CS 188: Artificial Intelligence

Probabilistic Inference:
Enumeration, Variable Elimination, Sampling

Pieter Abbeel – UC Berkeley
Many slides over this course adapted from Dan Klein, Stuart Russell, Andrew Moore

Bayes' Nets

- ✔ Representation
- ✔ Conditional Independences
- Probabilistic Inference
  - Enumeration (exact, exponential complexity)
  - Variable elimination (exact, worst-case exponential complexity, often better)
  - Probabilistic inference is NP-complete
  - Sampling (approximate)
- Learning Bayes' Nets from Data

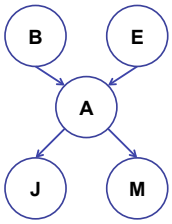
Inference

- Inference: calculating some useful quantity from a joint probability distribution
- Examples:
  - Posterior probability:

$$P(Q|E_1 = e_1, \ldots E_k = e_k)$$

  - Most likely explanation:

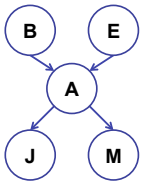
$$\text{argmax}_q \; P(Q = q|E_1 = e_1 \ldots)$$



Inference by Enumeration

- Given unlimited time, inference in BNs is easy
- Recipe:
  - State the marginal probabilities you need
  - Figure out ALL the atomic probabilities you need
  - Calculate and combine them
- Example:

$$P(+b|+j,+m) = \frac{P(+b,+j,+m)}{P(+j,+m)}$$

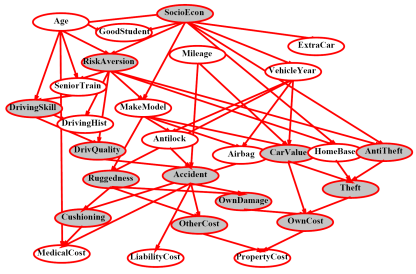


Example: Enumeration

- In this simple method, we only need the BN to synthesize the joint entries

$$P(+b,+j,+m) =$$
$$P(+b)P(+e)P(+a|+b,+e)P(+j|+a)P(+m|+a)+$$
$$P(+b)P(+e)P(-a|+b,+e)P(+j|-a)P(+m|-a)+$$
$$P(+b)P(-e)P(+a|+b,-e)P(+j|+a)P(+m|+a)+$$
$$P(+b)P(-e)P(-a|+b,-e)P(+j|-a)P(+m|-a)$$

Inference by Enumeration?



Variable Elimination

- Why is inference by enumeration so slow?
  - You join up the whole joint distribution before you sum out the hidden variables
- Idea: interleave joining and marginalizing!
  - Called "Variable Elimination"
  - Still NP-hard, but usually much faster than inference by enumeration
- We'll need some new notation to define VE

8

Factor Zoo I

- Joint distribution: P(X,Y)
  - Entries P(x,y) for all x, y
  - Sums to 1

$P(T,W)$

T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

- Selected joint: P(x,Y)
  - A slice of the joint distribution
  - Entries P(x,y) for fixed x, all y
  - Sums to P(x)

$P(cold,W)$

T	W	P
cold	sun	0.2
cold	rain	0.3

- Number of capitals = dimensionality of the table

9

Factor Zoo II

- Family of conditionals: P(X |Y)
  - Multiple conditionals
  - Entries P(x | y) for all x, y
  - Sums to |Y|

$P(W|T)$

T	W	P	
hot	sun	0.8	$P(W\|hot)$
hot	rain	0.2	
cold	sun	0.4	$P(W\|cold)$
cold	rain	0.6	

- Single conditional: P(Y | x)
  - Entries P(y | x) for fixed x, all y
  - Sums to 1

$P(W|cold)$

T	W	P
cold	sun	0.4
cold	rain	0.6

10

Factor Zoo III

- Specified family: P(y | X)
  - Entries P(y | x) for fixed y, but for all x
  - Sums to … who knows!

$P(rain|T)$

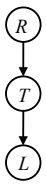
T	W	P	
hot	rain	0.2	$P(rain\|hot)$
cold	rain	0.6	$P(rain\|cold)$

- In general, when we write $P(Y_1 \ldots Y_N \mid X_1 \ldots X_M)$
  - It is a "factor," a multi-dimensional array
  - Its values are all $P(y_1 \ldots y_N \mid x_1 \ldots x_M)$
  - Any assigned X or Y is a dimension missing (selected) from the array

11

Example: Traffic Domain

- Random Variables
  - R: Raining
  - T: Traffic
  - L: Late for class!



$P(R)$

+r	0.1
-r	0.9

$P(T|R)$

+r	+t	0.8
+r	-t	0.2
-r	+t	0.1
-r	-t	0.9

$P(L|R)$

+t	+l	0.3
+t	-l	0.7
-t	+l	0.1
-t	-l	0.9

12

Variable Elimination Outline

- Track objects called factors
- Initial factors are local CPTs (one per node)

$P(R)$

+r	0.1
-r	0.9

$P(T|R)$

+r	+t	0.8
+r	-t	0.2
-r	+t	0.1
-r	-t	0.9

$P(L|T)$

+t	+l	0.3
+t	-l	0.7
-t	+l	0.1
-t	-l	0.9

- Any known values are selected
  - E.g. if we know $L = +\ell$, the initial factors are

$P(R)$

+r	0.1
-r	0.9

$P(T|R)$

+r	+t	0.8
+r	-t	0.2
-r	+t	0.1
-r	-t	0.9

$P(+\ell|T)$

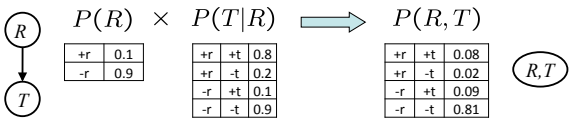
+t	+l	0.3
-t	+l	0.1

- VE: Alternately join factors and eliminate variables

13

Operation 1: Join Factors

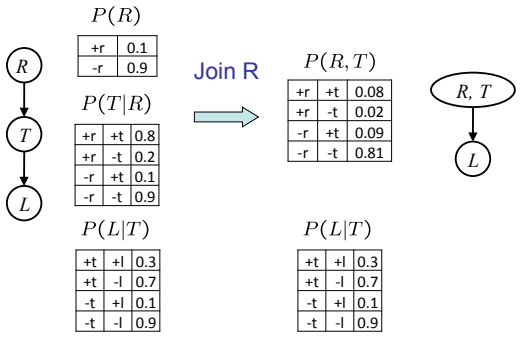
- First basic operation: joining factors
- Combining factors:
  - Just like a database join
  - Get all factors over the joining variable
  - Build a new factor over the union of the variables involved
- Example: Join on R



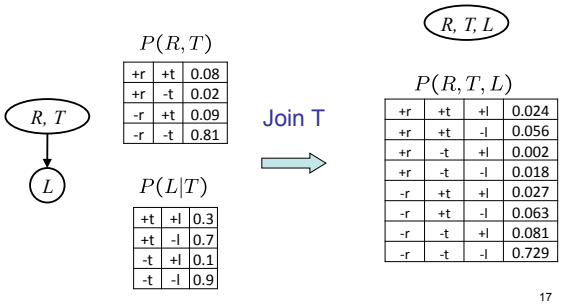
- Computation for each entry: pointwise products

$$\forall r, t: \quad P(r,t) = P(r) \cdot P(t|r)$$

Example: Multiple Joins

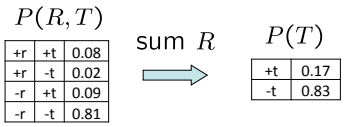


Example: Multiple Joins

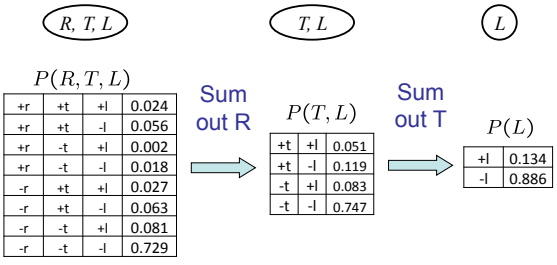


Operation 2: Eliminate

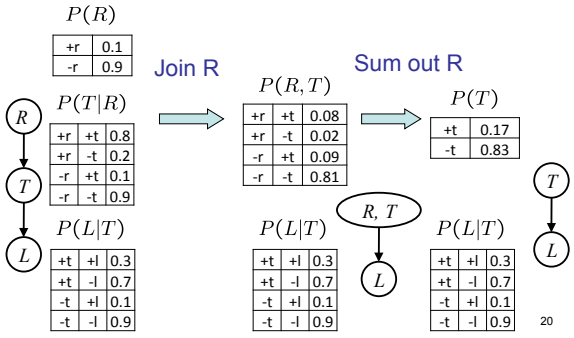
- Second basic operation: marginalization
- Take a factor and sum out a variable
  - Shrinks a factor to a smaller one
  - A projection operation
- Example:



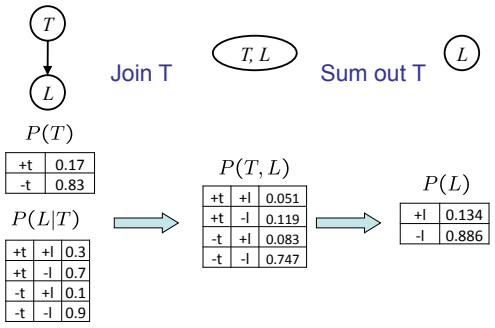
Multiple Elimination



P(L) : Marginalizing Early!



Marginalizing Early (aka VE*)



* VE is variable elimination

Evidence

- If evidence, start with factors that select that evidence
  - No evidence uses these initial factors:

$P(R)$

+r	0.1
-r	0.9

$P(T|R)$

+r	+t	0.8
+r	-t	0.2
-r	+t	0.1
-r	-t	0.9

$P(L|T)$

+t	+l	0.3
+t	-l	0.7
-t	+l	0.1
-t	-l	0.9

  - Computing $P(L|+r)$, the initial factors become:

$P(+r)$

+r	0.1

$P(T|+r)$

+r	+t	0.8
+r	-t	0.2

$P(L|T)$

+t	+l	0.3
+t	-l	0.7
-t	+l	0.1
-t	-l	0.9

- We eliminate all vars other than query + evidence

Evidence II

- Result will be a selected joint of query and evidence
  - E.g. for P(L | +r), we'd end up with:

- To get our answer, just normalize this!
- That's it!

General Variable Elimination

- Query: $P(Q|E_1 = e_1, \ldots E_k = e_k)$
- Start with initial factors:
  - Local CPTs (but instantiated by evidence)
- While there are still hidden variables (not Q or evidence):
  - Pick a hidden variable H
  - Join all factors mentioning H
  - Eliminate (sum out) H
- Join all remaining factors and normalize

Example

$P(B|j,m) \propto P(B,j,m)$

$P(B)$ $P(E)$ $P(A|B,E)$ $P(j|A)$ $P(m|A)$

Choose A

$P(A|B,E)$
$P(j|A)$
$P(m|A)$
→ (×) → $P(j,m,A|B,E)$ → (Σ) → $P(j,m|B,E)$

$P(B)$ $P(E)$ $P(j,m|B,E)$

Example

$P(B)$ $P(E)$ $P(j,m|B,E)$

Choose E

$P(E)$
$P(j,m|B,E)$
→ (×) → $P(j,m,E|B)$ → (Σ) → $P(j,m|B)$

$P(B)$ $P(j,m|B)$

Finish with B

$P(B)$
$P(j,m|B)$
→ (×) → $P(j,m,B)$ → Normalize → $P(B|j,m)$

Same Example in Equations

$P(B|j,m) \propto P(B,j,m)$

$P(B)$	$P(E)$	$P(A\|B,E)$	$P(j\|A)$	$P(m\|A)$

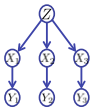
$$
\begin{aligned}
P(B|j,m) &\propto P(B,j,m) \\
&= \sum_{e,a} P(B,j,m,e,a) && \text{marginal can be obtained from joint by summing out} \\
&= \sum_{e,a} P(B)P(e)P(a|B,e)P(j|a)P(m|a) && \text{use Bayes' net joint distribution expression} \\
&= \sum_{e} P(B)P(e)\sum_{a} P(a|B,e)P(j|a)P(m|a) && \text{use x*(y+z) = xy + xz} \\
&= \sum_{e} P(B)P(e)f_1(B,e,j,m) && \text{joining on a, and then summing out gives f_1} \\
&= P(B)\sum_{e} P(e)f_1(B,e,j,m) && \text{x*(y+z) = xy + xz} \\
&= P(B)f_2(B,j,m) && \text{joining on e, and then summing out gives f_2}
\end{aligned}
$$

28

All we are doing is exploiting xy + xz = x(y+z) to improve computational efficiency!

Another (bit more abstractly worked out) Variable Elimination Example

Query: $P(X_3|Y_1=y_1, Y_2=y_2, Y_3=y_3)$



Start by inserting evidence, which gives the following initial factors:

$$p(Z)p(X_1|Z)p(X_2|Z)p(X_3|Z)p(y_1|X_1)p(y_2|X_2)p(y_3|X_3)$$

Eliminate X_1, this introduces the factor $f_1(Z,y_1) = \sum_{x_1} p(x_1|Z)p(y_1|x_1)$, and:

$$p(Z)f_1(Z,y_1)p(X_2|Z)p(X_3|Z)p(y_2|X_2)p(y_3|X_3)$$

Eliminate X_2, this introduces the factor $f_2(Z,y_2) = \sum_{x_2} p(x_2|Z)p(y_2|x_2)$, and:

$$p(Z)f_1(Z,y_1)f_2(Z,y_2)p(X_3|Z)p(y_3|X_3)$$

Eliminate Z, this introduces the factor $f_3(y_1,y_2,y_3,X_3) = \sum_z p(z)f_1(z,y_1)f_2(z,y_2)p(X_3|z)p(y_3|X_3)$, and:

$$f_3(y_1,y_2,y_3,X_3)$$

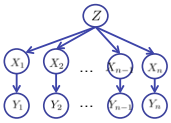
Normalizing over X_3 gives $P(X_3|y_1,y_2,y_3)$.

Computational complexity critically depends on the largest factor being generated in this process. Size of factor = number of entries in table. In example above (assuming binary) all factors generated are of size 2 --- as they all only have one variable (Z, Z, and X3 respectively).

29

Variable Elimination Ordering

- For the query $P(X_n|y_1,\ldots,y_n)$ work through the following two different orderings as done in previous slide: Z, X_1, …, X_{n-1} and X_1, …, X_{n-1}, Z. What is the size of the maximum factor generated for each of the orderings?



- Answer: 2^n versus 2 (assuming binary)
- In general: the ordering can greatly affect efficiency.

30

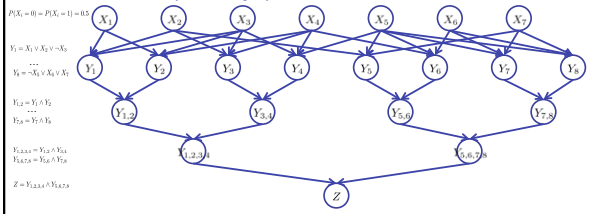
Computational and Space Complexity of Variable Elimination

- The computational and space complexity of variable elimination is determined by the largest factor
- The elimination ordering can greatly affect the size of the largest factor.
  - E.g., previous slide's example 2^n vs. 2
- Does there always exist an ordering that only results in small factors?
  - No!

31

Worst Case Complexity?

- Consider the 3-SAT clause:
$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee \neg x_4) \wedge (x_2 \vee \neg x_2 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_5) \wedge (x_2 \vee x_5 \vee x_7) \wedge (x_4 \vee x_5 \vee x_6) \wedge (\neg x_5 \vee x_6 \vee \neg x_7) \wedge (\neg x_5 \vee \neg x_6 \vee x_7)$
which can be encoded by the following Bayes' net:



- If we can answer P(z) equal to zero or not, we answered whether the 3-SAT problem has a solution.
- Subtlety: why the cascaded version of the AND rather than feeding all OR clauses into a single AND? Answer: a single AND would have an exponentially large CPT, whereas with representation above the Bayes' net has small CPTs only.
- Hence inference in Bayes' nets is NP-hard. No known efficient probabilistic inference in general.

32

Polytrees

- A polytree is a directed graph with no undirected cycles
- For poly-trees you can always find an ordering that is efficient
  - Try it!!
- Cut-set conditioning for Bayes' net inference
  - Choose set of variables such that if removed only a polytree remains
  - Think about how the specifics would work out!

33

Bayes' Nets

- ✓ Representation
- ✓ Conditional Independences
- Probabilistic Inference
  - ✓ Enumeration (exact, exponential complexity)
  - ✓ Variable elimination (exact, worst-case exponential complexity, often better)
  - ✓ Probabilistic inference is NP-complete
  - Sampling (approximate)
- Learning Bayes' Nets from Data

Sampling

- Simulation has a name: sampling (e.g., predicting the weather, basketball games, …)
- Basic idea:
  - Draw N samples from a sampling distribution S
  - Compute an approximate posterior probability
  - Show this converges to the true probability P
- Why sample?
  - Learning: get samples from a distribution you don't know
  - Inference: getting a sample is faster than computing the right answer (e.g. with variable elimination)

Sampling

- How do you sample?
  - Simplest way is to use a random number generator to get a continuous value uniformly distributed between 0 and 1 (e.g. random() in Python)
  - Assign each value in the domain of your random variable a sub-interval of [0,1] with a size equal to its probability
    - The sub-intervals cannot overlap

Sampling Example

- Each value in the domain of W has a sub-interval of [0,1] with a size equal to its probability

W	P(W)
Sun	0.6
Rain	0.1
Fog	0.3
Meteor	0.0

u is a uniform random value in $[0, 1]$

if $0.0 \leq u < 0.6, w = \text{sun}$

if $0.6 \leq u < 0.7, w = \text{rain}$

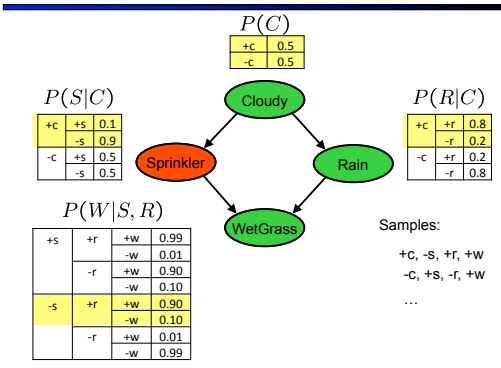
if $0.7 \leq u < 1.0, w = \text{fog}$

e.g. if `random() returns` u `= 0.83, then our sample is` w `= fog`

Sampling in Bayes' Nets

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

Prior Sampling



Prior Sampling

- To generate one sample from a Bayes' net with n variables. Assume variables are named such that ordering X_1, X_2, …, X_n is consistent with the DAG.
- For i=1, 2, …, n
  - Sample x_i from P(X_i | Parents(X_i))
- End For
- Return (x_1, x_2, …, x_n)

Prior Sampling

- This process generates samples with probability:

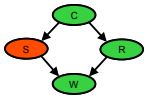
$$S_{PS}(x_1 \dots x_n) = \prod_{i=1}^{n} P(x_i|\text{Parents}(X_i)) = P(x_1 \dots x_n)$$

  …i.e. the BN's joint probability
- Let the number of samples of an event be $N_{PS}(x_1 \dots x_n)$
- Then $\lim_{N\to\infty} \hat{P}(x_1, \dots, x_n) = \lim_{N\to\infty} N_{PS}(x_1, \dots, x_n)/N = S_{PS}(x_1, \dots, x_n) = P(x_1 \dots x_n)$
- I.e., the sampling procedure is consistent

Example

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

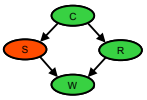
  +c, -s, +r, +w
  +c, +s, +r, +w
  -c, +s, +r, -w
  +c, -s, +r, +w
  -c, -s, -r, +w



- 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
  - Can estimate anything else, too
  - What about P(C| +w)? P(C| +r, +w)? P(C| -r, -w)?
  - Fast: can use fewer samples if less time (what's the drawback?)

Rejection Sampling

- Let's say we want P(C)
  - No point keeping all samples around
  - 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
  - It is also consistent for conditional probabilities (i.e., correct in the limit)



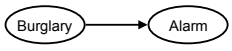
+c, -s, +r, +w
+c, +s, +r, +w
-c, +s, +r, -w
+c, -s, +r, +w
-c, -s, -r, +w

Rejection Sampling

- For i=1, 2, …, n
  - Sample x_i from P(X_i | Parents(X_i))
  - If x_i not consistent with the evidence in the query, exit this for-loop and no sample is generated
- End For
- Return (x_1, x_2, …, x_n)

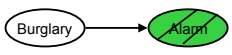
Likelihood Weighting

- Problem with rejection sampling:
  - If evidence is unlikely, you reject a lot of samples
  - You don't exploit your evidence as you sample
  - Consider P(B|+a)



-b, -a
-b, -a
-b, -a
-b, -a
+b, +a

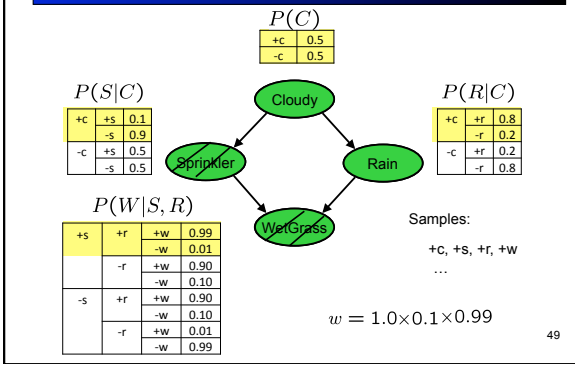
- Idea: fix evidence variables and sample the rest



-b +a
-b, +a
-b, +a
-b, +a
+b, +a

- Problem: sample distribution not consistent!
- Solution: weight by probability of evidence given parents

Likelihood Weighting



$$w = 1.0 \times 0.1 \times 0.99$$

Likelihood Weighting

- Set w = 1.0
- For i=1, 2, …, n
  - If X_i is an evidence variable
    - Set X_i = observation x_i for X_i
    - Set w = w * $P(x_i \mid \text{Parents}(X_i))$
  - Else
    - Sample x_i from $P(X_i \mid \text{Parents}(X_i))$
- End For
- Return $(x_1, x_2, \ldots, x_n)$, w

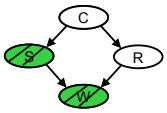
Likelihood Weighting

- Sampling distribution if z sampled and e fixed evidence

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

- Now, samples have weights

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

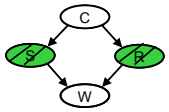


- Together, weighted sampling distribution is consistent

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

Likelihood Weighting

- Likelihood weighting is good
  - 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
  - → Gibbs sampling



Gibbs Sampling

- *Procedure:* keep track of a full instantiation $x_1, x_2, \ldots, x_n$. Start with an arbitrary instantation 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 sample is coming from the correct distribution
- *What's the point*: 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 want high weight.

Gibbs Sampling

- Say we want to sample P(S | R = +r)
- Step 1: Initialize
  - Set evidence (R = +r)
  - Set all other variables (S, C, W) to random values (e.g. by prior sampling or just uniformly sampling; say S = s, W = +w, C = -c)
- Steps 2+: Repeat the following for some number of iterations
  - Choose a non-evidence variable (S, W, or C in this case)
  - Sample this variable conditioned on nothing else changing
    - The first time through, if we pick S, we sample from P(S | R = +r, W = +w, C =-c)
  - The new sample can only be different in a single variable

Gibbs Sampling Example

- Want to sample from P(R | +s,-c,-w)
  - Shorthand for P(R | S=+s,C=-c,W=-w)

$$
\begin{aligned}
P(R|+s,-c,-w) &= \frac{P(R,+s,-c,-w)}{P(+s,-c,-w)} \\
&= \frac{P(R,+s,-c,-w)}{\sum_r P(R=r,+s,-c,-w)} \\
&= \frac{P(-c)P(+s|-c)P(R|-c)P(-w|+s,R)}{\sum_r P(-c)P(+s|-c)P(R=r|-c)P(-w|+s,R=r)} \\
&= \frac{P(R|-c)P(-w|+s,R)}{\sum_r P(R=r|-c)P(-w|+s,R=r)}
\end{aligned}
$$

- Many things cancel out -- just a join on R

56

Further Reading*

- Gibbs sampling is a special case of more general methods called Markov chain Monte Carlo (MCMC) methods
  - Metropolis-Hastings is one of the more famous MCMC methods (in fact, Gibbs sampling is a special case of Metropolis-Hastings)
- You may read about Monte Carlo methods – they're just sampling

57