Announcement

- Homework 3
  - Due: April 7, 11:59pm
- Programming Assignment 3
  - Due: April 12, 11:59pm

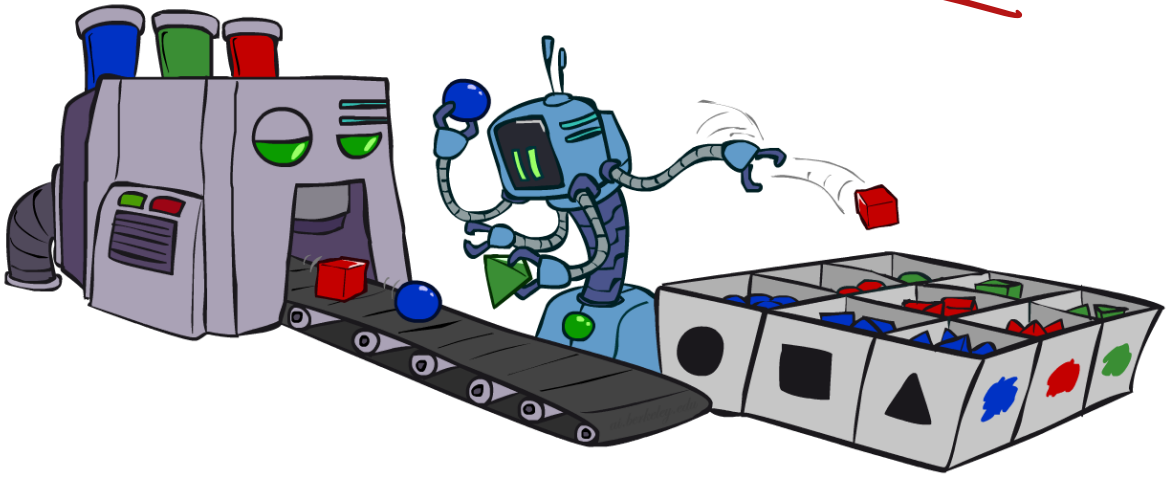
Announcement

- Midterm Review @April 12

- Midterm @April 14 (in class)
  - Location: TBA
  - “Introduction” – “Approximate inference on BN”
  - Format
    - Closed-book. You can bring an A4-size cheat sheet and nothing else.
    - Around 5 problems
  - Grade
    - 25% of the total grade

Sampling

Bayes Nets: Approximate Inference



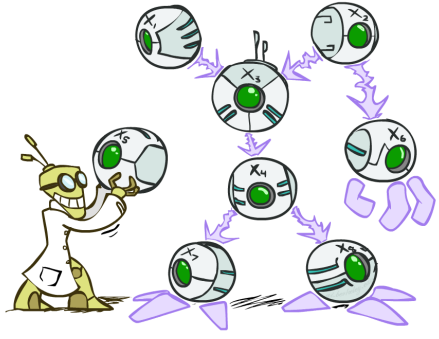
AIMA Chapter 14.5, PRML Chapter 11

[Adapted from slides by Dan Klein and Pieter Abbeel at UC Berkeley]

Bayes' Net Representation

- 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

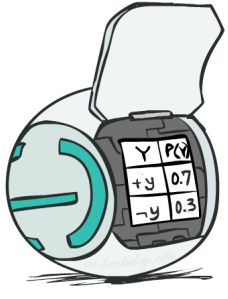
$$P(X|a_1 \dots a_n)$$



- Bayes' nets implicitly encode joint 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:

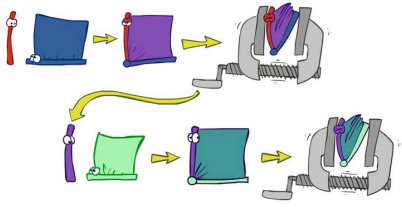
basic, review

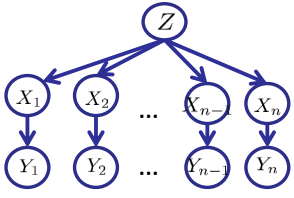
$$P(x_1, x_2, \dots x_n) = \prod_{i=1}^{n} P(x_i|parents(X_i))$$

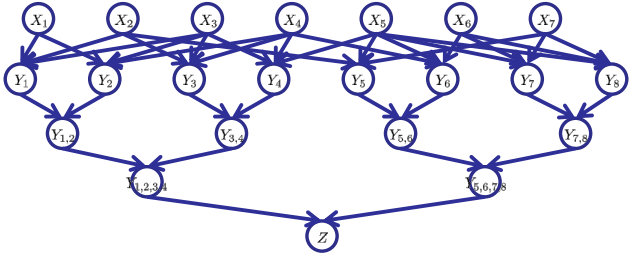


Variable Elimination

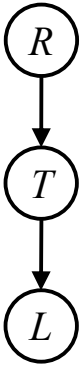
- Interleave joining and marginalizing
- d^k entries computed for a factor over k variables with domain sizes d expensive
- Ordering of elimination of hidden variables can affect size of factors generated
- Worst case: running time exponential in the size of the Bayes' net







Recap: Bayesian Inference (Exact)



$$P(L) = ?$$

- Inference by Enumeration

$$= \sum_t \sum_r P(L|t)P(r)P(t|r)$$

Join on r

Join on t

Eliminate r

Eliminate t

- Variable Elimination

$$= \sum_t P(L|t) \sum_r P(r)P(t|r)$$

Join on r

Eliminate r

Join on t

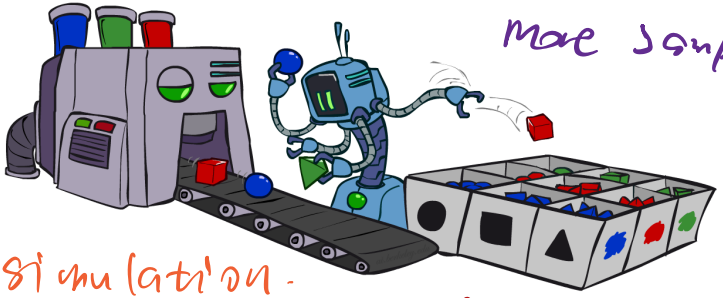
Eliminate t

Trade off: time - accuracy

Sampling

- Goal: probability *P*
- Basic idea
  - Draw *N* samples from a sampling distribution *S* defined by us.
  - Compute some quantity from the samples
  - Show this converges to the true probability *P* when *N* is inf
- Why sample?
  - Often very fast to get a decent approximate answer
  - The algorithms are very simple and general (easy to apply to fancy models)
  - They require very little memory (O(n))

Sampling on a known network is like a simulation.



more samples more time more accurate.

Sampling from a discrete distribution

- Sampling from given distribution
  - Step 1: Get sample *u* from uniform distribution over [0, 1)
    - Random() in many programing languages
  - Step 2: Convert this sample *u* into an outcome for the given distribution by associating each outcome *x* with a *P*(*x*)-sized sub-interval of [0,1)

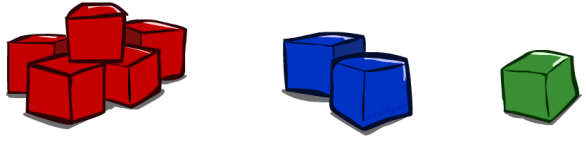
- Example

C	P(C)
red	0.6
green	0.1
blue	0.3

sample

$$0 \le u < 0.6, \rightarrow C = red$$
$$0.6 \le u < 0.7, \rightarrow C = green$$
$$0.7 \le u < 1, \rightarrow C = blue$$

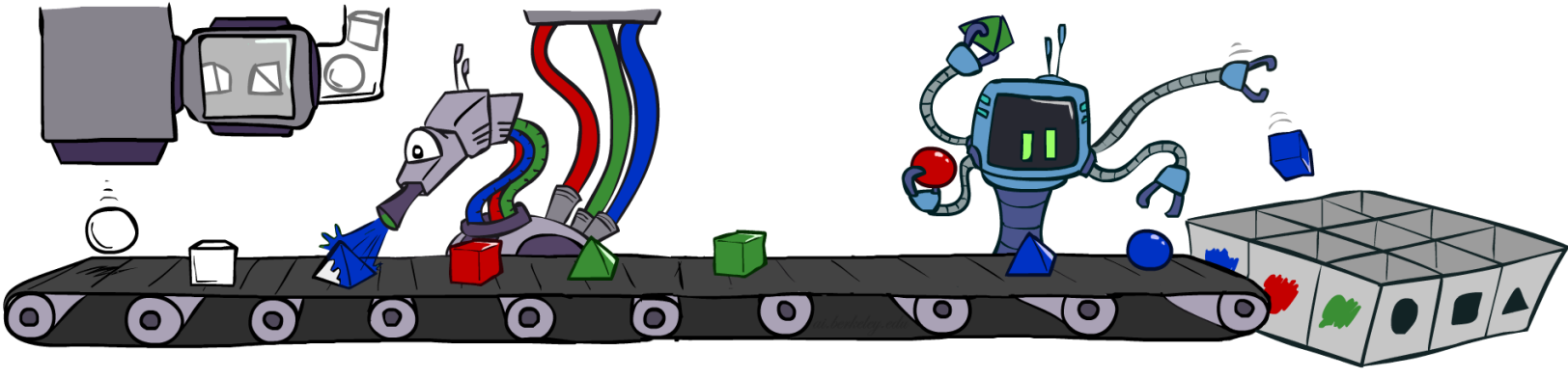
  - If random() returns *u* = 0.83, then our sample is *C* = blue
  - E.g, after sampling 8 times:



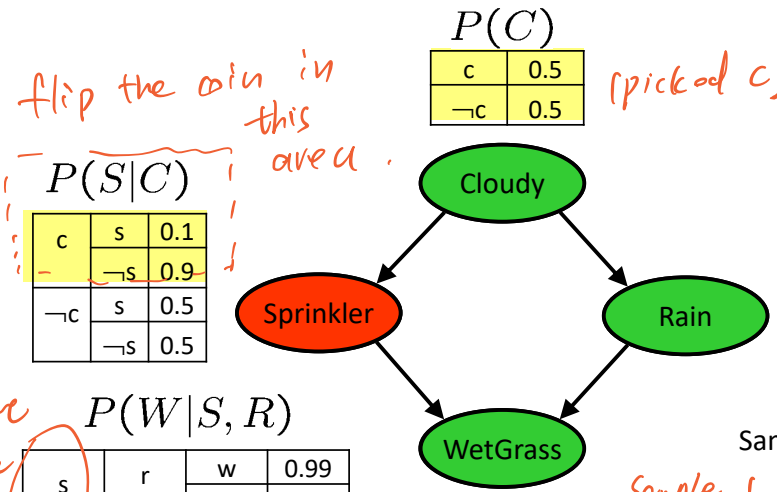
Sampling in Bayes Nets

- Prior Sampling
- Rejection Sampling
- Likelihood Weighting
- Gibbs Sampling

Prior Sampling



Prior Sampling



$P(C)$

c	0.5
¬c	0.5

flip the coin in this area.

(picked c)

$P(S|C)$

c	s	0.1
	¬s	0.9
¬c	s	0.5
	¬s	0.5

$P(R|C)$

c	r	0.8
	¬r	0.2
¬c	r	0.2
	¬r	0.8

$P(W|S,R)$

s	r	w	0.99
		¬w	0.01
	¬r	w	0.90
		¬w	0.10
¬s	r	w	0.90
		¬w	0.10
	¬r	w	0.01
		¬w	0.99

Son the overside

picked

Samples:

Sample 1: c, ¬s, r, w

Sample 2: ¬c, s, ¬r, w

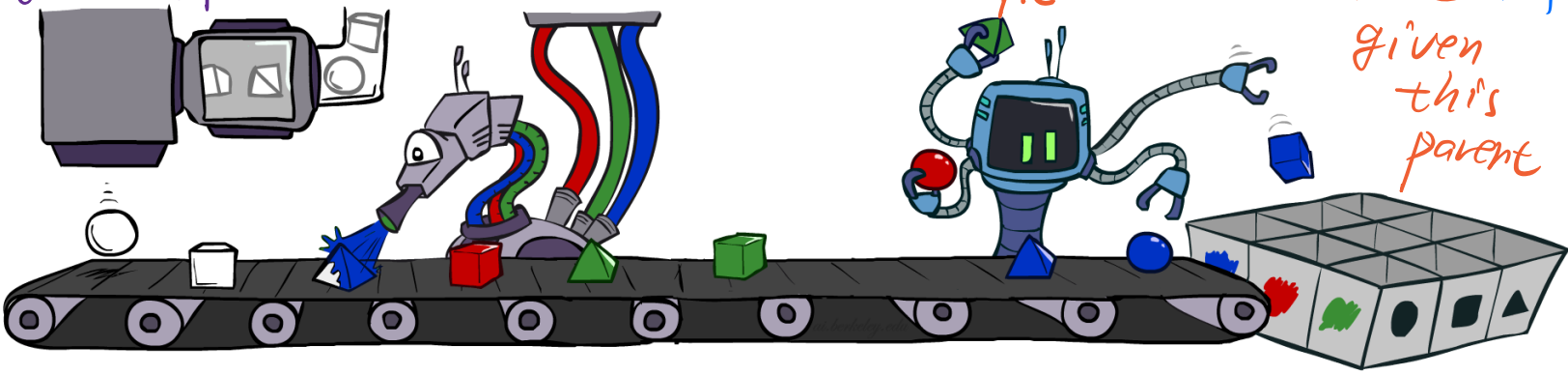
...

when π. find the prob of a certain joint distribution outcome

record samples

Prior Sampling

- For i=1, 2, …, n (in topological order)
  - Sample X_i from $P(X_i \mid parents(X_i))$
- Return $(x_1, x_2, \ldots, x_n)$



Prior Sampling

- This process generates samples with probability:

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

...i.e. the BN's joint probability

- Let the number of samples of an assignment be $N_{PS}(x_1 \dots x_n)$
- So $\hat{P}(x_1, \dots, x_n) = N_{PS}(x_1, \dots, x_n)/N$
- Then

$$\begin{aligned} \lim_{N\to\infty} \hat{P}(x_1, \dots, x_n) &= \lim_{N\to\infty} N_{PS}(x_1, \dots, x_n)/N \\ &= S_{PS}(x_1, \dots, x_n) \\ &= P(x_1 \dots x_n) \end{aligned}$$

- I.e., the sampling procedure is consistent

so so on every node.

Using samples

- We'll get a bunch of samples from the BN:

S_1 c, ¬s, r, w
S_2 ¬c, s, r, w
S_3 ¬c, s, r, ¬w
S_4 c, ¬s, r, w
S_5 ¬c, ¬s, ¬r, w

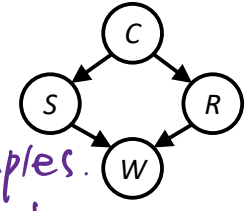
should be satisfied times

$P(c|+w) : \frac{3}{4}$

$P(c|+r,+w) : \frac{3}{3}$

$P(c|-r,-w)$: no samples.

$\hat{P}[w=+w] = 2.8.$

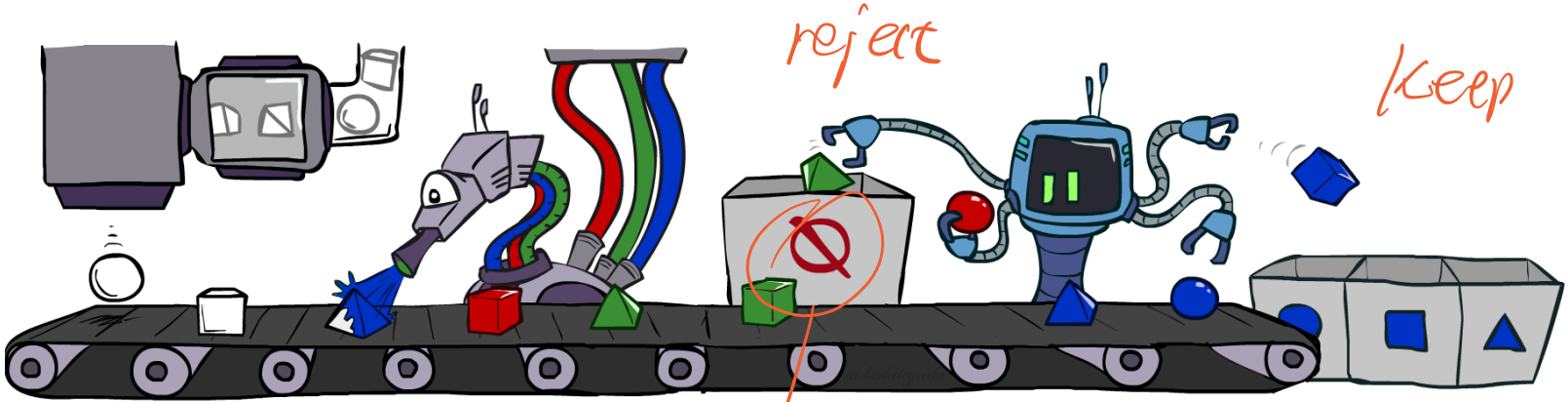


If rare enough never get some case!

- If we want to know P(*W*)
  - We have counts <w:4, ¬w:1>
  - Normalize to get P(*W*) = <w:0.8, ¬w:0.2>
  - This will get closer to the true distribution with more samples
- If we want to know P(*C*| *r, w*)
  - Count (*c, r, w*) and (¬*c, r, w*)
  - Normalize to get P(*C*| *r, w*) = <c:0.67, ¬c:0.33>

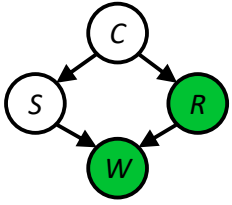
Rejection Sampling

metaphor: 打击，拒绝



Rejection Sampling

- A simple modification of prior sampling for conditional probabilities
- Let's say we want P(*C*| *r*, *w*)
- When generating a sample, reject it immediately if not *R*=true, *W*=true
- It is consistent for conditional probabilities (i.e., correct in the limit)



c, ¬s, r, w
c, s, ¬r
¬c, s, r, ¬w
c, ¬s, ¬r
¬c, ¬s, r, w

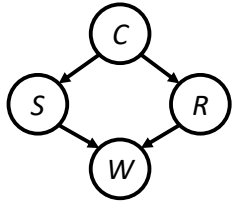
discard on the mid.

keep tallies about the events we care about as the sample stream by

Rejection Sampling

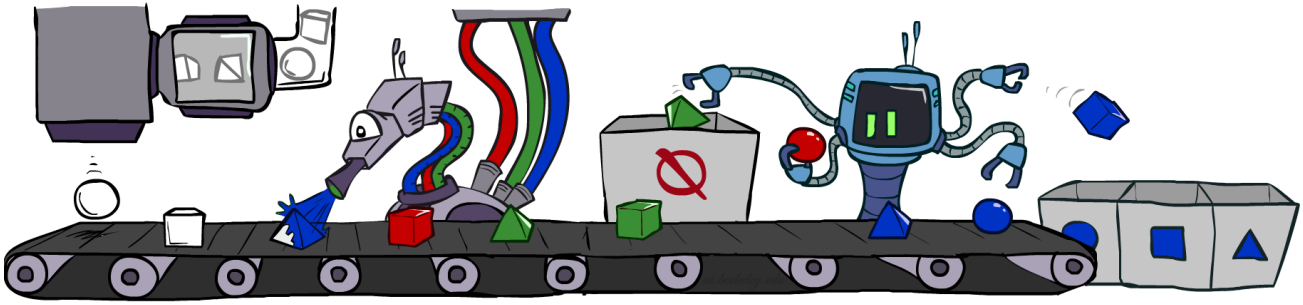
- Let's say we want P(C)
  - Just tally counts of C as we go

- Let's say we want P(C | +s)
  - Same thing: tally C outcomes, but ignore (reject) samples which don't have S=+s
  - This is called rejection sampling
  - We can toss out samples early!
  - It is also consistent for conditional probabilities (i.e., correct in the limit)

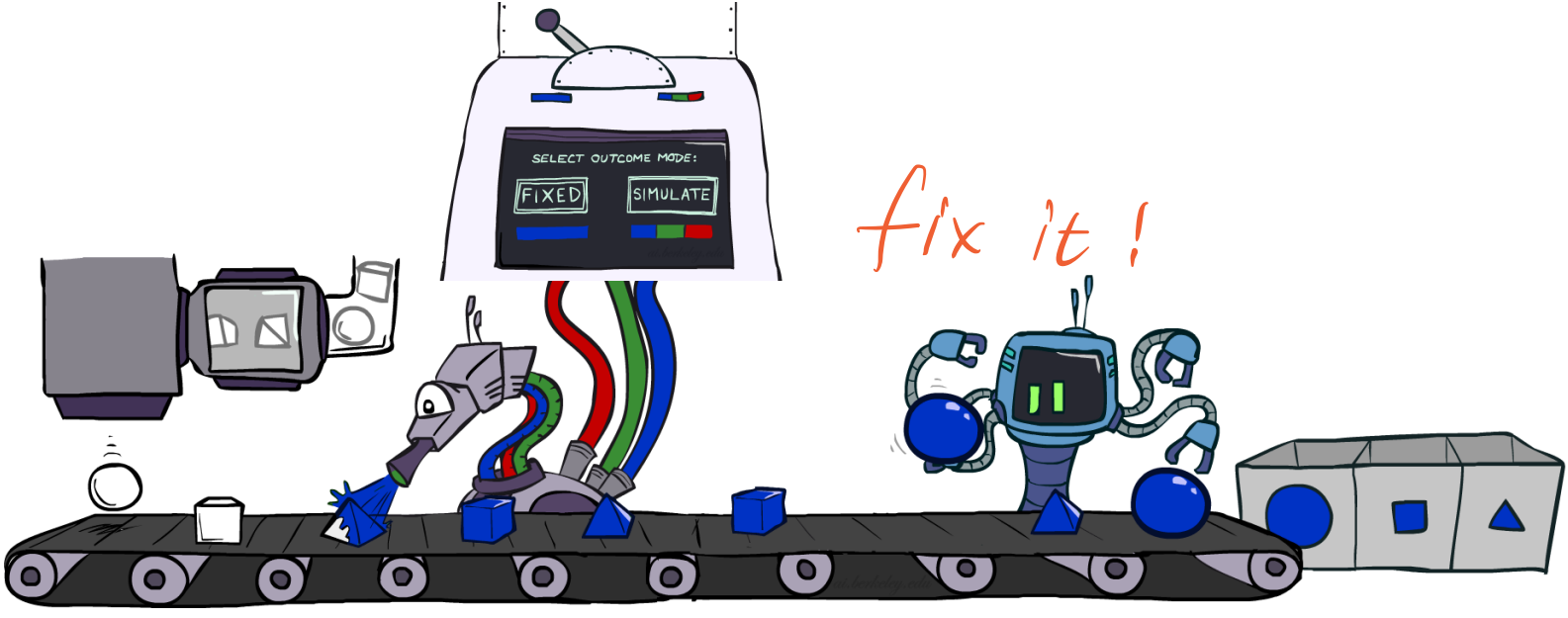


Rejection Sampling

- Input: evidence $e_1,..,e_k$ (instantiation)
- For i=1, 2, …, n
  - Sample X_i from $P(X_i \mid parents(X_i))$
  - If x_i not consistent with evidence
    - Reject: Return, and no sample is generated in this cycle
- Return $(x_1, x_2, …, x_n)$

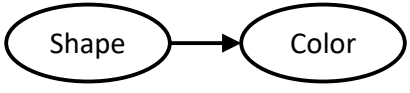


Likelihood Weighting

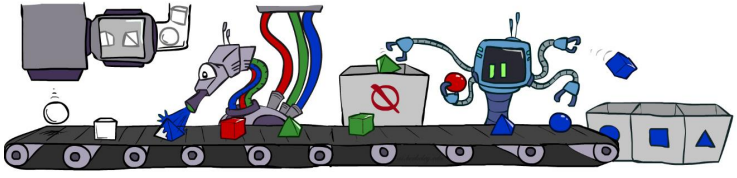


Likelihood Weighting

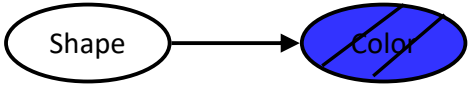
- Problem with rejection sampling:
  - If evidence is unlikely, rejects lots of samples
  - Evidence not exploited as you sample
  - Consider P(*Shape*|*Color=blue*)



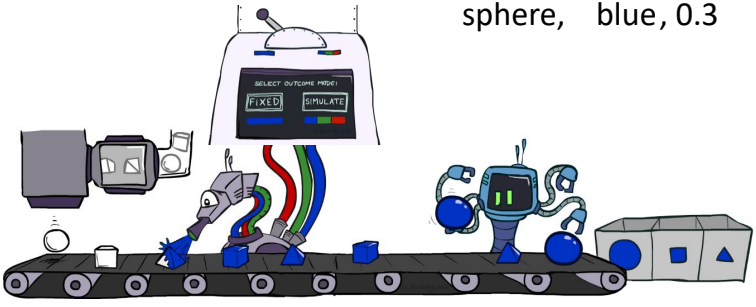
~~pyramid, green~~
~~pyramid, red~~
sphere, blue
~~cube, red~~
~~sphere, green~~



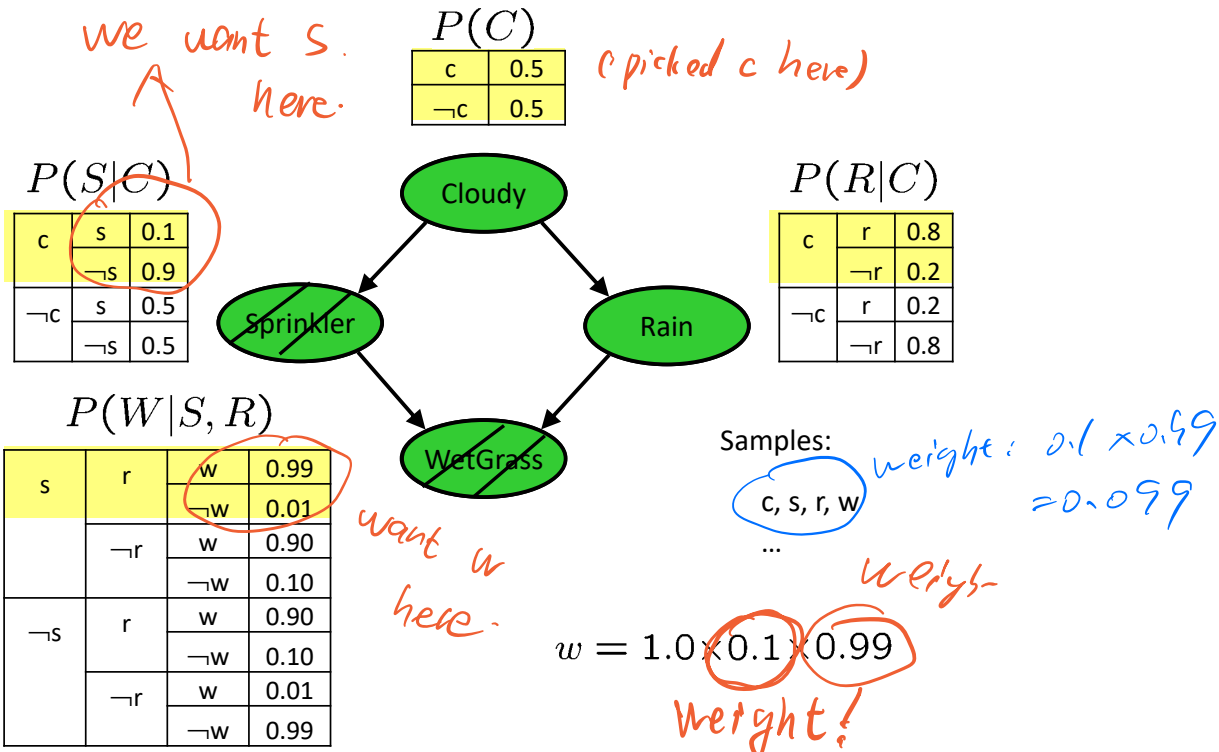
- Idea: fix evidence variables, sample the rest
  - Problem: sample distribution not consistent!
  - Solution: ***weight*** each sample by probability of evidence variables given parents



pyramid, blue , 0.4
pyramid, blue , 0.4
sphere, blue , 0.3
cube, blue , 0.8
sphere, blue , 0.3



Likelihood Weighting



Likelihood Weighting

- Input: evidence $e_1,..,e_k$
- $w = 1.0$
- for i=1, 2, …, n
  - if X_i is an evidence variable
    - x_i = observed value$_i$ for X_i
    - Set $w = w * P(x_i \mid Parents(X_i))$
  - else
    - Sample x_i from $P(X_i \mid Parents(X_i))$
- return $(x_1, x_2, …, x_n)$, w

target value.

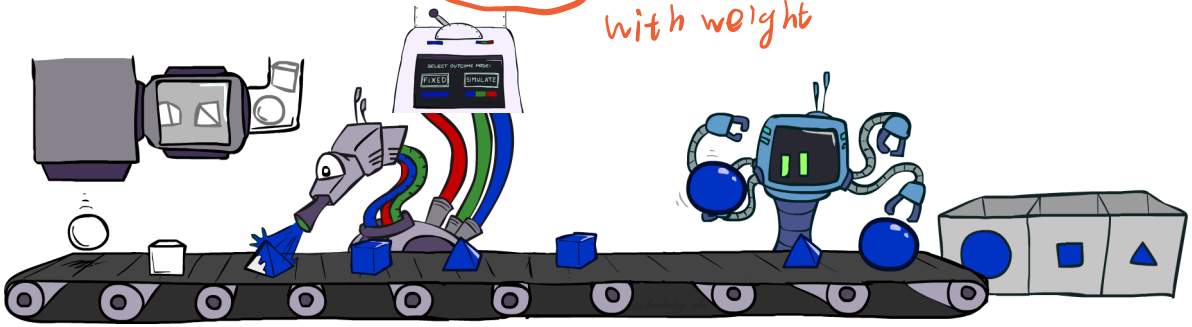
target

already known.

conditional prob.

normal case

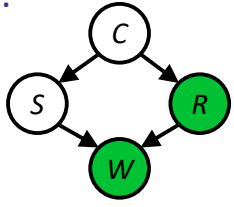
with weight



Using samples

- We'll get a bunch of weighted samples from the BN:

c, ¬s, r, w	0.1
c, s, r, w	0.2
¬c, s, r, w	0.3
c, ¬s, r, w	0.1
¬c, ¬s, r, w	0.5



- If we want to know P(*C*| *r, w*)
  - We have weight sums <(*c, r, w*): 0.4, (¬*c, r, w*): 0.8>
  - Normalize to get P(*C*| *r, w*) = <*c*: 0.33, ¬*c*: 0.67>
  - This will get closer to the true distribution with more samples

Likelihood Weighting

- Sampling distribution (z is sampled and e is fixed evidence)

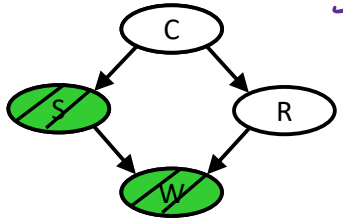
$$S_{WS}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{l} P(z_i | \text{Parents}(Z_i))$$

real sampled nodes.

- Now, samples have weights

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

other nodes



Start from the top

- Together, weighted sampling distribution is consistent

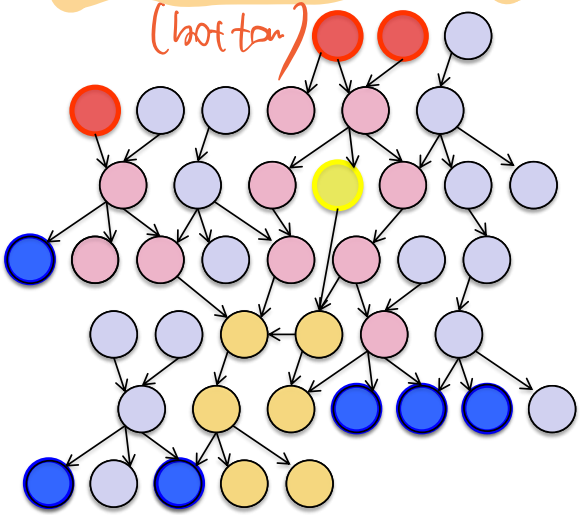
$$S_{\text{WS}}(z, e) \cdot w(z, e) = \prod_{i=1}^{l} P(z_i | \text{Parents}(z_i)) \prod_{i=1}^{m} P(e_i | \text{Parents}(e_i))$$

$$= P(\mathbf{z}, \mathbf{e})$$

consitent again

Likelihood Weighting

- Likelihood weighting is good
  - All samples are used
  - The values of ***downstream*** variables are influenced by ***upstream*** evidence



- Likelihood weighting still has weaknesses
  - The values of ***upstream*** variables are unaffected by ***downstream*** evidence
  - With many downstream evidence, we may
    - mostly get samples that are inconsistent with the evidence and thus have very small weights
    - get a few lucky samples with very large weights, which dominate the result

- We would like each variable to "see" ***all*** the evidence!

top

typical things

things don't consistent

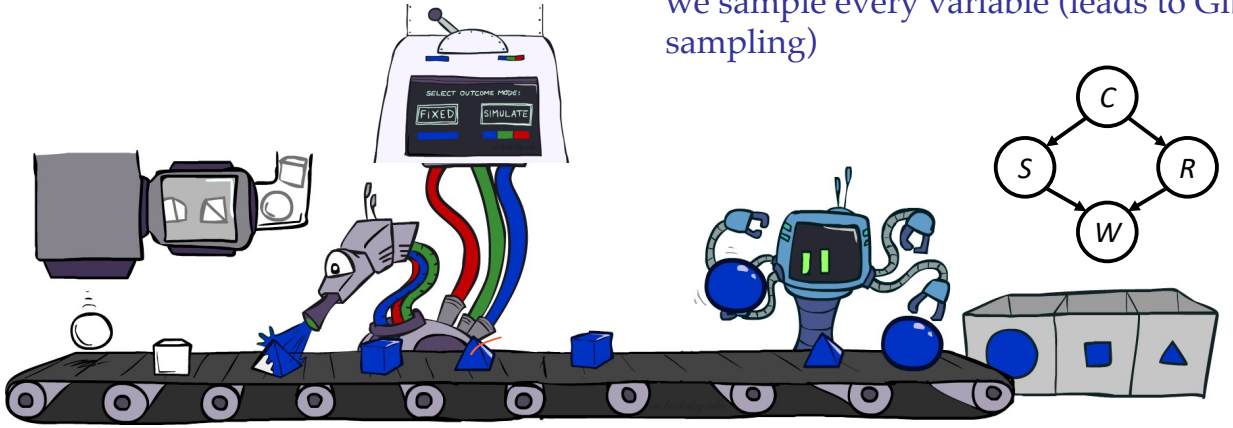
eg. want call (bottom)

but usually no earthquake (top)

result in small

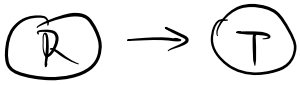
Likelihood Weighting

- Likelihood weighting is helpful
  - We have taken evidence into account as we generate the sample
  - E.g. here, W's value will get picked based on the evidence values of S, R
  - More of our samples will reflect the state of the world suggested by the evidence



- Likelihood weighting doesn't solve all our problems
  - Evidence influences the choice of downstream variables, but not upstream ones (C isn't more likely to get a value matching the evidence)
- We would like to consider evidence when we sample every variable (leads to Gibbs sampling)

Examples



P(R)
+r 0.3
-r 0.7

P(T|R)

	+r	-r
+t	0.9	0.5
-t	0.1	0.5

samples
-r, +t
-r, -t (reject)
+r, +t
+r, +t
...

Prior: ...
$\hat{P}(T) = \begin{matrix} +r & \frac{3}{4} \\ -r & \frac{1}{4} \end{matrix}$

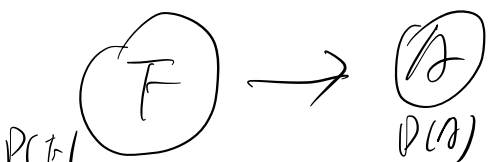
$\hat{P}(R|+t) = \begin{matrix} +r & \frac{2}{3} \\ -r & \frac{1}{3} \end{matrix}$

R(R|+t)
likelyhood:

weight

+r, +t	0.9	P(t|-r)
-r, +t	0.5	
+r, +t	0.9	
-r, +t		

+r: 1.8
-r: 0.5
normalized → answer



P(F|
+f 0.01
-f 0.99

P(a).

+f	-f
+a 0.9	+a 0.01
-a 0.1	-a 0.99

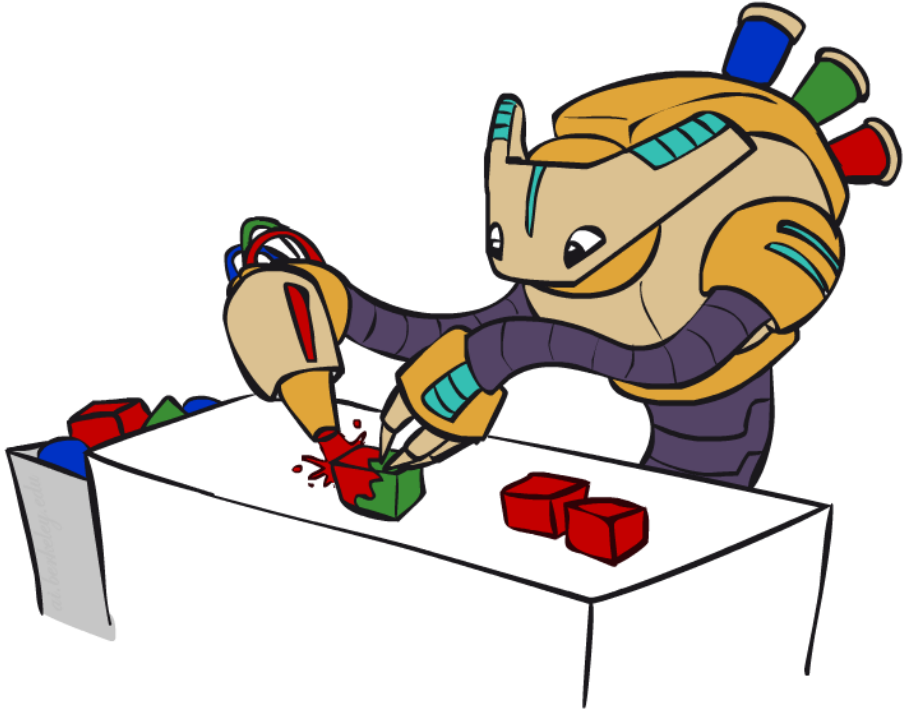
P(F|+a) { +f : 2/11
-f : 9/11

likely

-f,	+a	0.01
-f,	+a	0.09
+f	+a	0.9

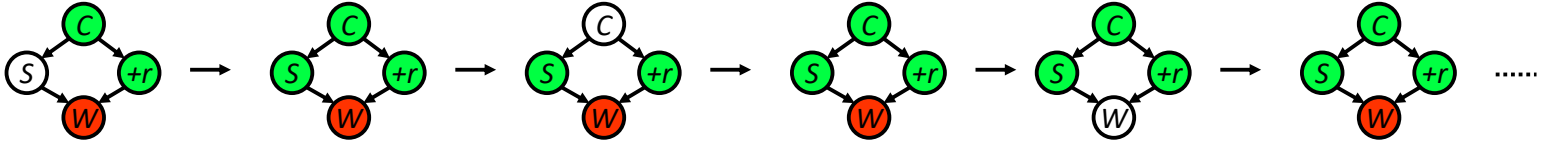
} 20

Gibbs Sampling



Gibbs Sampling

- ***Procedure***: keep track of a full instantiation $x_1, x_2, \ldots, x_n$. 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.
- ***Property***: in the limit of repeating this infinitely many times the resulting samples come from the correct distribution (i.e. conditioned on evidence).
- ***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. Sum of weights over all samples is indicative of how many "effective" samples were obtained, so we want high weight.



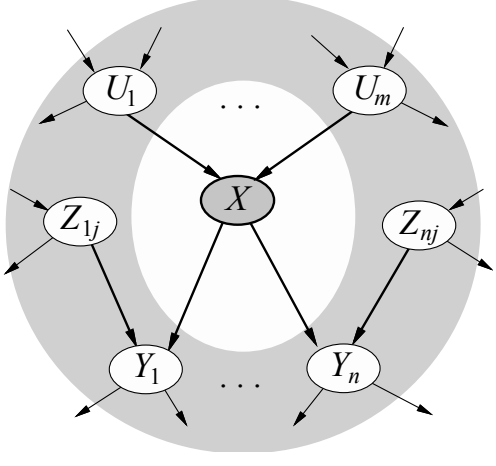
Gibbs Sampling

- Generate each sample by making a random change to the preceding sample
  - Evidence variables remain fixed. For each of the non-evidence variable, sample its value conditioned on all the other variables
  - $X_i' \sim P(X_i \mid x_1,..,x_{i-1},x_{i+1},..,x_n)$
  - In a Bayes net

    $P(X_i \mid x_1,..,x_{i-1},x_{i+1},..,x_n)$

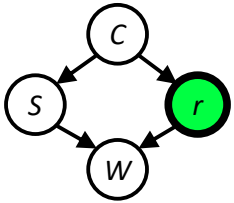
    $= P(X_i \mid markov\_blanket(X_i))$

    $= \alpha\ P(X_i \mid u_1,..,u_m)\ \prod_j P(y_j \mid parents(Y_j))$

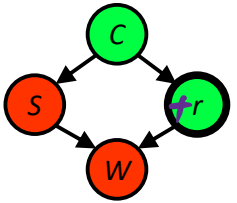


Gibbs Sampling Example: $P(S \mid r)$

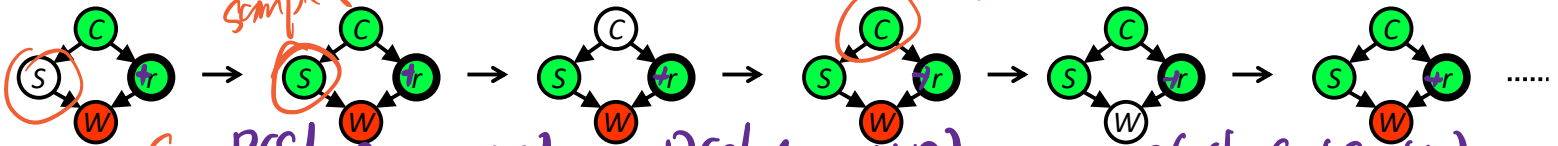
- Step 1: Fix evidence
  - R = true



- Step 2: Initialize other variables
  - Randomly



- Step 3: Repeat
  - Choose an arbitrary non-evidence variable X
  - Resample X from $P(X \mid markov\_blanket(X))$



Resampling of One Variable

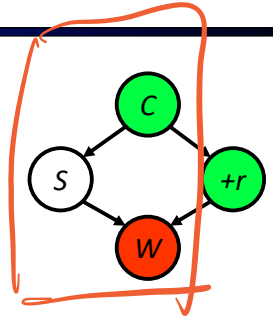
- Sample from P(S | +c, +r, -w)

$$P(S|+c,+r,-w) = \frac{P(S,+c,+r,-w)}{P(+c,+r,-w)}$$
$$= \frac{P(S,+c,+r,-w)}{\sum_s P(s,+c,+r,-w)}$$
$$= \frac{P(+c)P(S|+c)P(+r|+c)P(-w|S,+r)}{\sum_s P(+c)P(s|+c)P(+r|+c)P(-w|s,+r)}$$
$$= \frac{P(+c)P(S|+c)P(+r|+c)P(-w|S,+r)}{P(+c)P(+r|+c)\sum_s P(s|+c)P(-w|s,+r)}$$
$$= \frac{P(S|+c)P(-w|S,+r)}{\sum_s P(s|+c)P(-w|s,+r)}$$

sampled var

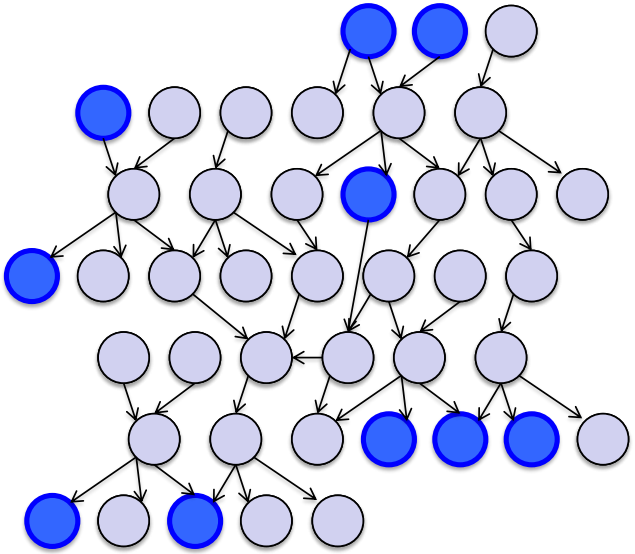
comes from BN

only which matters S as alters.



- Many things cancel out – only CPTs with S remain!
- More generally: only CPTs that have resampled variable need to be considered, and joined together

Why doing this?



- Samples soon begin to reflect all the evidence in the network
- Eventually they are being drawn from the true posterior!

- Theorem: Gibbs sampling is consistent

Gibbs consistent !

Why does it work? (see AIMA 14.5.2 for details)

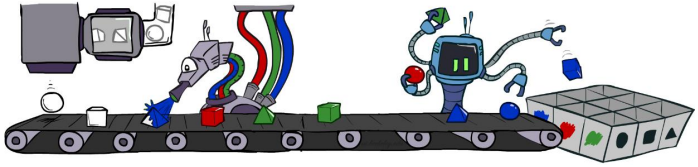
- Suppose we run it for a long time and predict the probability of reaching any given state at time t: $\pi_t(x_1,...,x_n)$ or $\pi_t(\underline{\mathbf{x}})$
- Each Gibbs sampling step (pick a variable, resample its value) applied to a state $\underline{\mathbf{x}}$ has a probability $q(\underline{\mathbf{x}}' \mid \underline{\mathbf{x}})$ of reaching a next state $\underline{\mathbf{x}}'$
- So $\pi_{t+1}(\underline{\mathbf{x}}') = \sum_{\underline{\mathbf{x}}} q(\underline{\mathbf{x}}' \mid \underline{\mathbf{x}})\, \pi_t(\underline{\mathbf{x}})$ or, in matrix/vector form $\pi_{t+1} = Q\pi_t$
- When the process is in equilibrium $\pi_{t+1} = \pi_t$ so $Q\pi_t = \pi_t$
- This has a unique solution $\pi_t = P(x_1,...,x_n \mid e_1,...,e_k)$
- So for large enough t the next sample will be drawn from the true posterior

Markov Chain Monte Carlo (MCMC)

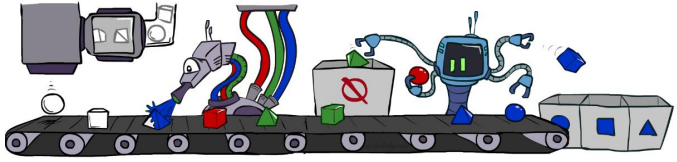
- MCMC is a family of randomized algorithms for approximating some quantity of interest over a very large state space
  - Markov chain = a sequence of randomly chosen states ("random walk"), where each state is chosen conditioned on the previous state
  - ~~Monte Carlo = a very expensive city in Monaco with a famous casino~~
  - Monte Carlo = an algorithm (usually based on sampling) that is likely to find a correct answer
- MCMC = sampling by constructing a Markov chain
- Gibbs, Metropolis-Hastings, Hamiltonian, Slice, etc.

Summary

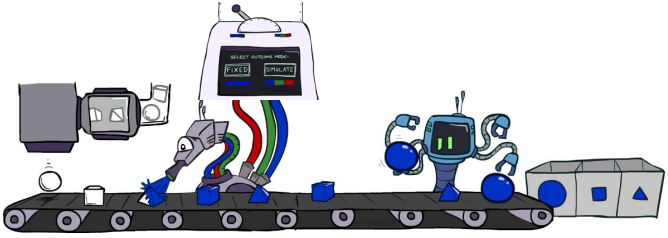
- Prior Sampling P



- Rejection Sampling P(Q | e)



- Likelihood Weighting P(Q | e)



- Gibbs Sampling P(Q | e)