P(!A) is going to be fairly rare as well

Need **many trials** to get a good estimate.

Likelihood Weighting

Suppose we have a simple two node Bayesian network



B is burglary

A is alarm went off

Consider calculating P(B | !A)

Idea: Fix our evidence variables to their observed values, then simulate the network

A lot of B=F and A=F values (don't have to reject samples at least)

Likelihood Weighting

Problem: Fixed our observed variables, but sample distribution not consistent!!

Solution: weight by probability of evidence given parents (which we know from table!)

So if we observe B=F and A=T we weight by P(A | B=F) from the Bayesian network table

Propagate through the network:

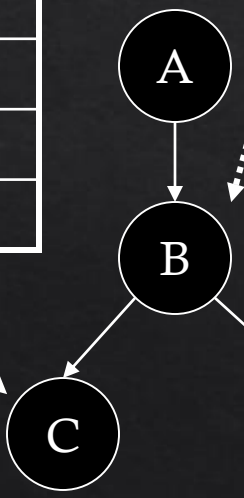
Sample a bunch of times as before

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

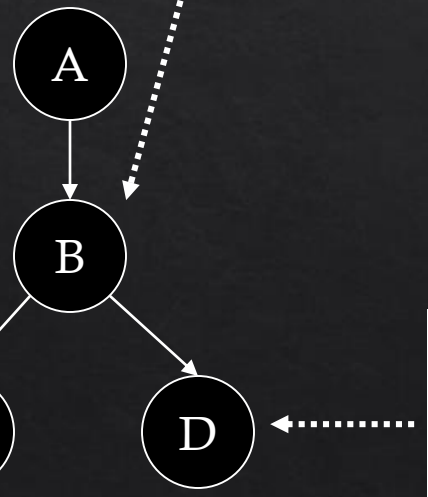
W = 1.0

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

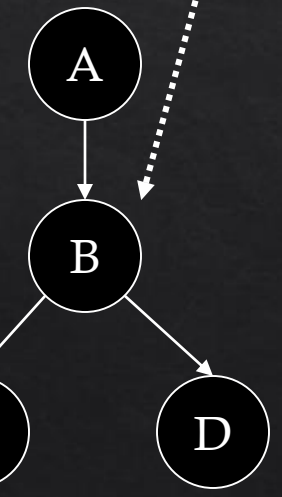
W = 1.0

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

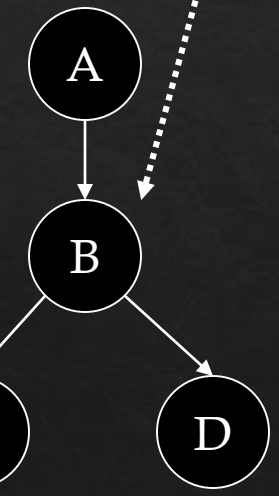
W = 1.0

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

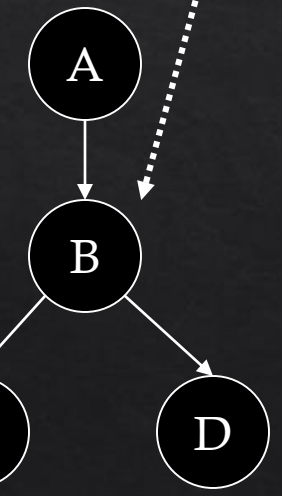
W = 1.0

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

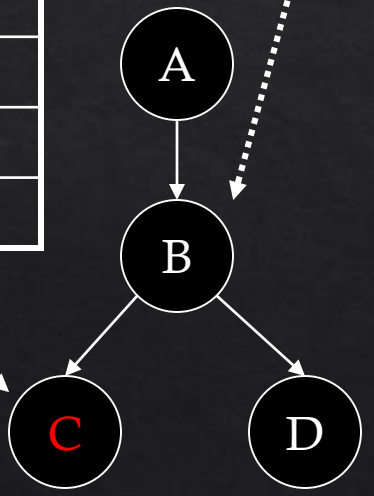
W = 1.0

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

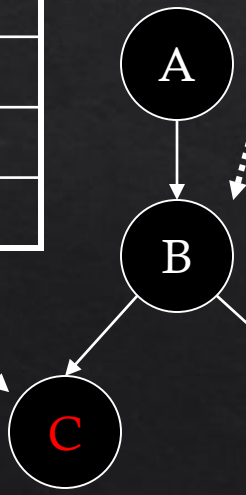
W = 1.0 * 0.4

Likelihood Weighting Example

A	P(A)
false	0.6
true	0.4

A	B	P(B|A)
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	P(C|B)
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	P(D|B)
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Suppose we are calculating:

P(A | C=F, D=T)

Keep a running weight:

W = 1.0 * 0.4 * 0.98

Likelihood Weighting Algorithm

Count = 0, Total = 0

For each sample we want to draw:

    W = 1.0 //weight

    For each node in network (topological order) N:

        if N is not observed, flip a coin to determine N based on parent's values (if parents exist)

        if N is observed, choose its given value always given parent's values (if parents exist)

            Multiply W by P(N | parents(N)) //the prob that was selected without coin flip

    If this sample is a "correct case": Count = Count + (1*W)

    Total = Total + (1*W)

Return Count / Total

Likelihood Weighting

Is ***Good***:

We have taken evidence into account as we generate samples

More of our samples will reflect the state of world suggested by evidence

Doesn't solve all problems:

Evidence influences the choice of downstream variables, but not upstream ones (root node probabilities never change)

So, this works best when evidence is high up in the network!

We would like to consider evidence when sampling every variable!

Likelihood Weighting

Suppose you have Two Cups:

Cup A: contains 1 penny and 2 quarters

Cup B: contains 2 pennies and 1 quarter

Suppose we flip a weighted coin (90,10) towards cup A to choose a cup

Then, we randomly pick a coin from that cup.

Suppose we want to calculate P(A | Q = false)

i.e., the probability Cup A is chosen given we didn't get a quarter (0.818181818181)