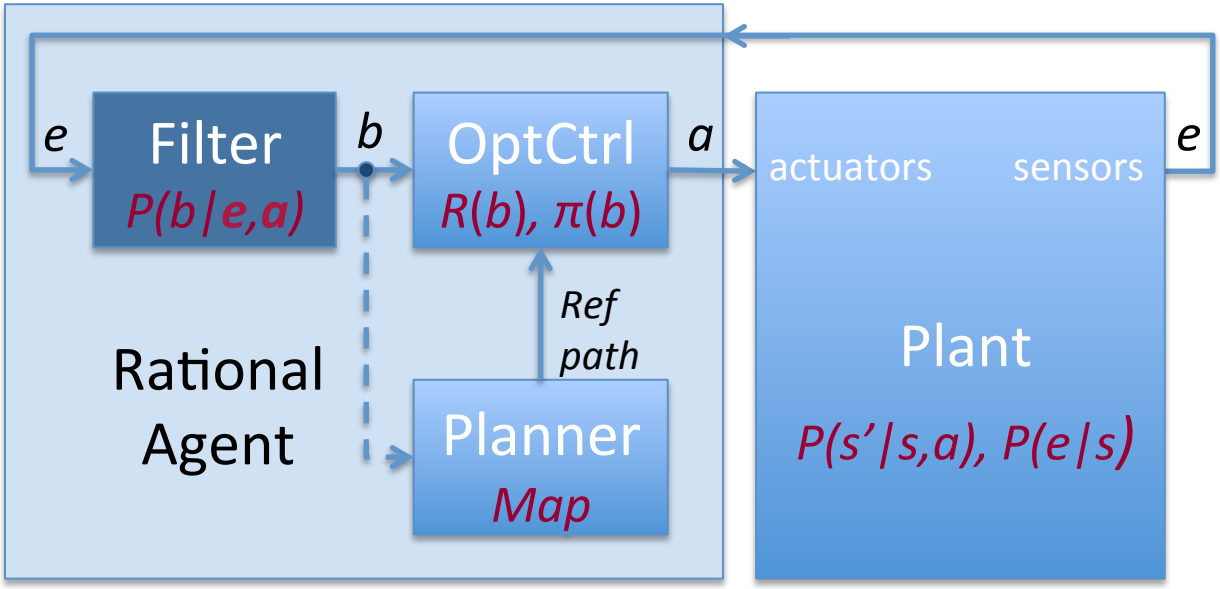
Probabilistic Reasoning

Chapter 14 (Inference)



Bayes' Nets

- ✓ Representation
- ✓ Conditional Independences

Bayes' Nets

- ✓ Representation
- ✓ Conditional Independences
- Probabilistic Inference

Bayes' Nets

- ✓ Representation
- ✓ Conditional Independences
- Probabilistic Inference
  - Enumeration (exact, exponential complexity)

Bayes' Nets

- ✓ Representation
- ✓ Conditional Independences
- Probabilistic Inference
  - Enumeration (exact, exponential complexity)
  - Variable elimination (exact, worst-case exponential complexity, often better)

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

Bayes' Nets

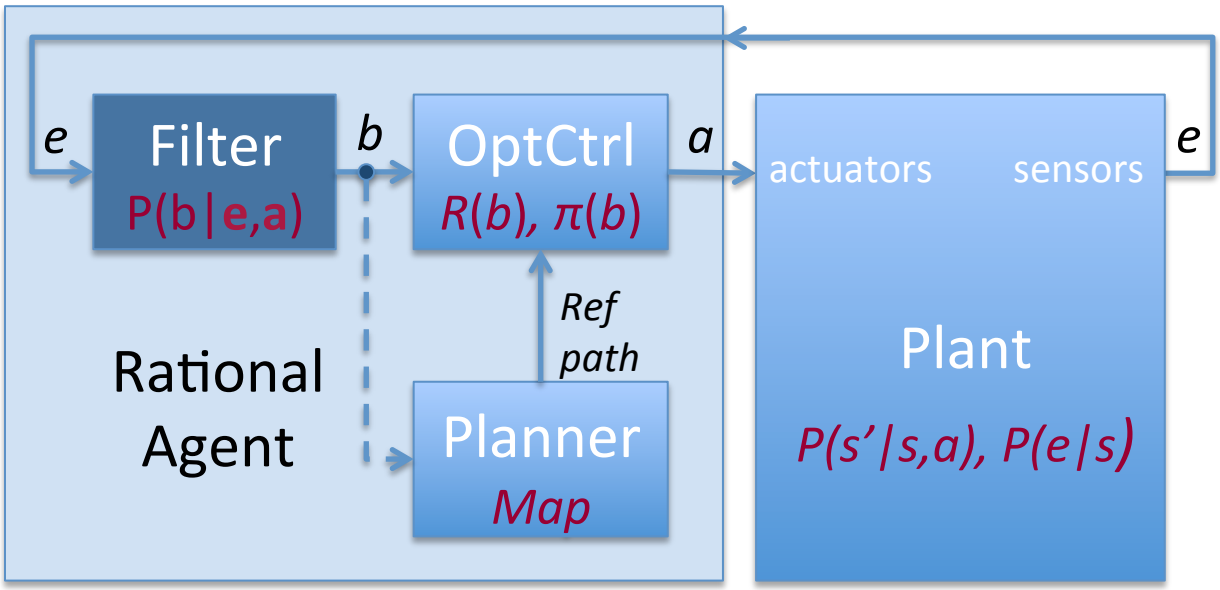
- ✓ Representation
- ✓ Conditional Independences
- Probabilistic Inference
  - Enumeration (exact, exponential complexity)
  - Variable elimination (exact, worst-case exponential complexity, often better)
  - Probabilistic inference is NP-complete
  - Sampling (approximate)

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

Exact Algorithms



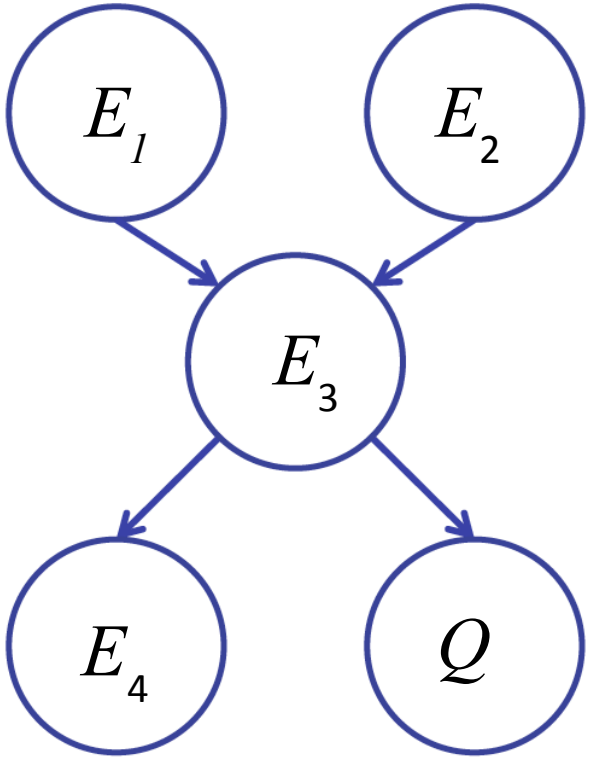
Inference

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

Inference

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

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



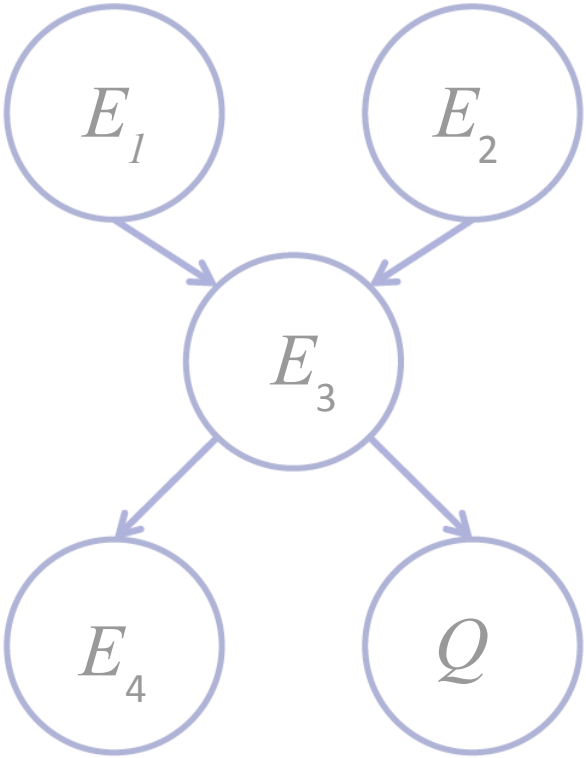
Inference

- 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

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

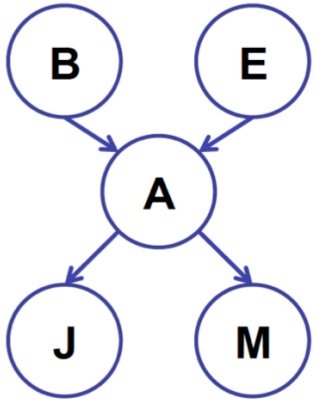
Inference by Enumeration

- 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

$$\mathbf{P}(X \mid \mathbf{e}) = \alpha\, \mathbf{P}(X, \mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y})$$

- X - query-, **Y** hidden-, **E** evidence - RVs, respectively

Example: Enumeration



- BN: complete description of the JPD
  - Sums of products of conditional probabilities

$$\boldsymbol{P}(B \mid +j,+m) = \alpha\ \boldsymbol{P}(B,+j,+m) = \alpha \sum_e \sum_a \boldsymbol{P}(B,+j,+m,e,a)$$

  - For simplicity, take $B = +b$

$$P(+b \mid +j,+m) = \alpha\ P(B,+j,+m) = \alpha \sum_e \sum_a P(+b,+j,+m,e,a)$$

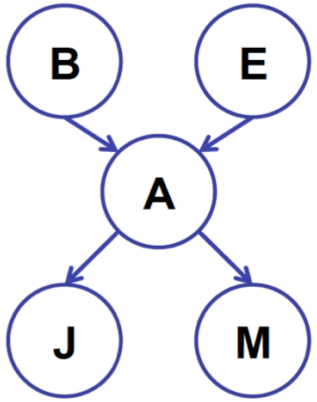
$$= \alpha \sum_e \sum_a P(+b)\,P(e)\,P(a \mid +b,e)\,P(+j \mid a)\,P(+m \mid a)$$

  - Compute by adding the 4 terms, each with 5 factors

–Worst case: Sum out n-1 (boolean) vars $O(n2^n)$

Example: Enumeration

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



$$P(+b,+j,+m) =$$

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

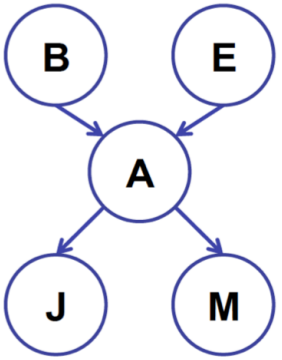
$$P(+b)P(+e)P(-a|+b,+e)P(+j|-a)P(+m|-a) +$$

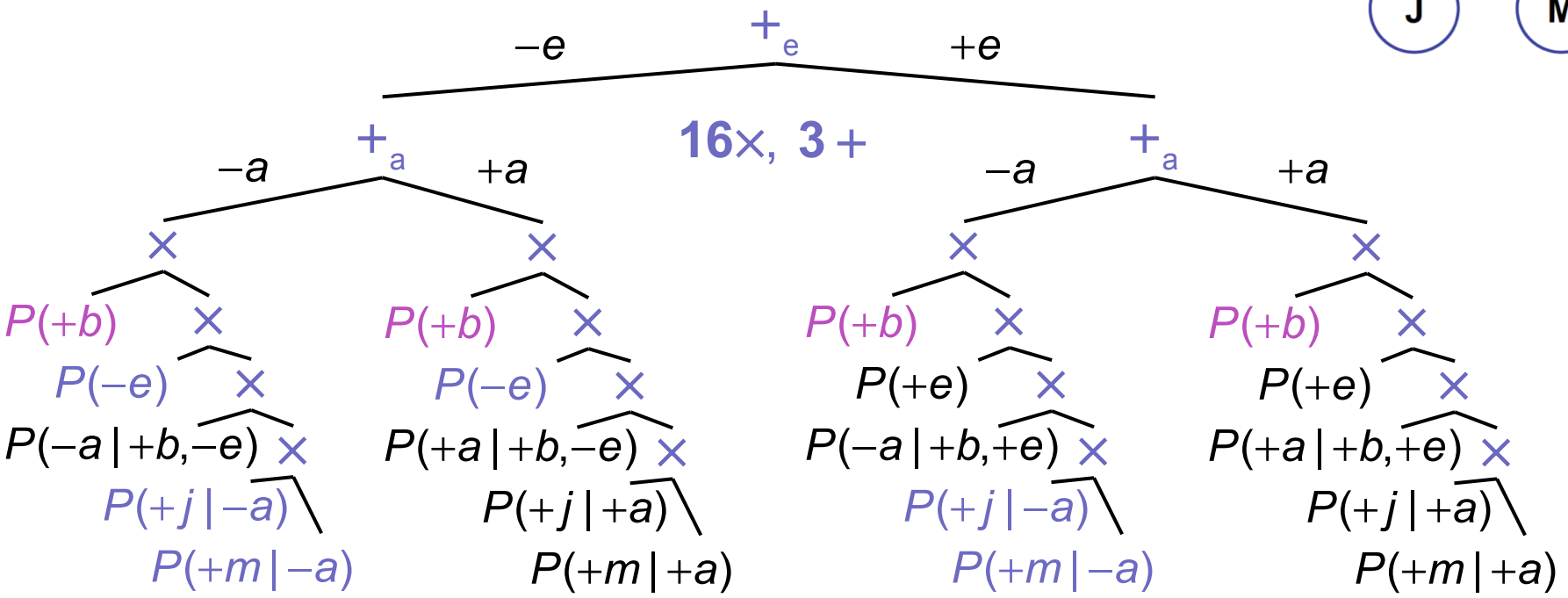
$$P(+b)P(-e)P(+a|+b,-e)P(+j|+a)P(+m|+a) +$$

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

Enumeration Complexity

$$P(B,+j,+m) = \sum_e \sum_a P(+b)\,P(e)\,P(a\,|\,+b,e)\,P(+j\,|\,a)\,P(+m\,|\,a)$$

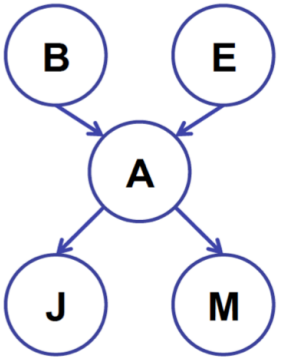


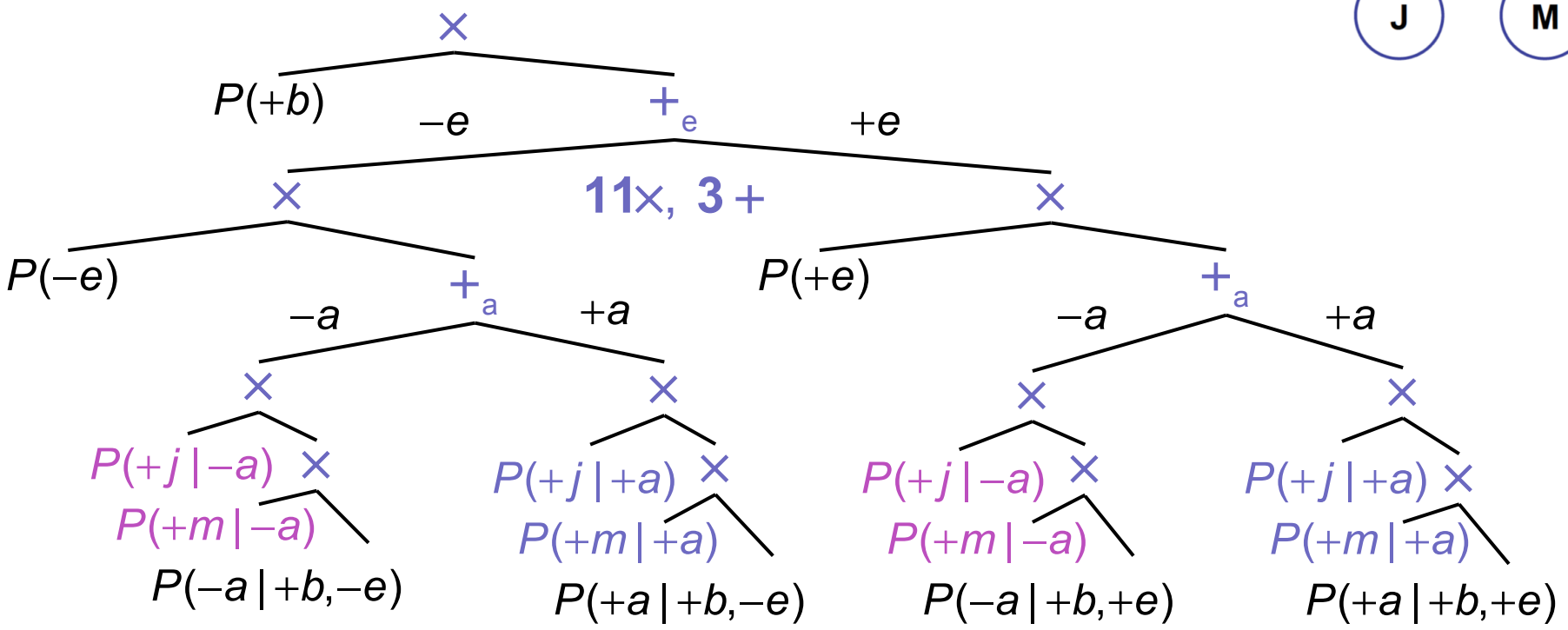


Evaluated by DFS traversal of the syntax tree $O(n2^n)$

Enumeration Improvement

$$P(+b,+j,+m) = P(+b)\sum_{e} P(e)\sum_{a} P(a\,|\,+b,e)\,P(+j\,|\,a)\,P(+m\,|\,a)$$





Evaluated by DFS traversal of the syntax tree $O(2^n)$

Enumeration Algorithm

function Enumeration-Ask (X,**e**,bn) returns $\widehat{P}(X|\mathbf{e})$

local $\mathbf{Q} = \varnothing$ // distribution over X initially empty

foreach ($x_i \in \text{domain}(X)$)

$\mathbf{Q}(x_i) = \text{Enumerate-All}(\text{bn.vars}, \mathbf{e}_{x_i})$ // $\mathbf{e}_{x_i} = \mathbf{e} : (X = x_i)$

return Normalize(**Q**)

function Enumerate-All (vars, **e**) returns $\mathbb{R}$

if $(\text{vars} = \varnothing)$ return 1.0

$Y = \text{First}(\text{vars})$

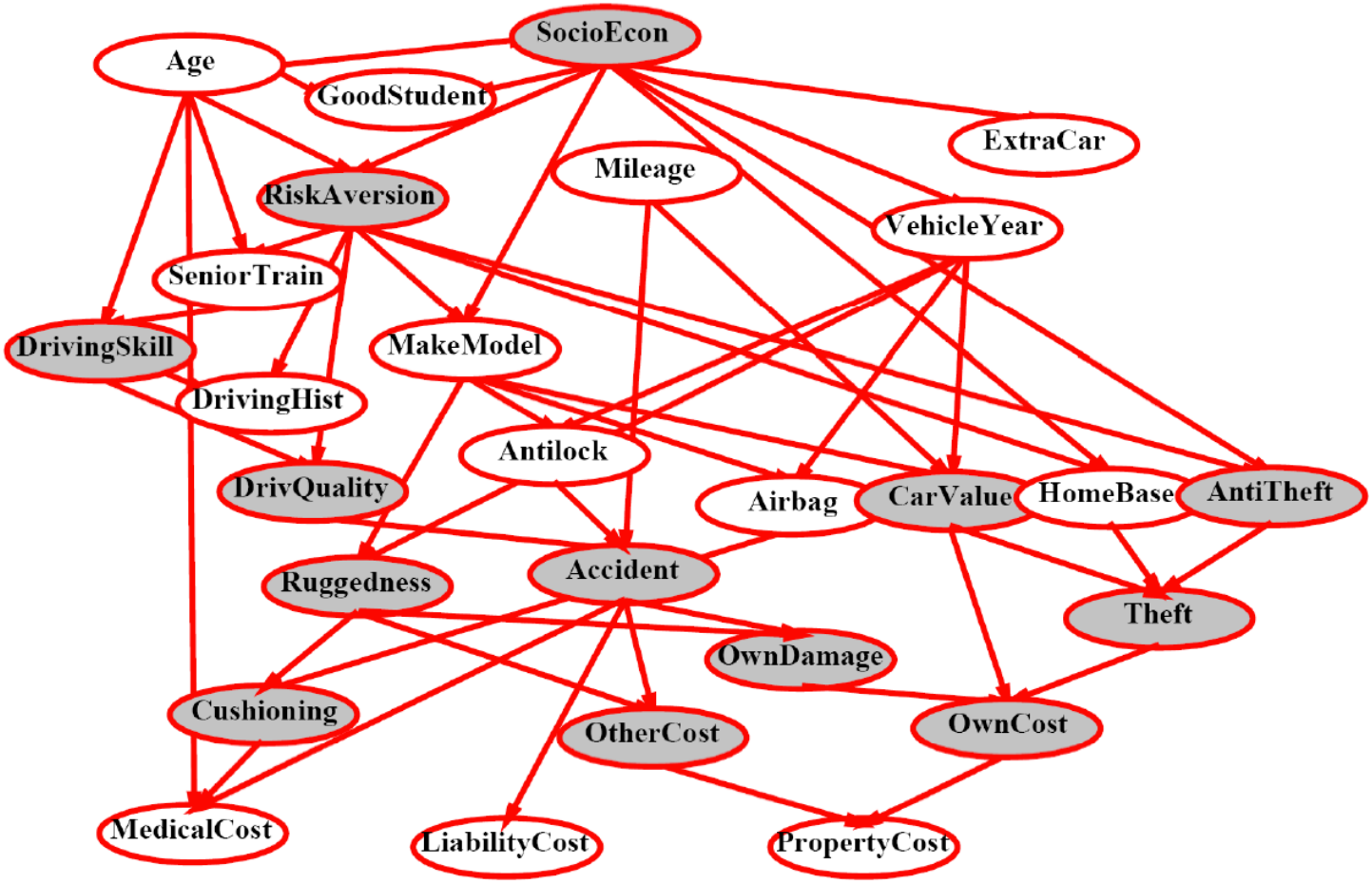
if $((y = \text{Val}(Y)) \in \mathbf{e})$

then return $P(y \mid \text{parents}(Y)) \times \text{Enumerate-All}(\text{Rest}(\text{vars}), \mathbf{e})$

else return $\sum_y P(y \mid \text{parents}(Y)) \times \text{Enumerate-All}(\text{Rest}(\text{vars}), \mathbf{e}_y)$

Evaluated by DFS traversal of the syntax tree $O(2^n)$

Inference by Enumeration?



Variable-Elimination Algorithm

- Eliminate repeated computation of sub-expressions
- Idea is very simple: Save intermediate results
  - This is a form of dynamic programming (bottom up, save IR)
- Various approaches: Variable Elimination (simplest)
  - Query/hidden variables not instantiated. Factor (matrix) Zoo
  - Expressions evaluated right-to-left, that is, bottom-up
  - Intermediate results are stored
  - Summations only over expressions depending on a variable

$$\mathbf{P}(B \mid +j,+m) = \alpha \underbrace{\mathbf{P}(B)}_{\mathbf{f}_1(B)} \sum_e \underbrace{P(e)}_{\mathbf{f}_2(E)} \sum_a \underbrace{\mathbf{P}(a \mid B,+e)}_{\mathbf{f}_3(A,B,E)} \underbrace{P(+j \mid a)}_{\mathbf{f}_4(A)} \underbrace{P(+m \mid +a)}_{\mathbf{f}_5(A)}$$

$$\mathbf{P}(B \mid +j,+m) = \alpha \, \mathbf{f}_1(B) \times \sum_e \mathbf{f}_2(E) \times \sum_a \mathbf{f}_3(A,B,E) \times \mathbf{f}_4(A) \times \mathbf{f}_5(A)$$

Variable-Elimination Algorithm

$$\mathbf{P}(B \mid +j,+m) = \alpha \underbrace{\mathbf{P}(B)}_{\mathbf{f}_1(B)} \sum_e \underbrace{P(e)}_{\mathbf{f}_2(E)} \sum_a \underbrace{\mathbf{P}(a \mid B,+e)}_{\mathbf{f}_3(A,B,E)} \underbrace{P(+j \mid a)}_{\mathbf{f}_4(A)} \underbrace{P(+m \mid +a)}_{\mathbf{f}_5(A)}$$

$$\mathbf{f}_5(A) = [\, P(+m \mid -a) : 0.70,\ P(+m \mid +a) : 0.01 \,]$$

$$\mathbf{f}_4(A) = [\, P(+j \mid -a) : 0.90,\ P(+j \mid +a) : 0.05 \,]$$

$$\mathbf{f}_2(E) = [P(-e) : 0.998,\ P(+e) : 0.002]$$

$$\mathbf{f}_1(B) = [P(-b) : 0.999,\ P(+b) : 0.001]$$

$$\mathbf{f}_6(B,E) = \sum_a \mathbf{f}_3(A,B,E) \times \mathbf{f}_4(A) \times \mathbf{f}_5(A)$$

$$= \mathbf{f}_3(+a,B,E) \times \mathbf{f}_4(+a) \times \mathbf{f}_5(+a) + \mathbf{f}_3(-a,B,E) \times \mathbf{f}_4(-a) \times \mathbf{f}_5(-a)$$

$$\mathbf{P}(B \mid +j,+m) = \alpha\ \mathbf{f}_1(B) \times \sum_e \mathbf{f}_2(E) \times \mathbf{f}_6(B,E)$$

$$\mathbf{f}_7(B) = \sum_e \mathbf{f}_2(E) \times \mathbf{f}_6(B,E) = \mathbf{f}_2(+e) \times \mathbf{f}_6(B,+e) + \mathbf{f}_2(-e) \times \mathbf{f}_6(B,-e)$$

$$\mathbf{P}(B \mid +j,+m) = \alpha\ \mathbf{f}_1(B) \times \mathbf{f}_7(B)$$

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

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

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

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
hot	rain	0.2
cold	sun	0.4
cold	rain	0.6

$P(W|hot)$

$P(W|cold)$

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
hot	rain	0.2
cold	sun	0.4
cold	rain	0.6

$P(W|hot)$

$P(W|cold)$

- 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

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
cold	rain	0.6

$P(rain|hot)$

$P(rain|cold)$

Factor Zoo III

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

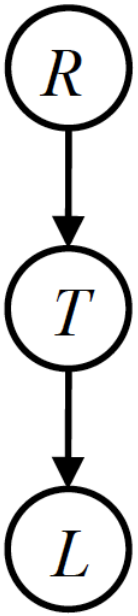
$P(rain|T)$

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

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

$P(L|T)$

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

Variable Elimination Outline

- Track objects called factors

Variable Elimination Outline

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

$P(R)$

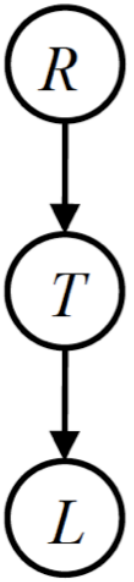
+r	0.1
-r	0.9

$P(T|R)$

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

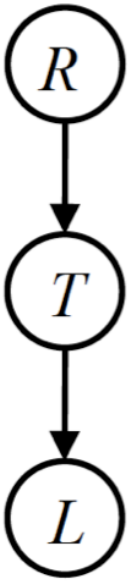
$P(L|T)$

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



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

$P(T|R)$

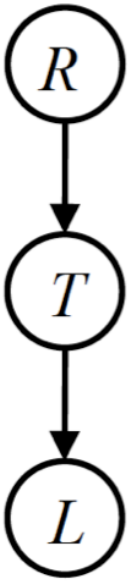
+r	+t	0.8
+r	-t	0.2
-r	+t	0.1
-r	-t	0.9

$P(+\ell|T)$

+t	+l	0.3
-t	+l	0.1

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

Operation 1: Join Factors

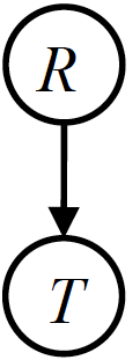
- First basic operation: joining factors

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

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



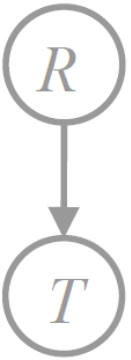
$$P(R) \times P(T|R)$$

+r	0.1
-r	0.9

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

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



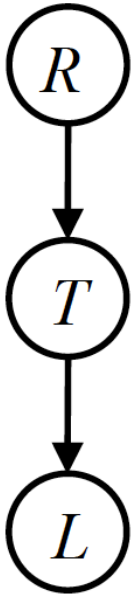
$$P(R) \times P(T|R) \rightarrow P(R,T)$$

+r	0.1
-r	0.9

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

+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

Example: Multiple Joins



$P(R)$

+r	0.1
-r	0.9

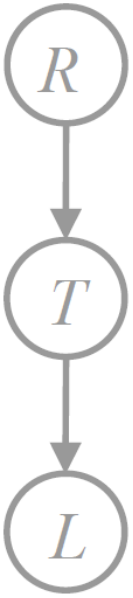
$P(T|R)$

+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

Example: Multiple Joins



$P(R)$

+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

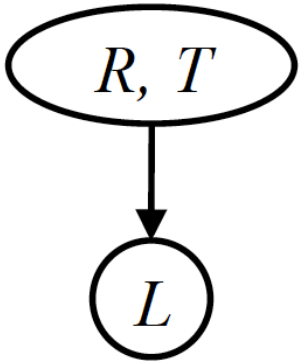
$P(L|T)$

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

Join R

$P(R,T)$

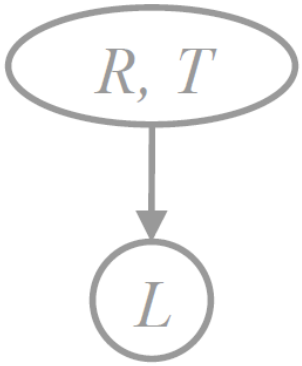
+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81



$P(L|T)$

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

Example: Multiple Joins



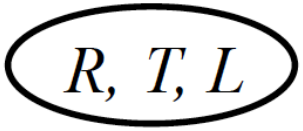
$P(R,T)$

+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

$P(L|T)$

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

Join T



$P(R,T,L)$

+r	+t	+l	0.024
+r	+t	-l	0.056
+r	-t	+l	0.002
+r	-t	-l	0.018
-r	+t	+l	0.027
-r	+t	-l	0.063
-r	-t	+l	0.081
-r	-t	-l	0.729

Operation 2: Eliminate

- Second basic operation: marginalization

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

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:

$P(R, T)$

+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

sum R

$P(T)$

+t	0.17
-t	0.83

Multiple Elimination

$P(R, T, L)$

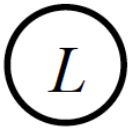
+r	+t	+l	0.024
+r	+t	-l	0.056
+r	-t	+l	0.002
+r	-t	-l	0.018
-r	+t	+l	0.027
-r	+t	-l	0.063
-r	-t	+l	0.081
-r	-t	-l	0.729

Sum out R

$P(T, L)$

+t	+l	0.051
+t	-l	0.119
-t	+l	0.083
-t	-l	0.747

Multiple Elimination



$P(R, T, L)$

+r	+t	+l	0.024
+r	+t	-l	0.056
+r	-t	+l	0.002
+r	-t	-l	0.018
-r	+t	+l	0.027
-r	+t	-l	0.063
-r	-t	+l	0.081
-r	-t	-l	0.729

Sum out R

$P(T, L)$

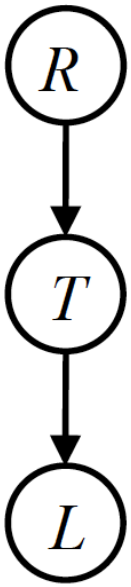
+t	+l	0.051
+t	-l	0.119
-t	+l	0.083
-t	-l	0.747

Sum out T

$P(L)$

+l	0.134
-l	0.886

P(L) : Marginalizing Early!



$P(R)$

+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

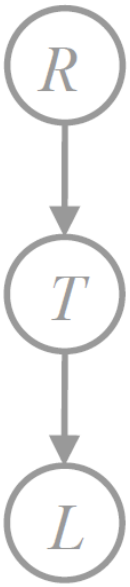
$P(L|T)$

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

P(L) : Marginalizing Early!

$P(R)$

+r	0.1
-r	0.9



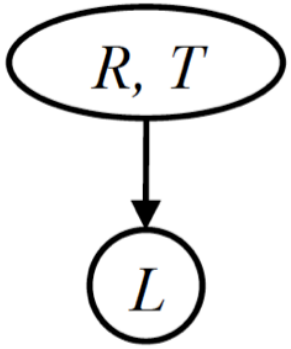
$P(T|R)$

+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

Join R



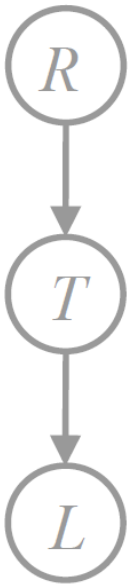
$P(R,T)$

+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

$P(L|T)$

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

P(L) : Marginalizing Early!



$P(R)$

+r	0.1
-r	0.9

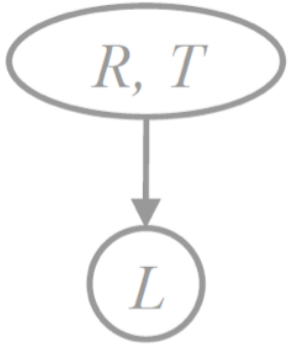
$P(T|R)$

+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

Join R



$P(R,T)$

+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

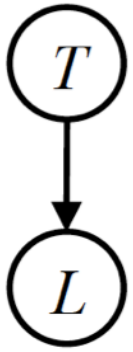
$P(L|T)$

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

Sum out R

$P(T)$

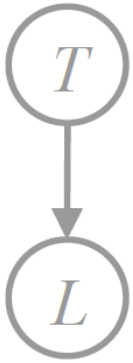
+t	0.17
-t	0.83



$P(L|T)$

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

Marginalizing Early (aka VE*)



Join T

$P(T)$

+t	0.17
-t	0.83

$P(L|T)$

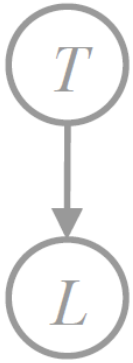
+t	+l	0.3
+t	-l	0.7
-t	+l	0.1
-t	-l	0.9

$P(T, L)$

+t	+l	0.051
+t	-l	0.119
-t	+l	0.083
-t	-l	0.747

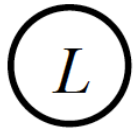
* VE is variable elimination

Marginalizing Early (aka VE*)



Join T

Sum out T



$P(T)$

+t	0.17
-t	0.83

$P(L|T)$

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

$P(T, L)$

+t	+l	0.051
+t	-l	0.119
-t	+l	0.083
-t	-l	0.747

$P(L)$

+l	0.134
-l	0.886

* VE is variable elimination

Evidence

- If evidence, start with factors that select that evidence

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

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

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:

$P(+r, L)$

+r	+l	0.026
+r	-l	0.074

Normalize

$P(L|+r)$

+l	0.26
-l	0.74

Evidence II

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

$P(+r, L)$

+r	+l	0.026
+r	-l	0.074

Normalize

$P(L|+r)$

+l	0.26
-l	0.74

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

General Variable Elimination

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

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)

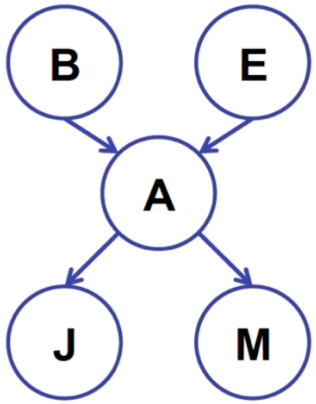
General Variable Elimination

- 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

General Variable Elimination

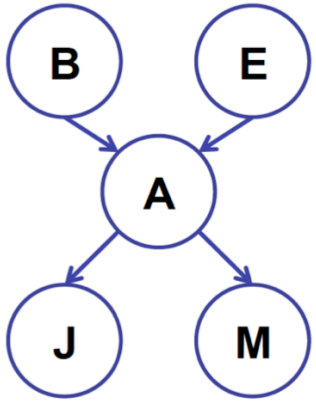
- Query: $P(Q|E_1 = e_1, \ldots E_k = e_k)$
- Start with initial factors:
  - Local CPTs (but instantiated by evidence)
- While there are still hidden variables (not Q or evidence):
  - Pick a hidden variable H
  - Join all factors mentioning H
  - Eliminate (sum out) H
- Join all remaining factors and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

Example



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

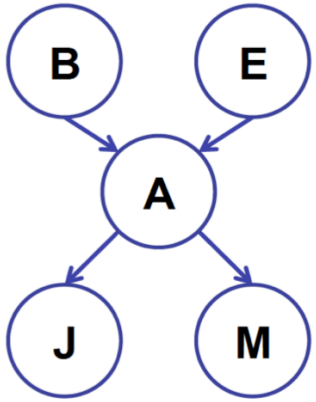
Example



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

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

Example

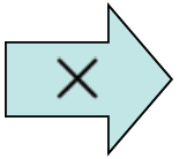


$$P(B|j,m) \propto P(B,j,m)$$

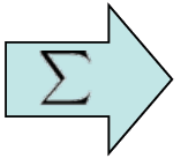
$P(B)$	$P(E)$	$P(A\|B,E)$	$P(j\|A)$	$P(m\|A)$

Choose A

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

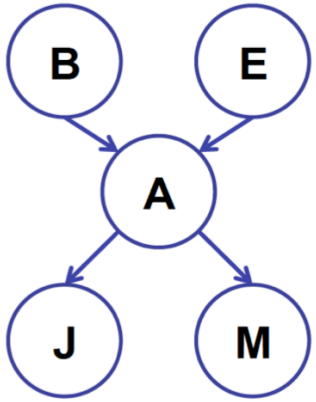


$P(j,m,A|B,E)$



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

Example



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

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

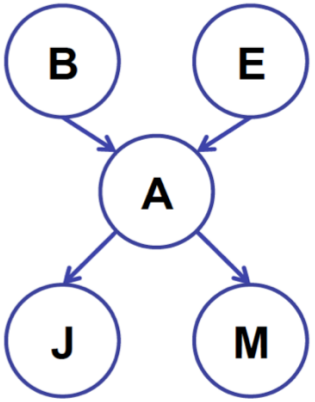
Choose A

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

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

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

Example

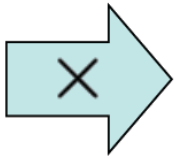


$P(B)$ $P(E)$ $P(j,m|B,E)$

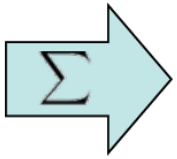
Choose E

$P(E)$

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

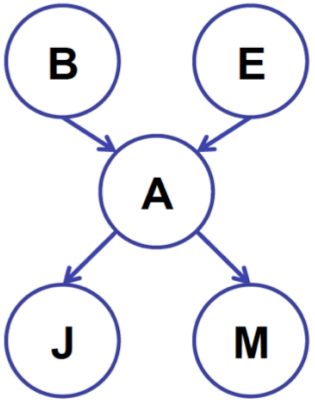


$P(j,m,E|B)$



$P(j,m|B)$

Example



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

Choose E

$P(E)$

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

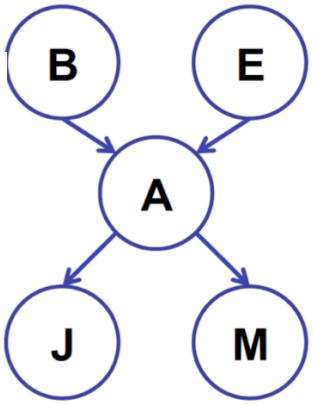
$P(B)$ $P(j,m|B)$

Finish with B

$P(B)$

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

Same Example in Equations



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

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

$$
\begin{aligned}
P(B|j,m) &\propto P(B,j,m) \\
&= \sum_{e,a} P(B,j,m,e,a) && \text{marginal can be obtained from joint by summing out} \\
&= \sum_{e,a} P(B)P(e)P(a|B,e)P(j|a)P(m|a) && \text{use Bayes' net joint distribution expression} \\
&= \sum_{e} P(B)P(e) \sum_{a} P(a|B,e)P(j|a)P(m|a) && \text{use } x*(y+z) = xy + xz \\
&= \sum_{e} P(B)P(e) f_1(B,e,j,m) && \text{joining on a, and then summing out gives f\_1} \\
&= P(B) \sum_{e} P(e) f_1(B,e,j,m) && x*(y+z) = xy + xz \\
&= P(B) f_2(B,j,m) && \text{joining on e, and then summing out gives f\_2}
\end{aligned}
$$

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

Variable-Elimination Algorithm

$$\mathbf{P}(B \mid +j,+m) = \alpha\ \mathbf{P}(B) \times \overbrace{\sum_e P(e) \times \underbrace{\sum_a \mathbf{P}(a \mid B,+e) \times P(+j \mid a) \times P(+m \mid +a)}_{\mathbf{f}_1(B,E)}}^{\mathbf{f}_2(B)}$$

$$\mathbf{f}_1(B,E) = \mathbf{P}(+a \mid B,E) \times P(+j \mid +a) \times P(+m \mid +a) + \mathbf{P}(-a \mid B,E) \times P(+j \mid -a) \times P(+m \mid -a) =$$

$$[\overbrace{0.95}^{+b+e}, \overbrace{0.94}^{+b-e}, \overbrace{0.29}^{-b+e}, \overbrace{0.01}^{-b-e}] \times \overbrace{0.9}^{P(+j \mid +a)} \times \overbrace{0.7}^{P(+m \mid +a)} + [\overbrace{0.05}^{+b+e}, \overbrace{0.06}^{+b-e}, \overbrace{0.71}^{-b+e}, \overbrace{0.99}^{-b-e}] \times \overbrace{0.05}^{P(+j \mid -a)} \times \overbrace{0.01}^{P(+m \mid -a)} =$$

$$= \overbrace{[\underbrace{0.598525}_{+b+e}, \underbrace{0.59223}_{+b-e}, \underbrace{0.183055}_{-b+e}, \underbrace{0.0011295}_{-b-e}]}^{\mathbf{f}_1(B,E)}$$

Variable-Elimination Algorithm

$$\mathbf{P}(B \mid +j,+m) = \alpha\ \mathbf{P}(B) \times \overbrace{\sum_e P(e) \times \underbrace{\sum_a \mathbf{P}(a \mid B,+e) \times P(+j \mid a) \times P(+m \mid +a)}_{\mathbf{f}_1(B,E)}}^{\mathbf{f}_2(B)}$$

$$\mathbf{f}_1(B,E) = [\overbrace{\underbrace{0.598525}_{+b+e}, \underbrace{0.59223}_{+b-e}, \underbrace{0.183055}_{-b+e}, \underbrace{0.0011295}_{-b-e}}^{f_1(B,E)}]$$

$$\mathbf{f}_2(B) = P(+e) \times \mathbf{f}_1(B,+e) + P(-e) \times \mathbf{f}_1(B,-e) =$$

$$= \overbrace{0.002}^{P(+e)} \times [\overbrace{\underbrace{0.598525}_{+b+e}, \underbrace{0.183055}_{-b+e}}^{f_1(B,+e)}] + \overbrace{0.998}^{P(-e)} \times [\overbrace{\underbrace{0.59223}_{+b-e}, \underbrace{0.0011295}_{-b-e}}^{f_1(B,-e)}]$$

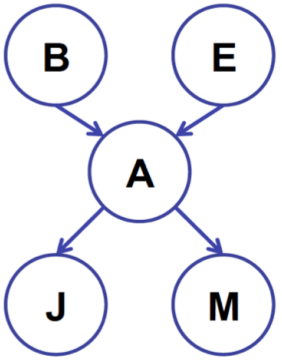
$$= [0.59224259,\ 0.001493351]$$

$$\mathbf{P}(B \mid +j,+m) = \alpha\ \mathbf{P}(B) \times \mathbf{f}_2(B)$$

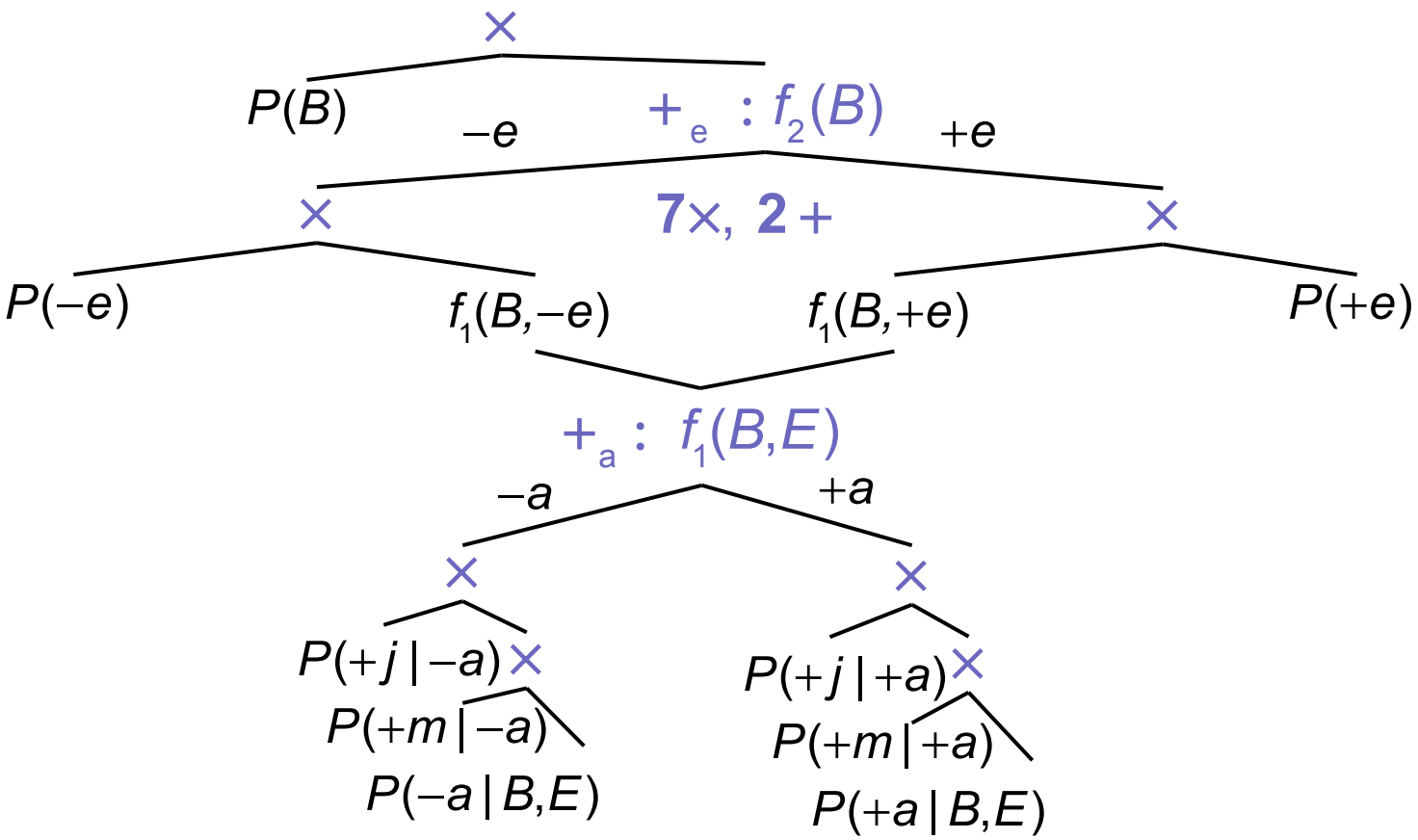
$$= \alpha\ [0.001,\ 0.999] \times [0.59224259,\ 0.001493351]$$

$$= \alpha\ [0.00059224,\ 0.00149185] \approx [0.2843,\ 0.7158]$$

Variable-Elimination Algorithm



$$P(B,+j,+m) = P(B)\sum_e P(e)\sum_a P(a\,|\,B,E)\,P(+j\,|\,a)\,P(+m\,|\,a)$$

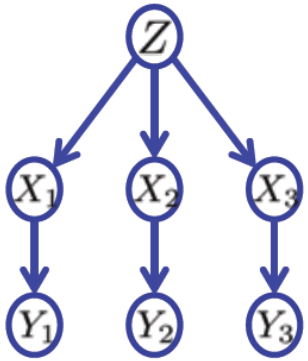


Variable-Elimination Algorithm

```
function Elimination-Ask (X,e,bn) returns P(X|e)
  local factors = ∅  // ordered set of factors
  foreach ( Y ∈ bn.vars )
    factors ∪= Make-Factor(Y, e, bn)  // add factor at the end
    if (Y ≠ X) then factors = Sum-Out(X, factors)
  return Normalize(Pointwise-Product(factors))
```

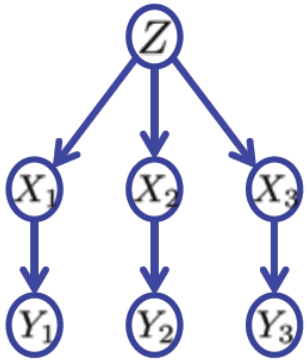
Another (bit more abstractly worked out) Variable Elimination Example

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



Another (bit more abstractly worked out) Variable Elimination Example

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

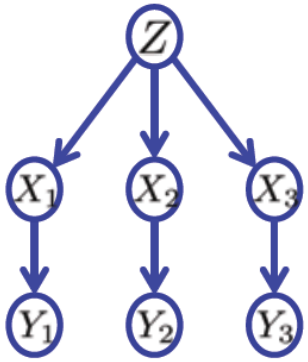


Start by inserting evidence, which gives the following initial factors:

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

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:

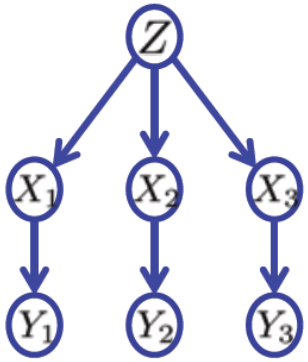
$$p(Z)p(X_1|Z)p(X_2|Z)p(X_3|Z)p(y_1|X_1)p(y_2|X_2)p(y_3|X_3)$$

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:

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

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:

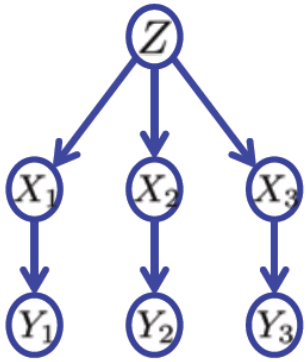
$$p(Z)f_1(Z, y_1)p(X_2|Z)p(X_3|Z)p(y_2|X_2)p(y_3|X_3)$$

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:

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

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:

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

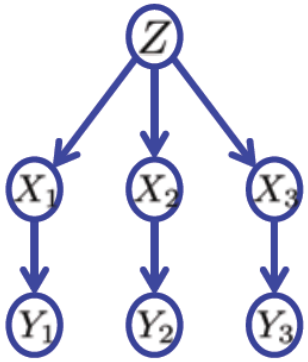
Eliminate X_2, this introduces the factor $f_2(Z, y_2) = \sum_{x_2} p(x_2|Z)p(y_2|x_2)$, and:

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

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:

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

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:

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

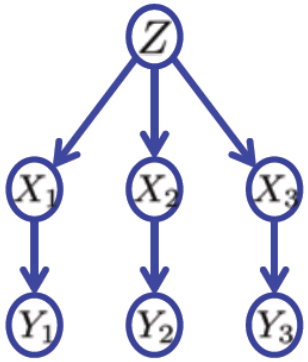
Eliminate Z, this introduces the factor
$f_3(y_1, y_2, y_3, X_3) = \sum_z p(z)f_1(z, y_1)f_2(z, y_2)p(X_3|z)p(y_3|X_3)$, and:

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

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

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:

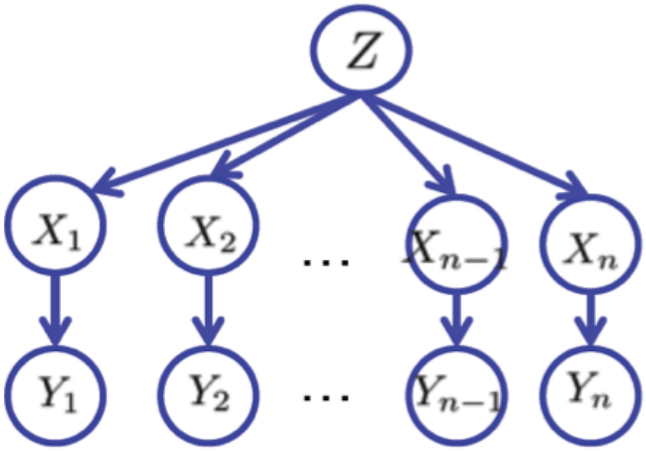
$$f_3(y_1, y_2, y_3, X_3)$$

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).

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:

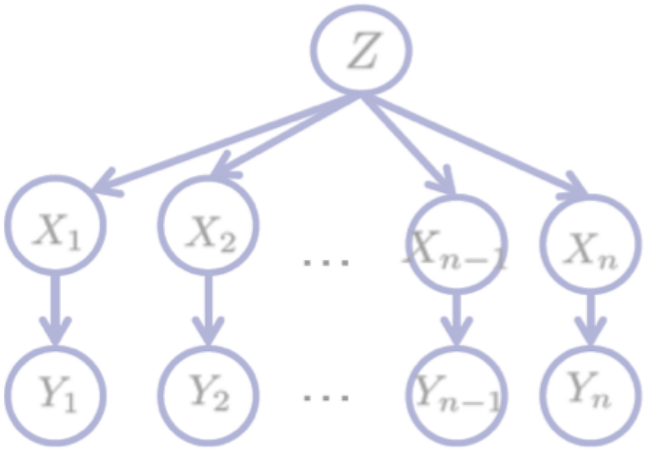


- $Z, X_1, \ldots, X_{n-1}$
- $X_1, \ldots, X_{n-1}, Z$

- What is the size of the maximum factor generated for each of the orderings?

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:

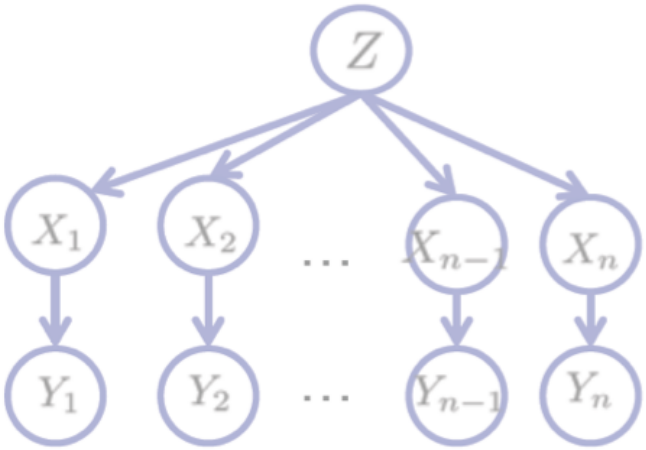


- $Z, X_1, \ldots, X_{n-1}$
- $X_1, \ldots, X_{n-1}, Z$

- What is the size of the maximum factor generated for each of the orderings?
- Answer: 2^n versus 2 (assuming binary)

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:



- $Z, X_1, \ldots, X_{n-1}$
- $X_1, \ldots, X_{n-1}, Z$

- 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.

Computational and Space Complexity of Variable Elimination

- The computational and space complexity of variable elimination is determined by the largest factor

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

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?

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!

Worst Case Complexity?

- Consider the 3-SAT clause:

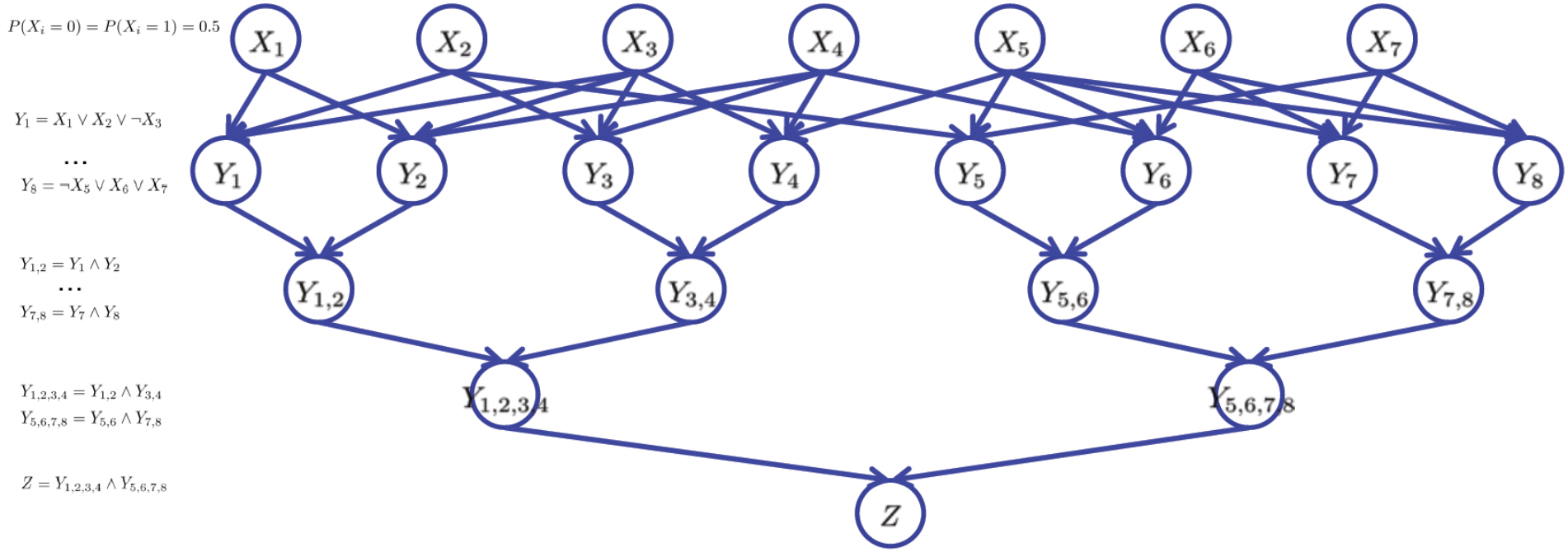
$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee \neg x_4) \wedge (x_2 \vee \neg x_2 \vee x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_5) \wedge (x_2 \vee x_5 \vee x_7) \wedge (x_4 \vee x_5 \vee x_6) \wedge (\neg x_5 \vee x_6 \vee \neg x_7) \wedge (\neg x_5 \vee \neg x_6 \vee x_7)$$

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:

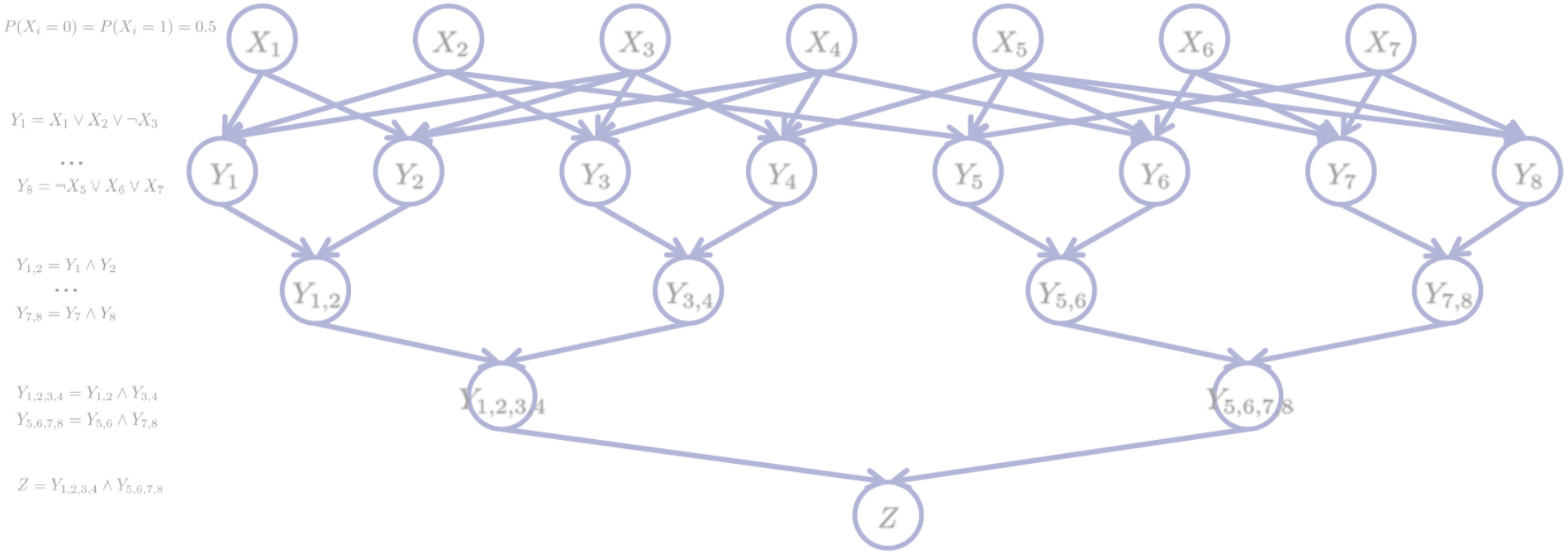


Worst Case Complexity?

- 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:



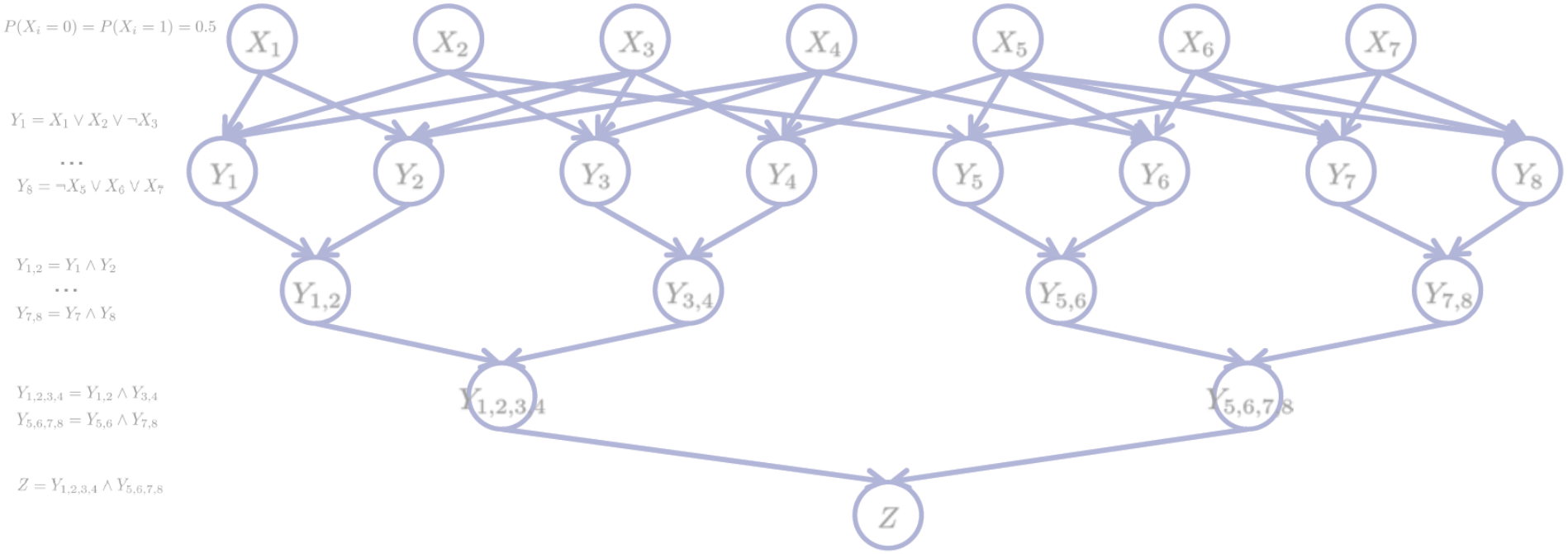
- If we can answer P(z) equal to zero or not, we answered whether the 3-SAT problem has a solution.

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:



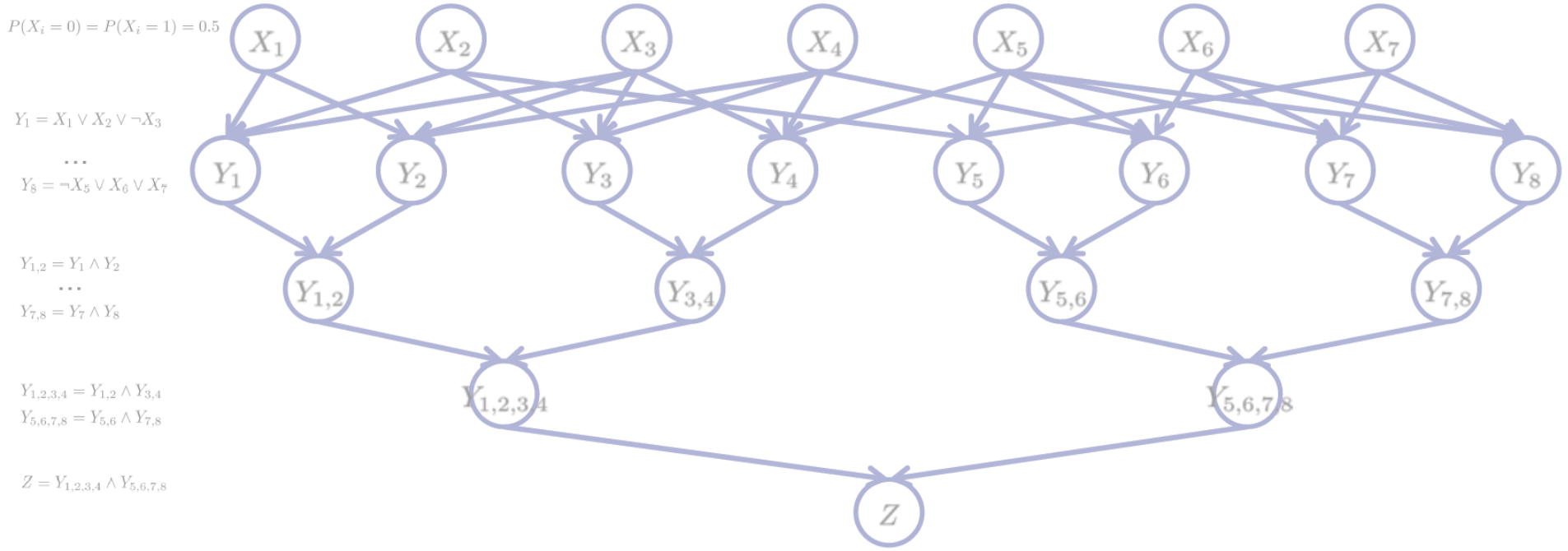
- If we can answer P(z) equal to zero or not, we answered whether the 3-SAT problem has a solution.
- Subtlety: why the cascaded version of the AND rather than feeding all OR clauses into a single AND?

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:



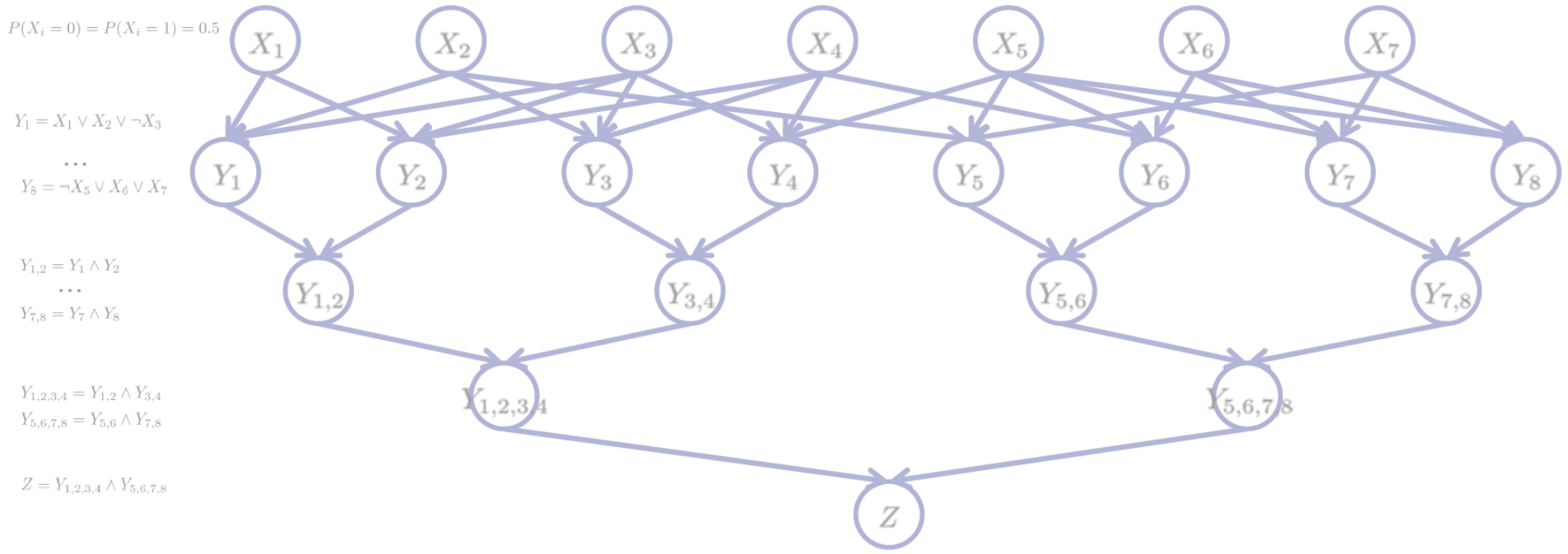
- If we can answer P(z) equal to zero or not, we answered whether the 3-SAT problem has a solution.
- Subtlety: why the cascaded version of the AND rather than feeding all OR clauses into a single AND? Answer: a single AND would have an exponentially large CPT, whereas with representation above the Bayes' net has small CPTs only.

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.

Polytrees

- A polytree is a directed graph with no undirected cycles

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!!

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

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

Polytrees

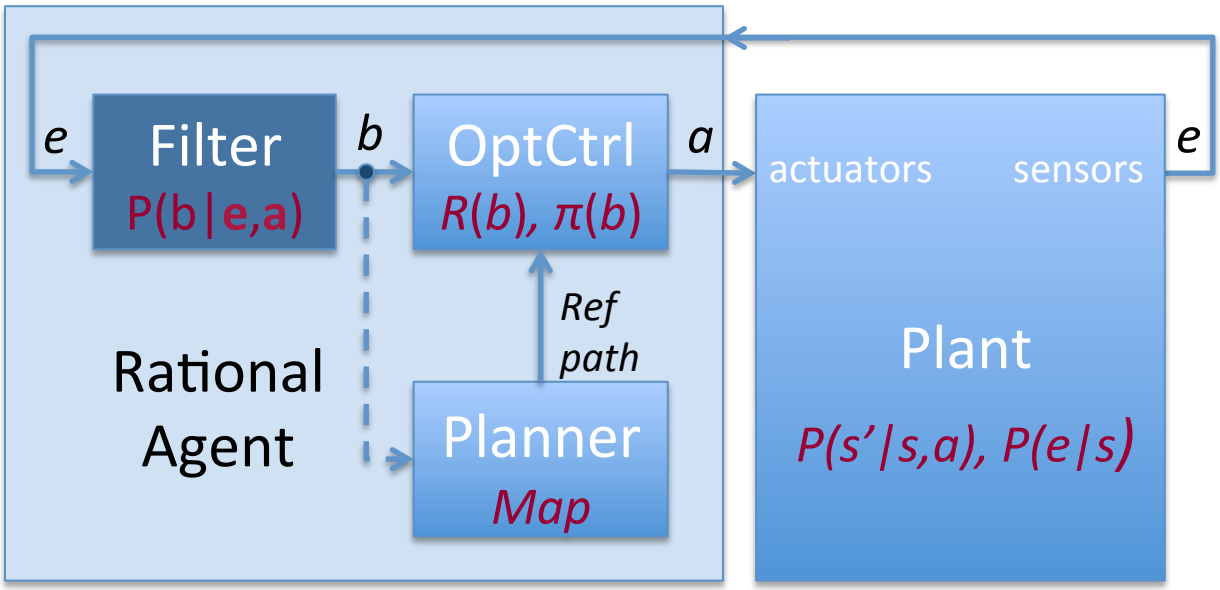
- A polytree is a directed graph with no undirected cycles
- For poly-trees you can always find an ordering that is efficient
  - Try it!!
- Cut-set conditioning for Bayes' net inference
  - Choose set of variables such that if removed only a polytree remains
  - Think about how the specifics would work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

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

Approximation Algorithms



Sampling

- Simulation has a name: sampling (e.g., predicting the weather, basketball games, …)

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

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?

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)

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

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

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

$$u \text{ is a uniform random value in } [0, 1]$$
$$\begin{aligned} \text{if} \quad & 0.0 \leq u < 0.6, w = \text{sun} \\ \text{if} \quad & 0.6 \leq u < 0.7, w = \text{rain} \\ \text{if} \quad & 0.7 \leq u < 1.0, w = \text{fog} \end{aligned}$$

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 $[0, 1]$

$$\begin{aligned} \text{if} \quad & 0.0 \leq u < 0.6, w = \text{sun} \\ \text{if} \quad & 0.6 \leq u < 0.7, w = \text{rain} \\ \text{if} \quad & 0.7 \leq u < 1.0, w = \text{fog} \end{aligned}$$

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

Sampling in Bayes' Nets

- Prior Sampling

Sampling in Bayes' Nets

- Prior Sampling
- Rejection Sampling

Sampling in Bayes' Nets

- Prior Sampling
- Rejection Sampling
- Likelihood Weighting

Sampling in Bayes' Nets

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

Prior Sampling

- Simplest kind of random-sampling process
- Generates events from a BN with no evidence
- Idea: Sample variables in a topological order
- PD of current table conditioned on parents sample

Prior-Sampling Algorithm

To generate one sample (OS) in a BN with n RVs

- Assume RVs $X_1,\ldots,X_n$ are topologically ordered (TO)
- Any topological order consistent with the BN dag

function Prior-Sample (TO bn) returns OS **x**

x = tuple with n elements

foreach X_i in bn

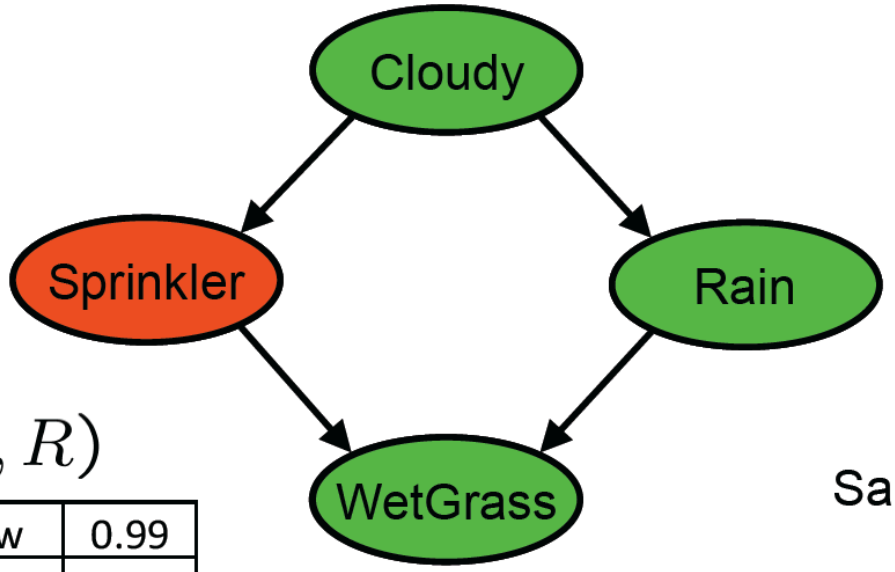
x[i] = randomSampleFrom($P(X_i \mid \text{Parents}(X_i)$ in **x**))

return **x**

Example: Wet-Grass BN

$P(C)$

+c	0.5
-c	0.5



$P(S|C)$

+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

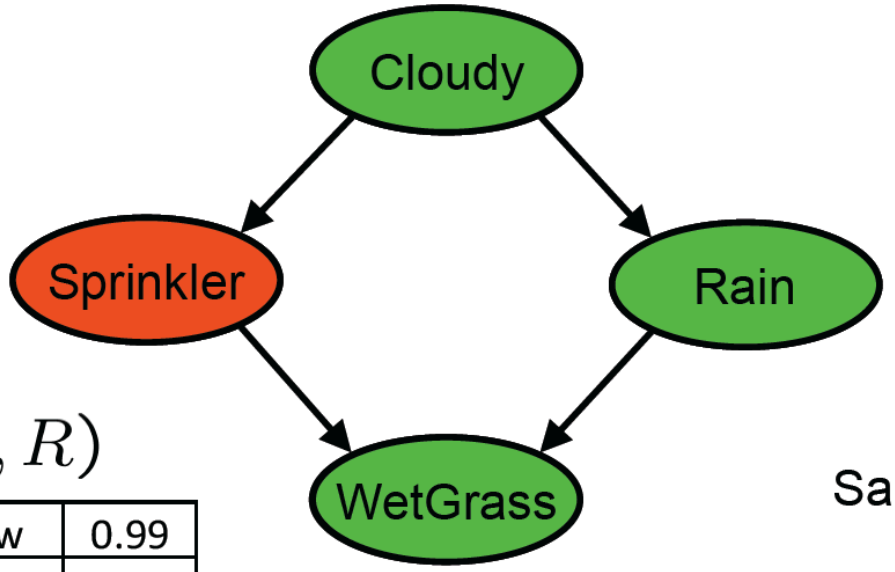
Samples:

Example: Wet-Grass BN

$P(C)$

+c	0.5
-c	0.5

+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

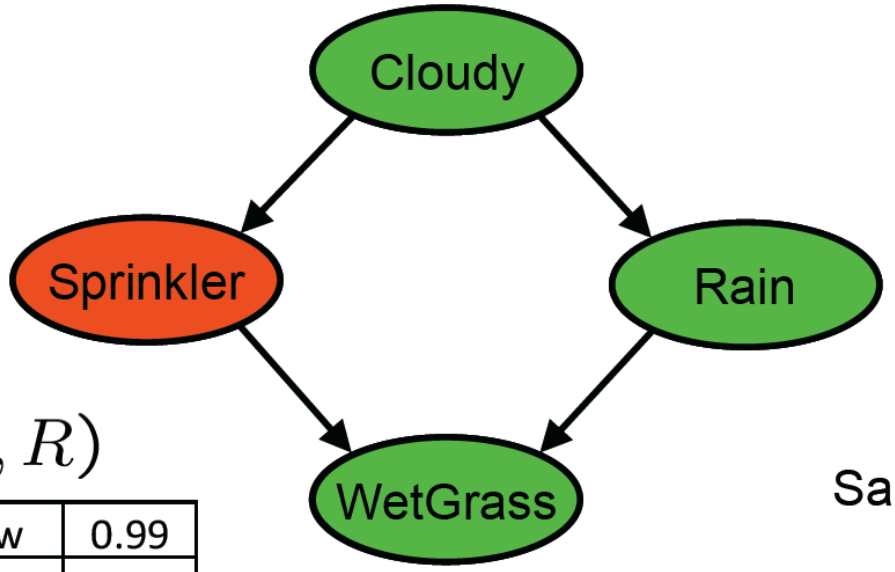
$P(W|S,R)$

+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

Samples:

+c,

Example: Wet-Grass BN



$P(C)$

+c	0.5
-c	0.5

+c

$P(S|\,+c\,)$

+c	+s	0.1
	-s	0.9

$P(R|\,+c\,)$

+c	+r	0.8
	-r	0.2

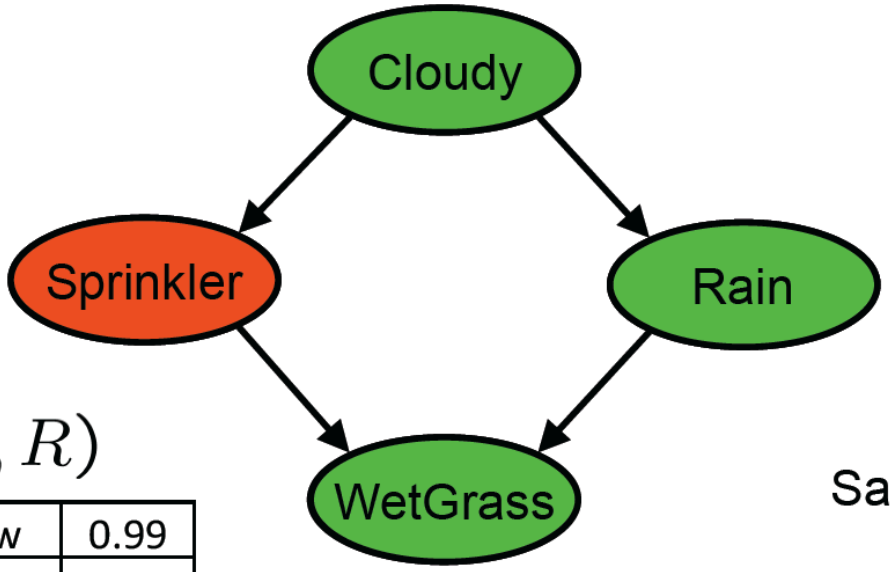
$P(W|S,R)$

+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

Samples:

+c,

Example: Wet-Grass BN



$P(C)$

+c	0.5
-c	0.5

+c

$P(S|\,+c\,)$

-s

+c	+s	0.1
	-s	0.9

$P(R|\,+c\,)$

+r

+c	+r	0.8
	-r	0.2

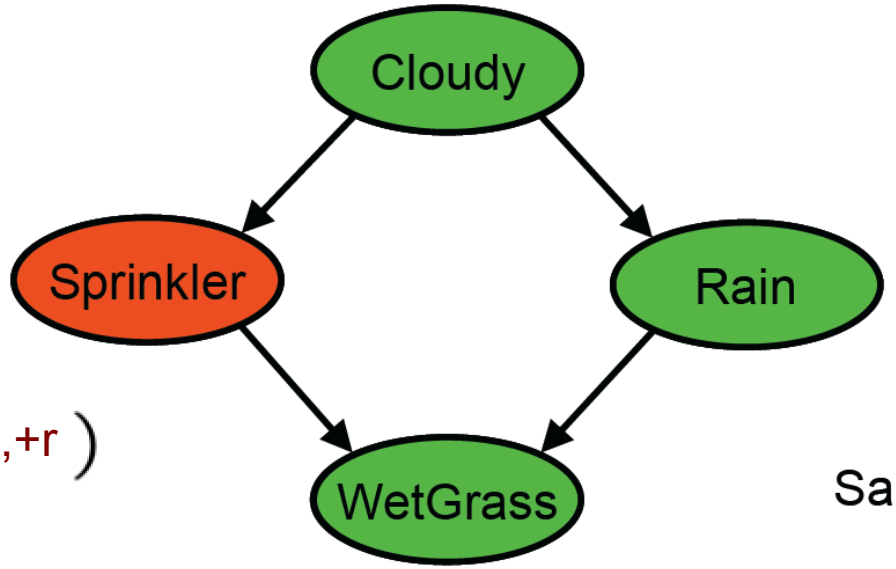
$P(W|S,R)$

+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

Samples:

+c, -s, +r,

Example: Wet-Grass BN



$P(C)$

+c	0.5
-c	0.5

+c

$P(S|\ +c)$

-s

+c	+s	0.1
	-s	0.9

$P(R|\ +c)$

+r

+c	+r	0.8
	-r	0.2

$P(W|\ -s,+r)$

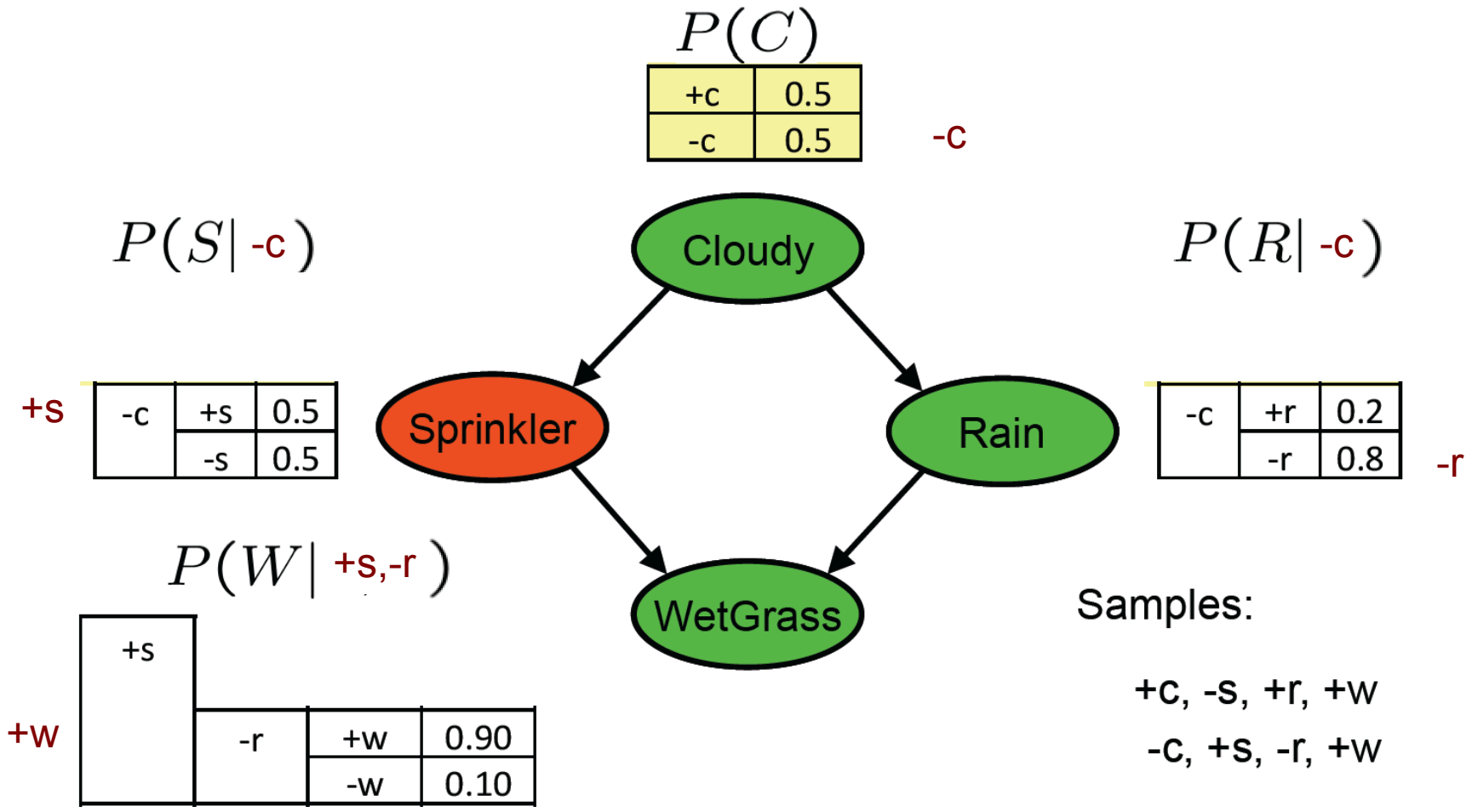
+w

-s	+r	+w	0.90
		-w	0.10

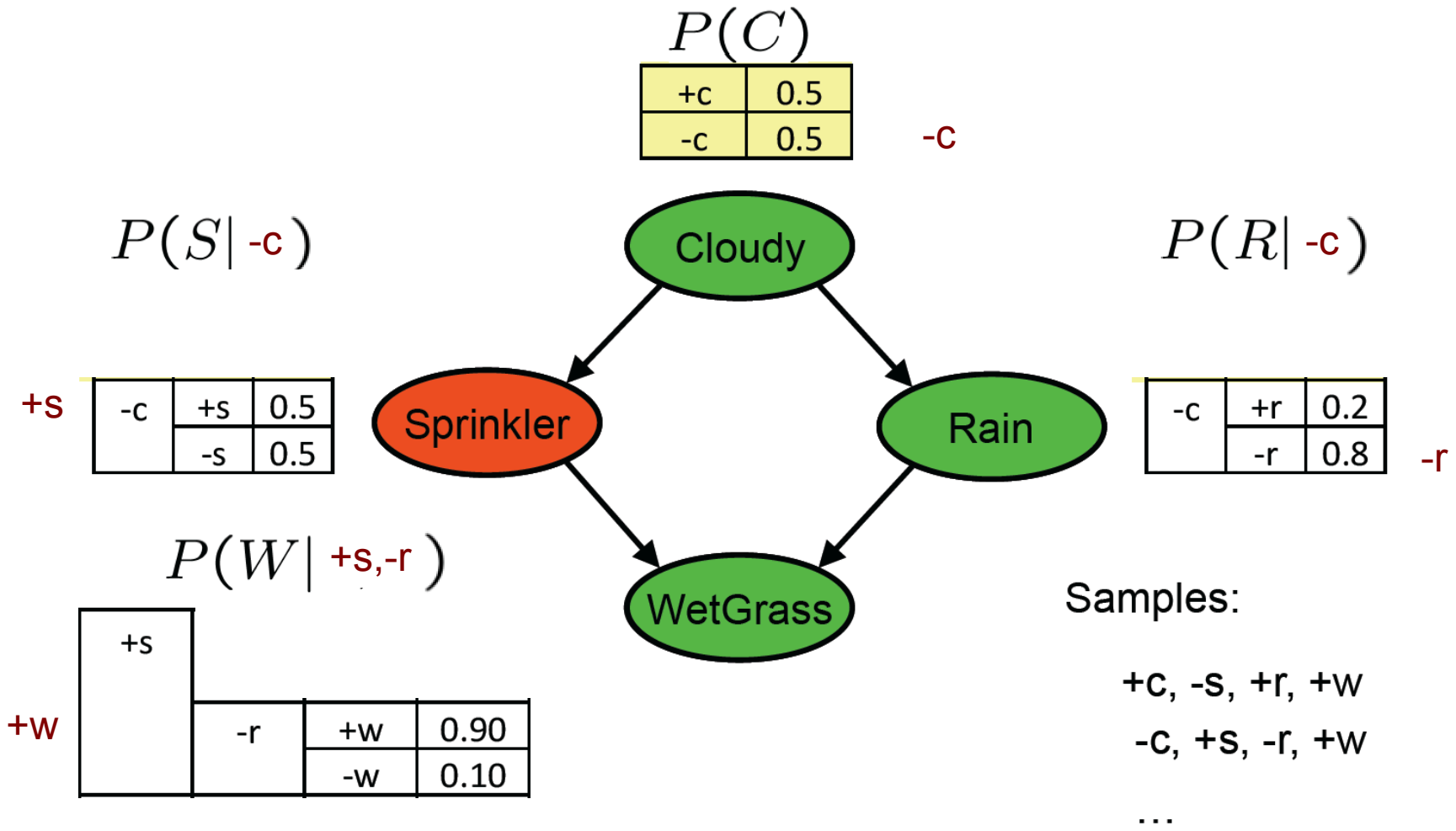
Samples:

+c, -s, +r, +w

Example: Wet-Grass BN



Example: Wet-Grass BN



Prior Sampling

- This process generates samples with probability:

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

…i.e. the BN's joint probability

Prior Sampling

- This process generates samples with probability:

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

…i.e. the BN's joint probability

- Let the number of samples of an event be $N_{PS}(x_1 \ldots x_n)$

Prior Sampling

- This process generates samples with probability:

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

…i.e. the BN's joint probability

- Let the number of samples of an event be $N_{PS}(x_1 \ldots x_n)$

- Then $\lim_{N \to \infty} \hat{P}(x_1, \ldots, x_n) = \lim_{N \to \infty} N_{PS}(x_1, \ldots, x_n)/N$

$$= S_{PS}(x_1, \ldots, x_n)$$

$$= P(x_1 \ldots x_n)$$

Prior Sampling

- This process generates samples with probability:

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

…i.e. the BN's joint probability

- Let the number of samples of an event be $N_{PS}(x_1 \ldots x_n)$

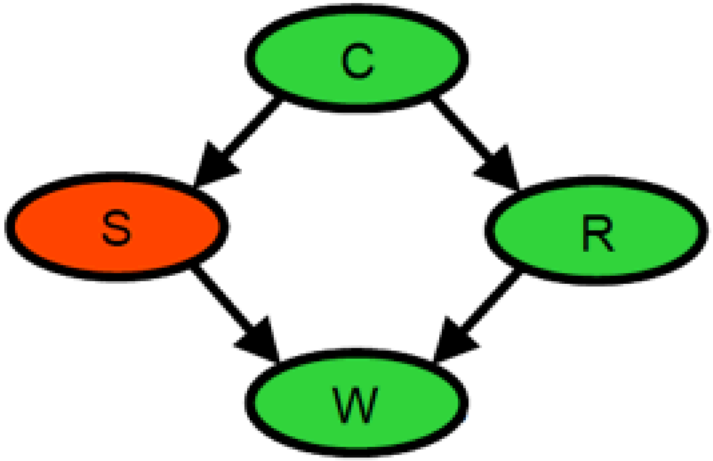
- Then $$\begin{aligned} \lim_{N\to\infty} \hat{P}(x_1, \ldots, x_n) &= \lim_{N\to\infty} N_{PS}(x_1, \ldots, x_n)/N \\ &= S_{PS}(x_1, \ldots, x_n) \\ &= P(x_1 \ldots x_n) \end{aligned}$$

- I.e., the sampling procedure is consistent

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



Estimate $\hat{P}(W)$ by counting how often w occurs

$$\mathbf{C} = \{+w: 4,\ -w: 1\}$$

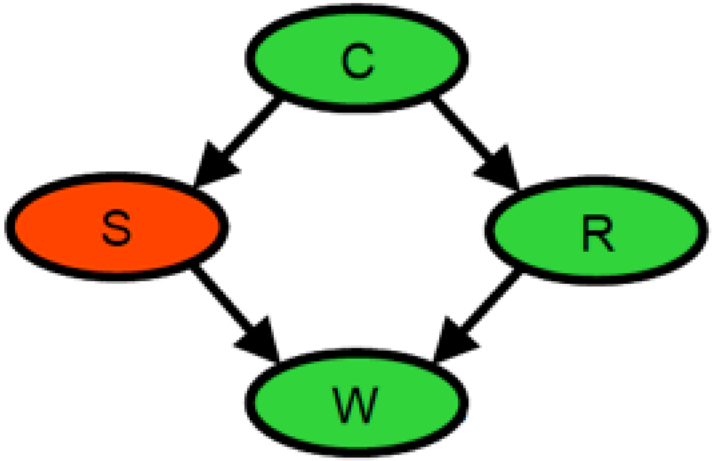
$$\hat{P}(W) = \text{Normalize}(\mathbf{C}) = [0.8, 0.2]$$

Estimate improves as number of samples increases

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



Can estimate anything else, too? What about:

$$\hat{P}(C \mid +w),\ \hat{P}(C \mid +r,+w),\ \hat{P}(C \mid -r,-w)$$

Rejection Sampling

- In its general form it is a method for:
  - Producing samples from a hard-to-sample distribution
  - Given an easy-to-sample distribution
- In its simplest form it is a method for:
  - Computing conditional probabilities given evidence
  - Determine $P(X \mid \mathbf{e})$
- The main idea is as follows:
  - Generate samples by using prior sampling
  - Reject samples which do not match the evidence
  - Estimate $\widehat{P}(X = x \mid \mathbf{e})$ by counting how often x occurs

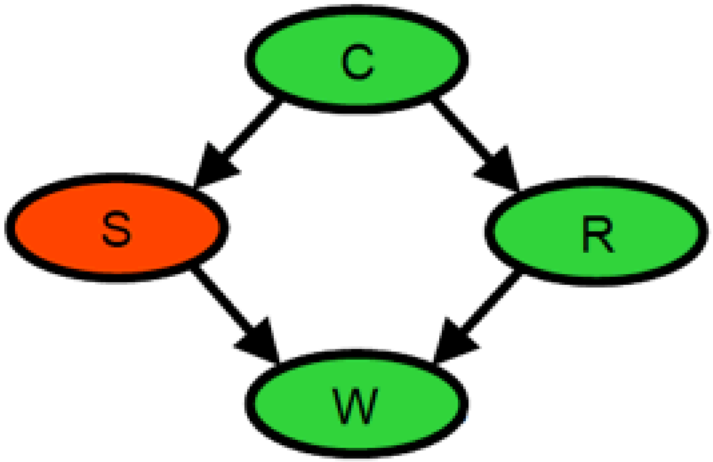
Rejection-Sampling Algorithm

```
function Rejection-Sampling (X,e,bn,N) returns P̂(X|e)
    local C = 0   // vector of counts for each value of X
    for i = 1:N {
        x = Prior-Sample(bn)   // let x be value of X in x
        if (x is consistent with e) C[x]++ }
    return Normalize(C)
```

Example: Compute P(C|+s)

Generate samples by using prior sampling

C	S	R	W
+c	-s	+r	+w
+c	+s	+r	+w
-c	+s	+r	-w
+c	-s	+r	+w
-c	-s	-r	+w

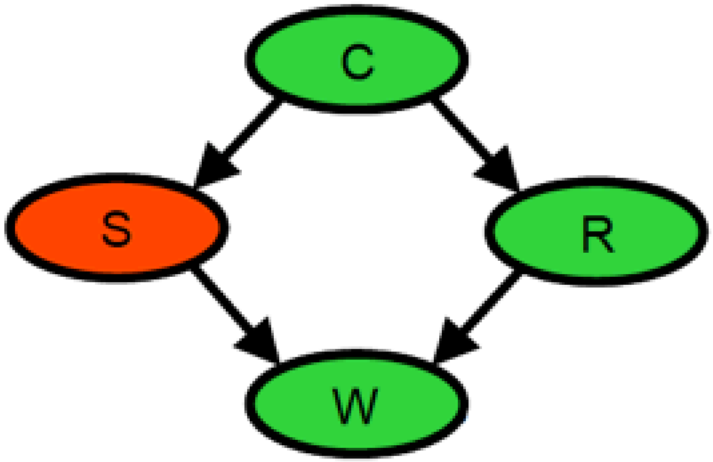


Example: Compute P(C|+s)

Generate samples by using prior sampling

Reject samples which do not match the evidence

C	S	R	W
+c	-s	+r	+w
+c	+s	+r	+w
-c	+s	+r	-w
+c	-s	+r	+w
-c	-s	-r	+w

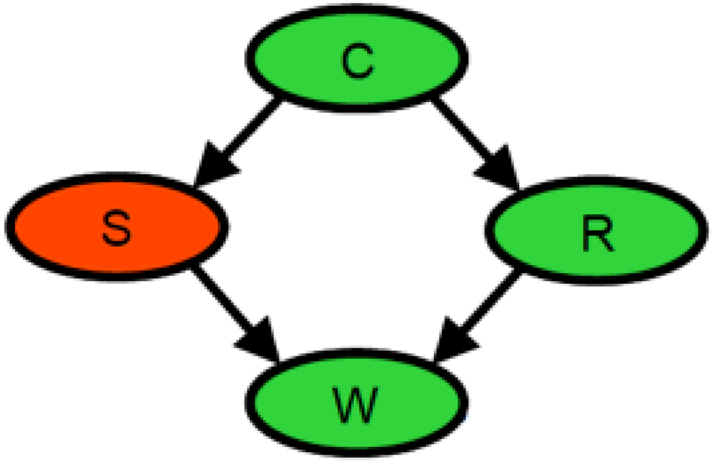


Example: Compute P(C|+s)

Generate samples by using prior sampling

Reject samples which do not match the evidence

C	S	R	W
+c	-s	+r	+w
+c	+s	+r	+w
-c	+s	+r	-w
+c	-s	+r	+w
-c	-s	-r	+w



Estimate $\hat{P}(C \mid +s)$ by counting how often x occurs

$$\mathbf{C} = \{+c : 1,\ -c : 1\}$$

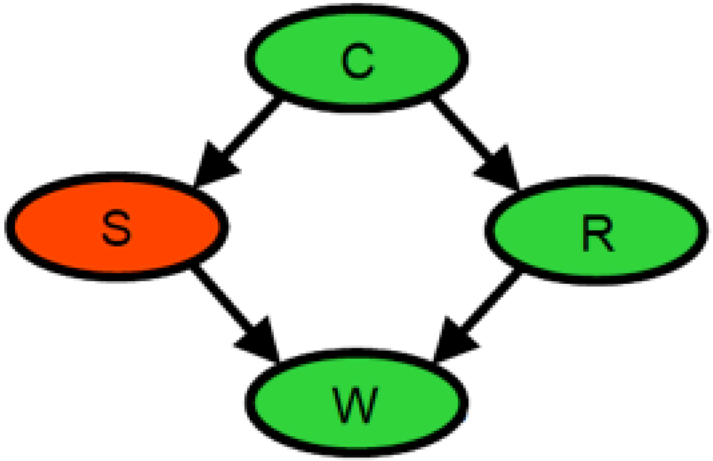
$$\hat{P}(C \mid +s) = \text{Normalize}(\mathbf{C}) = [0.5, 0.5]$$

Example: Compute P(C|+s)

Generate samples by using prior sampling

Reject samples which do not match the evidence

C	S	R	W
+c	-s	+r	+w
+c	+s	+r	+w
-c	+s	+r	-w
+c	-s	+r	+w
-c	-s	-r	+w



Estimate $\hat{P}(C|+s)$ by counting how often x occurs

It is also consistent for CPDs (that is, correct in the limit)

Rejection-Sampling: Consistency

Let $\hat{\mathbf{P}}(X \mid \mathbf{e})$ be the estimated distribution. From definition:

$$\hat{\mathbf{P}}(X \mid \mathbf{e}) = \alpha \mathbf{N}_{PS}(X, \mathbf{e}) = \frac{\mathbf{N}_{PS}(X, \mathbf{e})}{N_{PS}(\mathbf{e})}$$

Assuming that $P(x_1, \ldots, x_n) \approx N_{PS}(x_1, \ldots, x_n) / N$ we get:

$$\hat{\mathbf{P}}(X \mid \mathbf{e}) \approx \frac{\mathbf{P}(X, \mathbf{e})}{P(\mathbf{e})} = \mathbf{P}(X \mid \mathbf{e})$$

RS produces a consistent estimate of the true probability.

Standard deviation of error in each probability $\sigma \propto 1/\sqrt{N}$

Problems with Rejection Sampling

- It rejects too many samples:
  - No of samples consistent with evidence **e**
  - Drops exponentially with the no of evidence variables
- It is therefore unusable for complex problems
- It is however, similar to estimation in real world
  - To estimate P(Rain | RedNightSky = true):
  - Count how many times it rained
  - Ignoring the times the sky was not red
- This could take a lot of time

Likelihood Weighting

- Avoids the inefficiency of rejection sampling by:
  - Only generating events consistent with evidence **e**
- An instance of importance sampling technique
  - Tailored to Bayesian networks
- The main idea of LW is as follows:
  - Fix evidence variables **E** and sample **others** only
  - Not all events are equal however, so LW needs to
  - Weight events by how much they accord to evidence
    - As measured by $\prod_{e_i \in \mathbf{e}} P(e_i \mid Parents(e_i))$

Likelihood-Weighting Algorithm

function Likelihood-Weighting (X,**e**,bn,N) returns $\hat{P}(X|\mathbf{e})$
  local $\mathbf{W} = \mathbf{0}$ // vector of weighted counts for each value of X
  for i = 1:N {
    $(\mathbf{x}, w) = $ Weighted-Sample(bn,**e**)
    $\mathbf{W}[x] \mathrel{+}= w$ } // where x is value of X in **x**
  return Normalize(**W**)

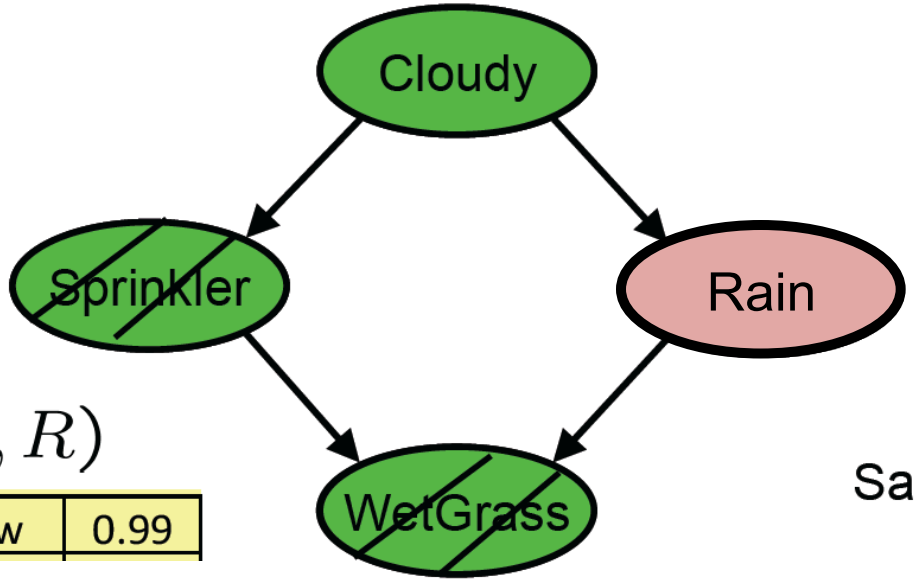
function Weighted-Sample (bn,**e**) returns (**x**,w)
  $\mathbf{x} = $ Initialize-From(**e**); $w = 1$
  foreach ($X_i \in$ bn) { // bn is the topological order $X_1 \ldots X_n$
    if (Evidence(X_i) $\wedge$ x_i = Value(X_i,**e**)) $w \times = P(X_i = x_i \mid \text{parents}(X_i))$
    else $\mathbf{x}[i] = $ Random-Sample-From $P(X_i = x_i \mid \text{parents}(X_i))$ }
  return (**x**,w)

Likelihood Weighting: P(Rain | +s,+w)

$P(C)$

+c	0.5
-c	0.5

+c



$P(S|C)$

+s

+c	+s	0.1

w = 1 x 0.1

$P(R|C)$

+c	+r	0.8
	-r	0.2

+r

$P(W|S,R)$

+s	+r	+w	0.99
	-r		
-s	+r		
	-r		

w = 0.1 x 0.99

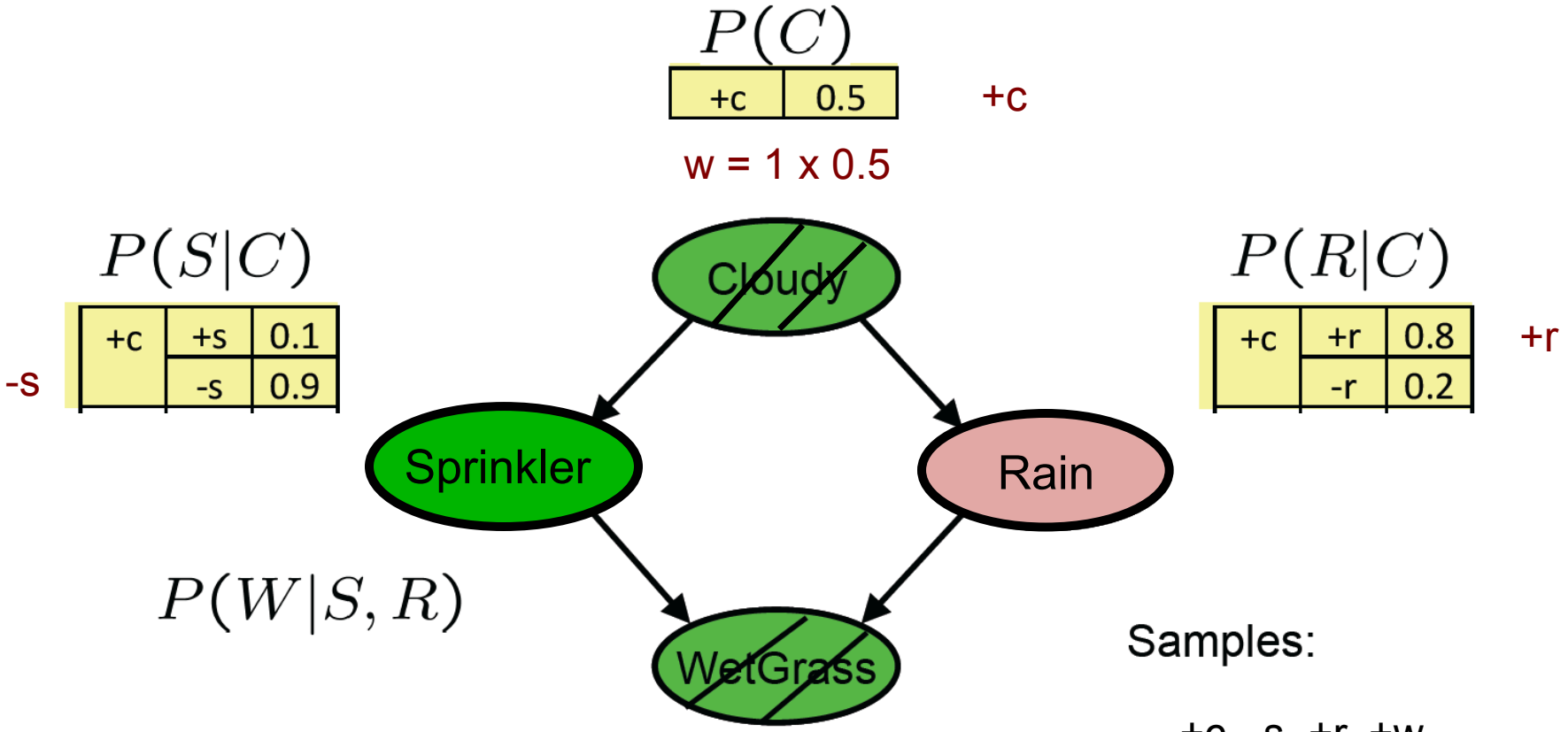
Samples:

+c, +s, +r, +w

…

+r with weight w = 0.099

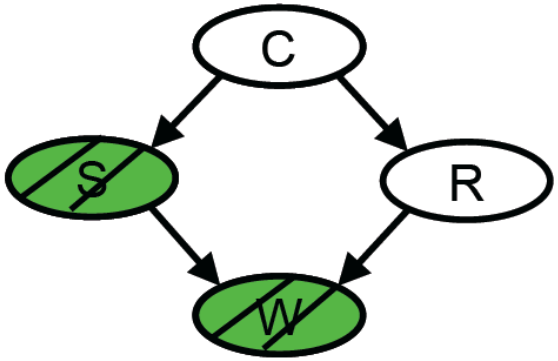
Likelihood Weighting: P(Rain | +c,+w)



Likelihood Weighting

- Sampling distribution if z sampled and e fixed evidence

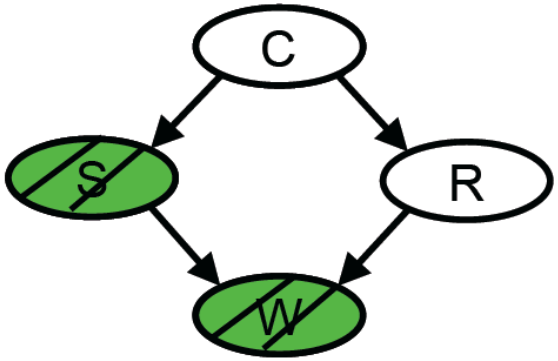
$$S_{WS}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{l} P(z_i | \text{Parents}(Z_i))$$



Likelihood Weighting

- Sampling distribution if z sampled and e fixed evidence

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



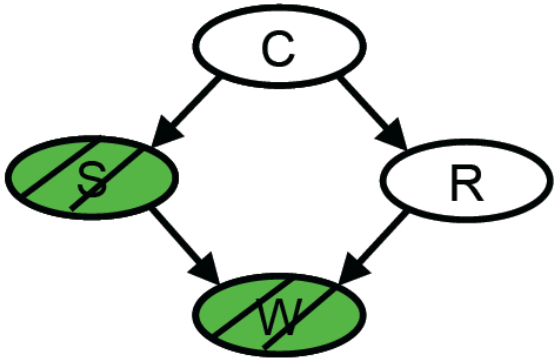
- Now, samples have weights

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

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

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

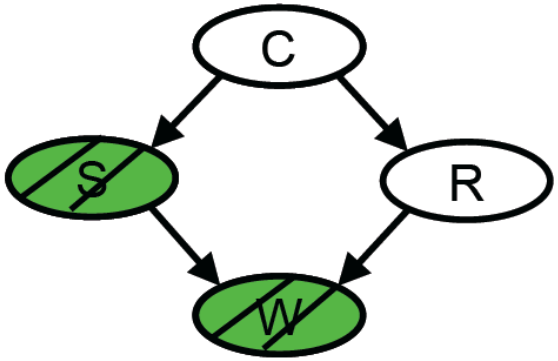
- Together, weighted sampling distribution is consistent

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

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: Consistency

$$\hat{\mathbf{P}}(x \mid \mathbf{e}) = \alpha \sum_{\mathbf{y}} N_{WS}(x,\mathbf{y},\mathbf{e})\, w(x,\mathbf{y},\mathbf{e}) \quad \text{// from Likelihood Weighting}$$

$$\approx \alpha' \sum_{\mathbf{y}} S_{WS}(x,\mathbf{y},\mathbf{e})\, w(x,\mathbf{y},\mathbf{e}) \quad \text{// for Large N}$$

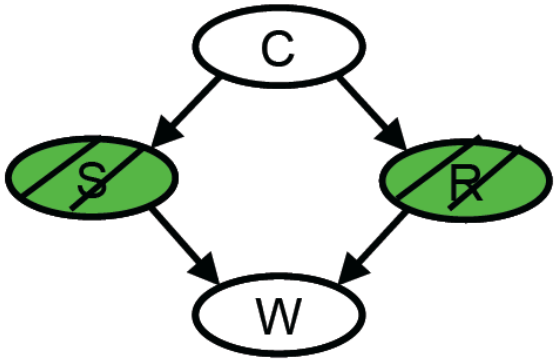
$$\approx \alpha' \sum_{\mathbf{y}} P(x,\mathbf{y},\mathbf{e}) \quad \text{// from Previous Slide}$$

$$\approx \alpha' P(x,\mathbf{e}) \quad \text{// sum Out } \mathbf{y}$$

$$\approx P(x \mid \mathbf{e}) \quad \text{// normalize}$$

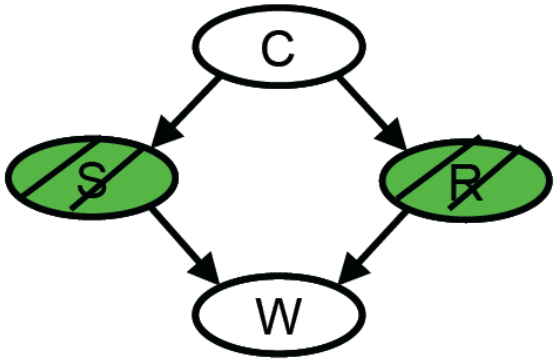
Likelihood Weighting

- 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

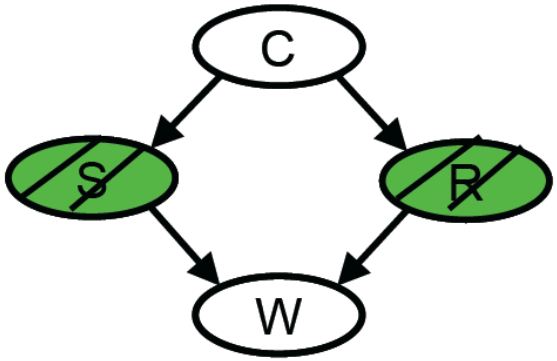
- Likelihood weighting is good
  - We have taken evidence into account as we generate the sample
  - E.g. here, W's value will get picked based on the evidence values of S, R
  - More of our samples will reflect the state of the world suggested by the evidence
- Likelihood weighting doesn't solve all 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)



Likelihood Weighting

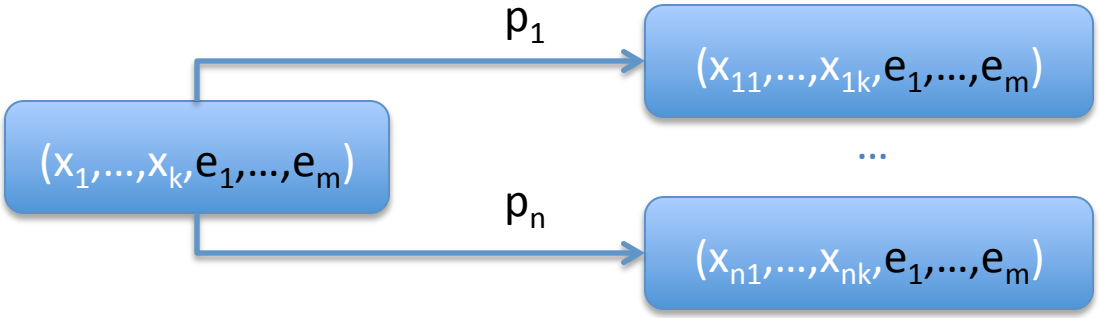
- Likelihood weighting is good
  - We have taken evidence into account as we generate the sample
  - E.g. here, W's value will get picked based on the evidence values of S, R
  - More of our samples will reflect the state of the world suggested by the evidence
- Likelihood weighting doesn't solve all 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



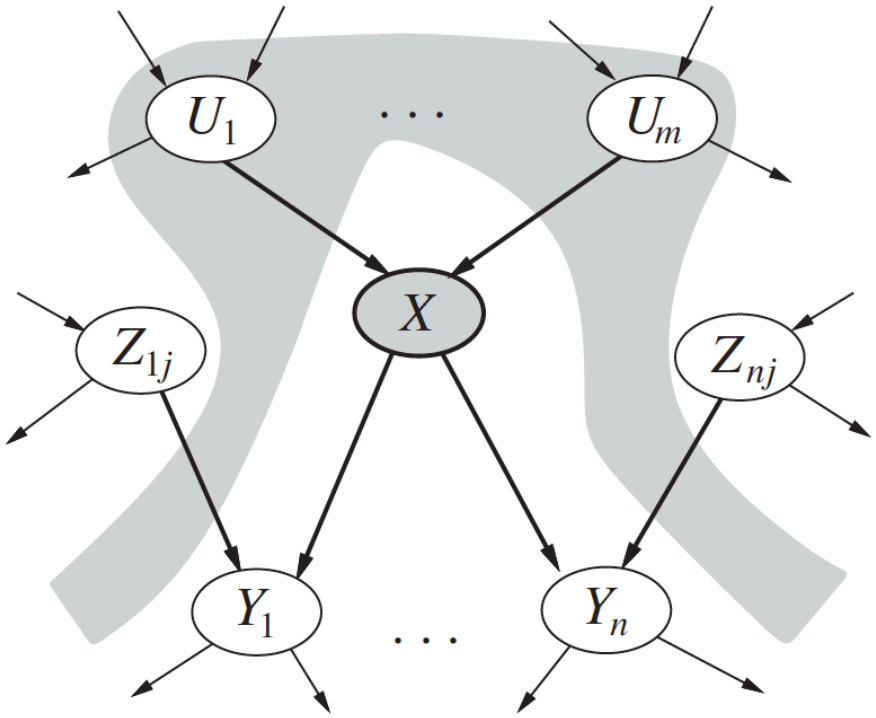
Gibbs Sampling and MCMC

- An instance of Markov chain Monte-Carlo (MCMC)
  - Generate sample as a random change to **previous one**
- An MCMC is always in a particular, current state:
  - Specifying the value of all variables (X,E)



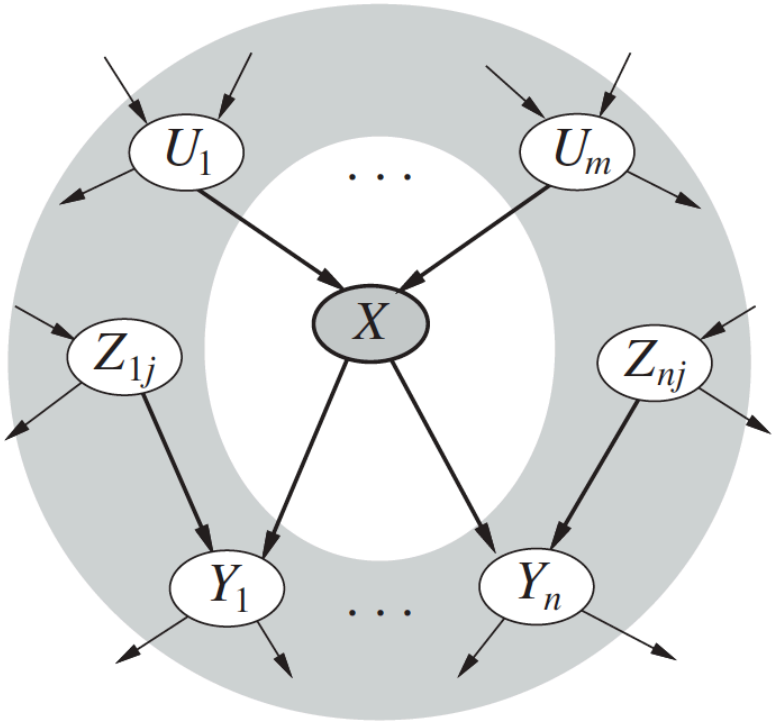
- An MCMC generates a next state:
  - By making random changes to the current state

Markov Blanket



Topological Semantics

X conditionally independent of its non-descendants given its parents



Markov Blanket (MB)

X conditionally independent of all other nodes given its MB

$$P(x'_i \mid mb(X_i)) = \alpha P(x'_i \mid Parents(X_i)) \times \prod_{Y_j \in Children(X_i)} P(y_j \mid Parents(Y_j))$$

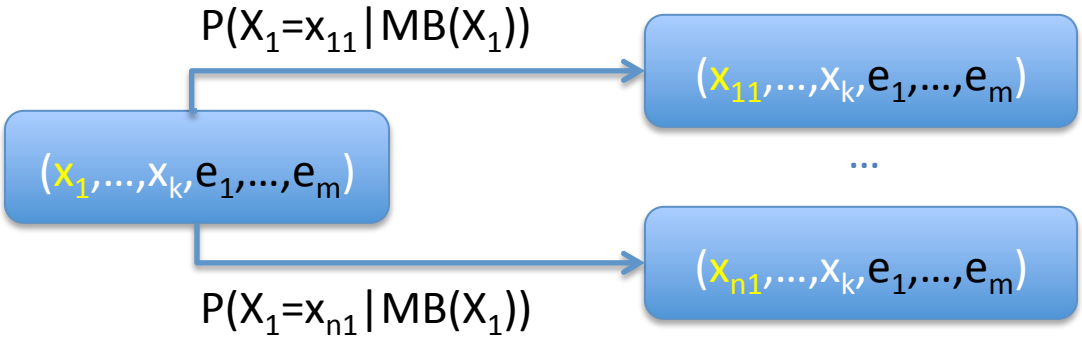
Gibbs Sampling

- Starts with an arbitrary initial state with:
  - Evidence variables fixed at their observed values

$(x_1,\ldots,x_k,e_1,\ldots,e_m)$

Gibbs Sampling

- Starts with an arbitrary initial state with:
  - Evidence variables fixed at their observed values



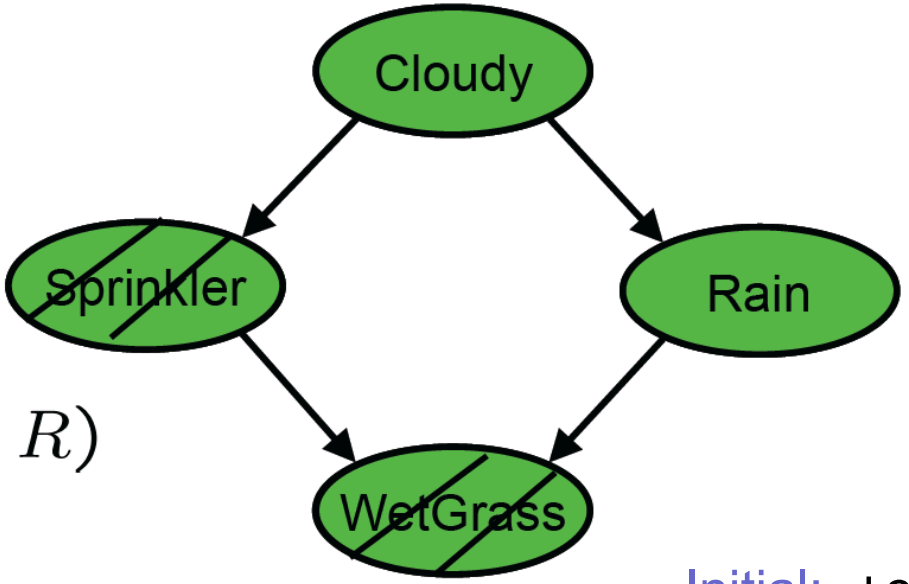
- Generates a next state by randomly:
  - Sampling a value for **one** of the **non-evidence** RVs X_i
  - Conditioned on **current values** of RVs in the **MB** of X_i

Gibbs Sampling: P(Rain | +s,+w)

$P(C)$

+c	0.5
-c	0.5

+c



$P(S|C)$

+s

+c	+s	0.1

$P(R|C)$

+c	+r	0.8
	-r	0.2

-r

$P(W|S,R)$

+w

+s			
	-r	+w	0.90

Initial: +c, +s, -r, +w

random values

Gibbs Sampling: P(Rain | +s,+w)

$P(C)$

+c	0.5
-c	0.5

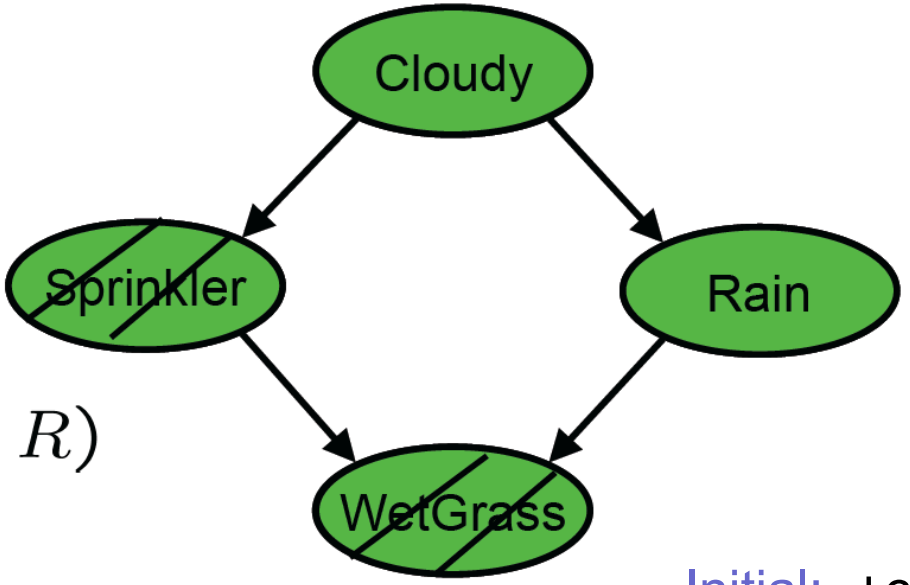
P(C | mb(C)) =

P(C | +s,+w) ↦ −c

$P(S|C)$

+c	+s	0.1

+s



$P(R|C)$

+c	+r	0.8
	-r	0.2

-r

$P(W|S,R)$

+w

+s			
	-r	+w	0.90

Initial: +c, +s, -r, +w

Next: -c, +s, -r, +w

Gibbs Sampling: P(Rain | +s,+w)

$P(C)$

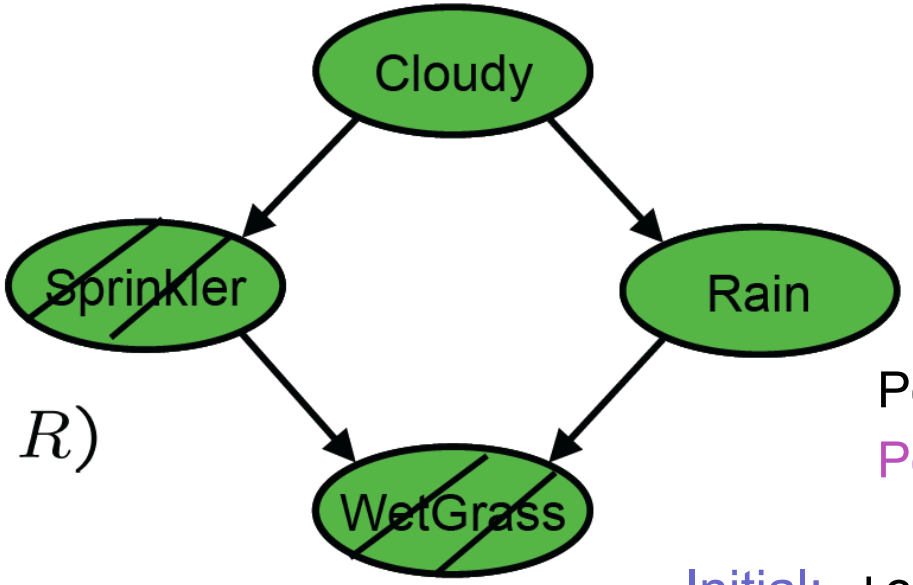
+c	0.5
-c	0.5

-c

$P(S|C)$

+s

+c	+s	0.1



$P(R|C)$

-c	+r	0.2
	-r	0.8

P(R | mb(R)) =

P(R | −c,+w,+s) ↦ +r

$P(W|S,R)$

+w

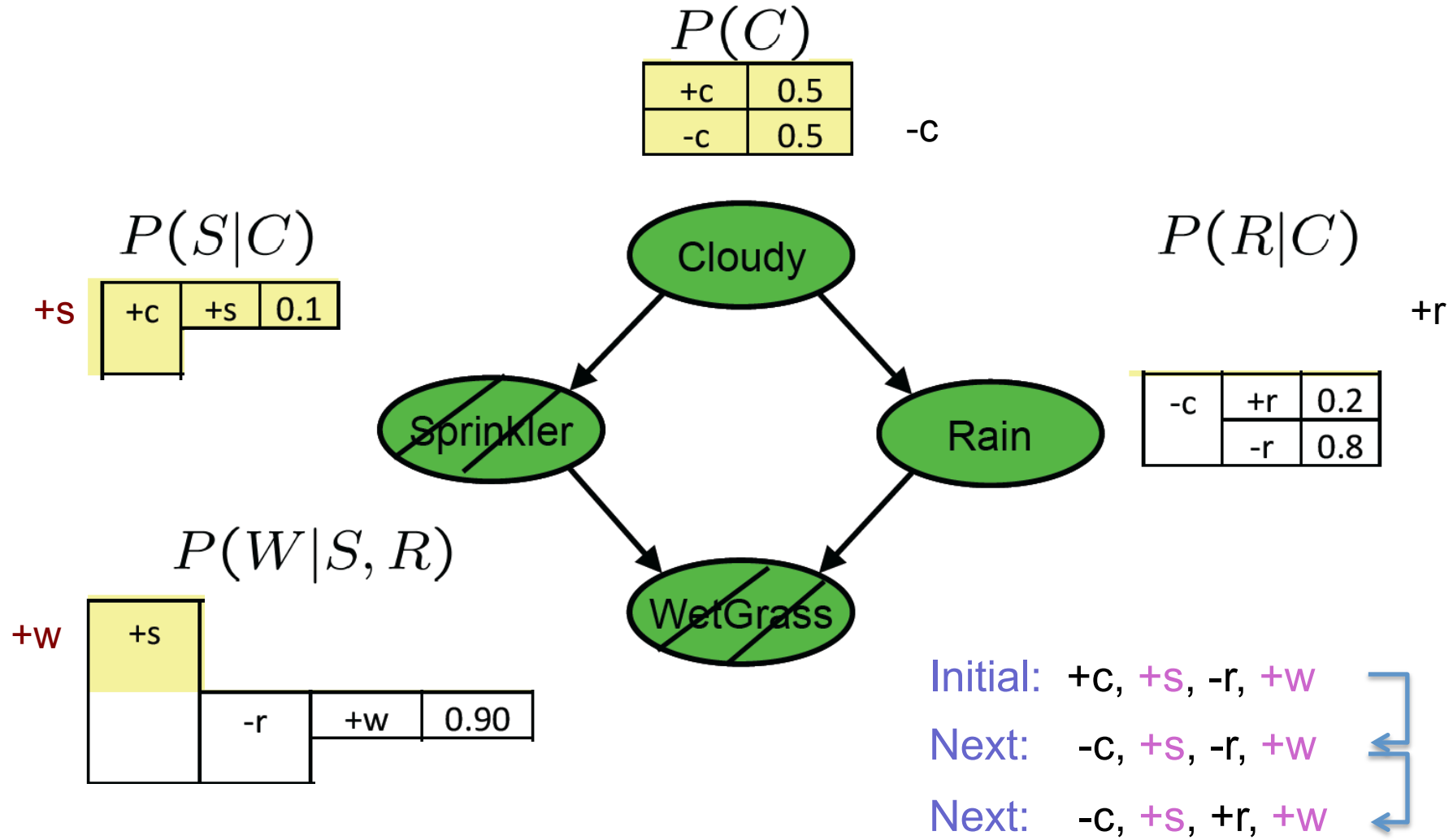
+s	-r	+w	0.90

Initial: +c, +s, -r, +w

Next: -c, +s, -r, +w

Next: -c, +s, +r, +w

Gibbs Sampling: P(Rain | +s,+w)

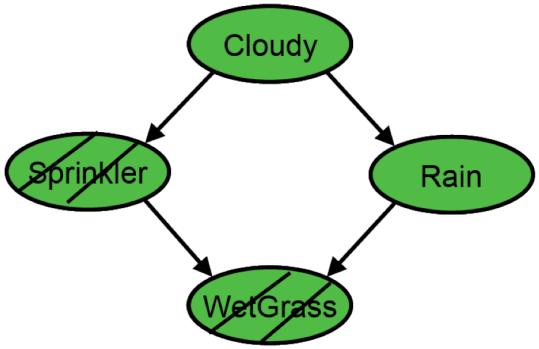


80 states : [+r : 20, - r : 60] ↦ [+r : 0.25, - r : 0.75]

Gibbs Sampling: Initial state

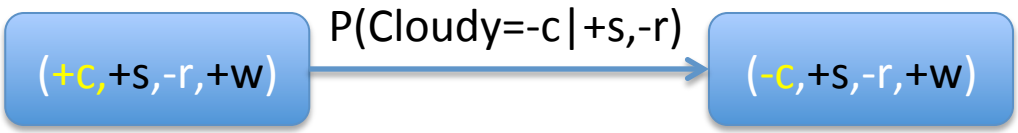
- Evidence random Variables
  - Fixed to observed values Sprinkler = +s, WetGrass = +w
- Non-evidence random Variables
  - Chosen randomly, eg. Cloudy = +c, R = -r

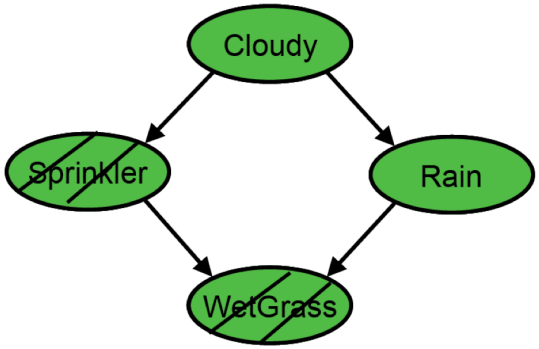
(+c,+s,-r,+w)



Gibbs Sampling: Next state

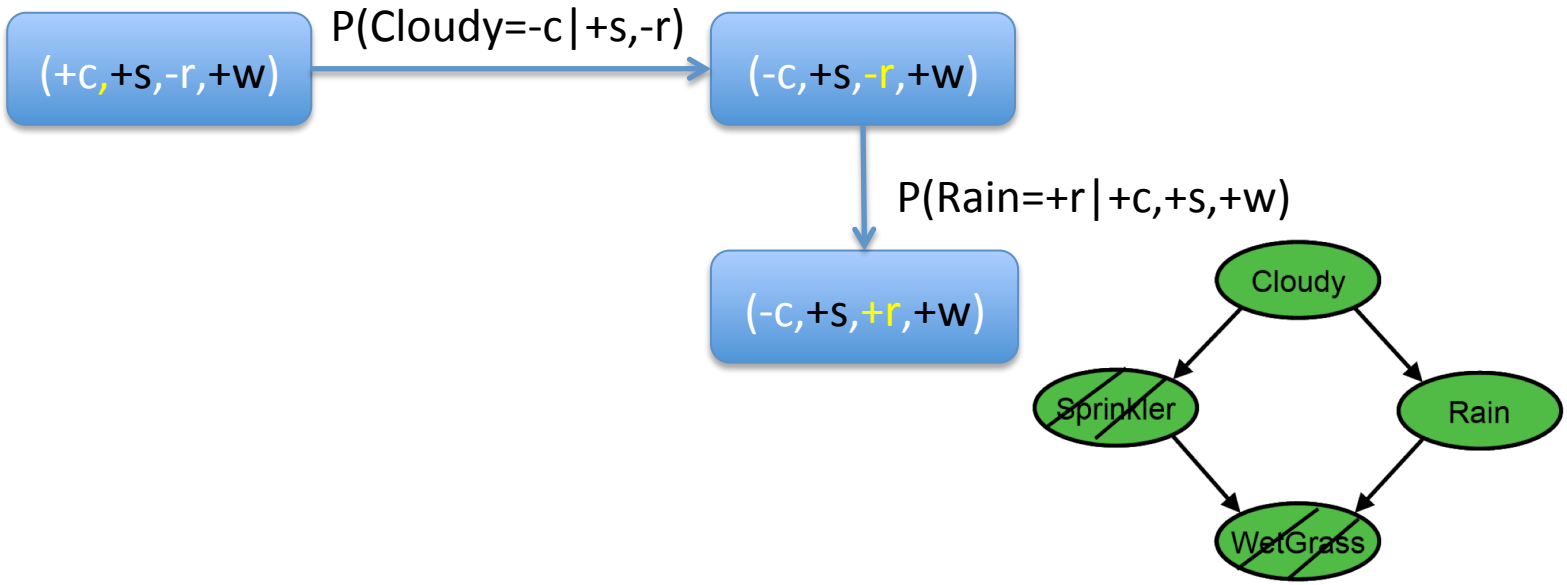
- Non-evidence RVs are sampled repeatedly
  - In arbitrary order For example pick first Cloudy





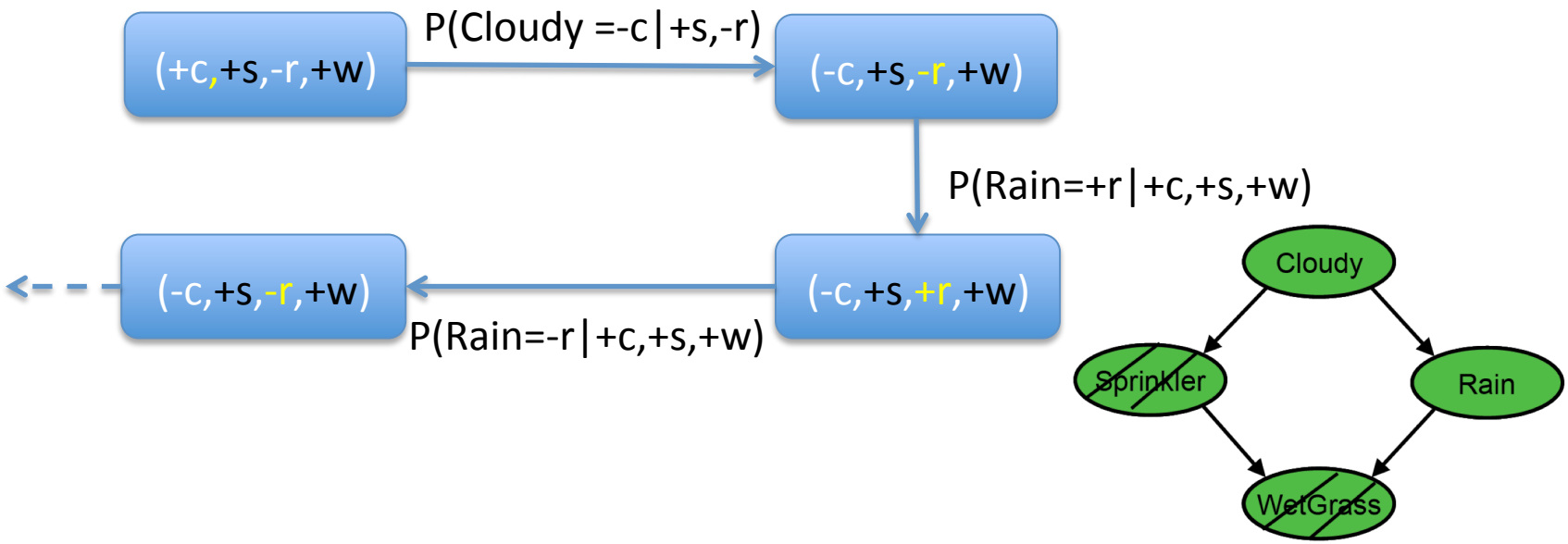
Gibbs Sampling: Next state

- Non-evidence RVs are sampled repeatedly
  - In arbitrary order For example pick next Rain



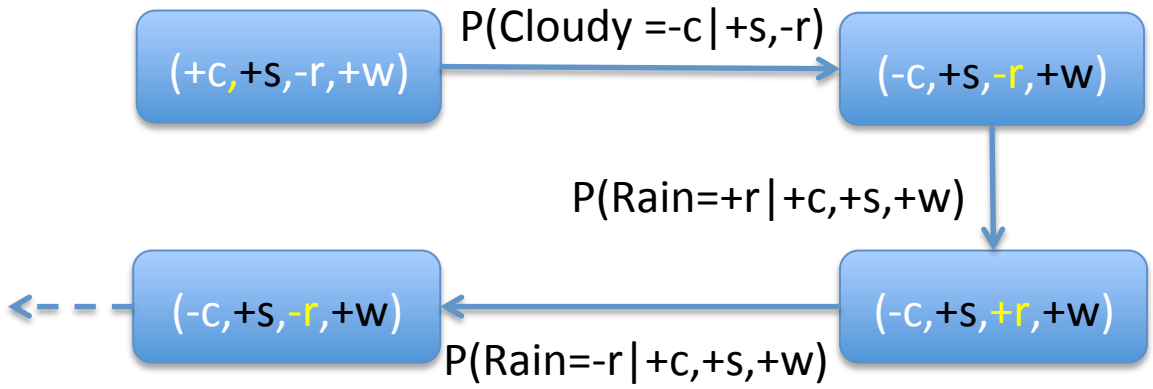
Gibbs Sampling: Next state

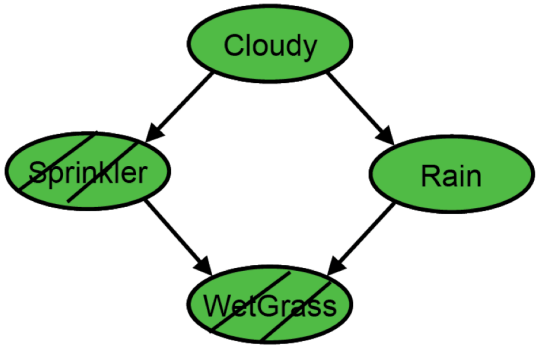
- Non-evidence RVs are sampled repeatedly
  - Suppose 20 times +r, and 60 times -r
  - Then P(Rain | +s,+w) = Normalize([20,60]) = [0.25,0.75]



Gibbs Sampling: Next state

- Only a finite number of instances of (C,S,R,W)
  - From each state sum of outgoing probabilities = 1
  - Markov chain M = (states S, initial-distribution I, matrix A)





- In the limit converges to the right (limit) distribution

Gibbs-Sampling: Why it Works?

Markov chain: $P(\mathbf{x}' \mid \mathbf{x})$ transition probability

States: $\pi_t(\mathbf{x})$ probability of being in state $\mathbf{x}$ at time t

$$\pi_{t+1}(\mathbf{x}') = \sum_{\mathbf{x}} P(\mathbf{x}' \mid \mathbf{x})\, \pi_t(\mathbf{x}) \qquad \forall \mathbf{x}'$$

Stationary distribution: $\pi_{t+1}(\mathbf{x}) = \pi_t(\mathbf{x}) = \pi(\mathbf{x}) \qquad \forall \mathbf{x}$

$$\pi(\mathbf{x}') = \sum_{\mathbf{x}} P(\mathbf{x}' \mid \mathbf{x})\, \pi(\mathbf{x}) \qquad \forall \mathbf{x}'$$

Markov chain is called ergodic if:

- There is a number N such that
- Every state is reachable from every other in exactly N steps

$\mapsto$ There is a unique stationary distribution π

Gibbs-Sampling: Why it Works?

Stationary distribution: $\pi_{t+1}(\mathbf{x}) = \pi_t(\mathbf{x}) = \pi(\mathbf{x}) \quad \forall \mathbf{x}$

$$\pi(\mathbf{x}') = \sum_{\mathbf{x}} P(\mathbf{x}' \mid \mathbf{x})\, \pi(\mathbf{x}) \qquad \forall \mathbf{x}'$$

Expected outflow from x = expected inflow to x

Sufficient condition: Above is satisfied for any pair $\mathbf{x}, \mathbf{x}'$

$$P(\mathbf{x}' \mid \mathbf{x})\, \pi(\mathbf{x}) = P(\mathbf{x} \mid \mathbf{x}')\, \pi(\mathbf{x}') \quad \forall \mathbf{x}, \mathbf{x}' \quad \text{(detailed balance)}$$

Detailed balance satisfies stationarity:

$$\sum_{\mathbf{x}} P(\mathbf{x}' \mid \mathbf{x})\, \pi(\mathbf{x}) = \sum_{\mathbf{x}} P(\mathbf{x} \mid \mathbf{x}')\, \pi(\mathbf{x}') =$$

$$\pi(\mathbf{x}') \sum_{\mathbf{x}} P(\mathbf{x} \mid \mathbf{x}') = \pi(\mathbf{x}') \qquad \forall \mathbf{x}'$$

Gibbs-Sampling: Why it Works?

Sampling one variable conditioned on all other: Same as MB

$$P(x_i', \bar{\mathbf{x}}_i, \mathbf{e} \mid x_i, \bar{\mathbf{x}}_i, \mathbf{e}) = P(x_i' \mid \bar{\mathbf{x}}_i, \mathbf{e}) = P(X_i \mid MB(X_i)) \qquad \forall \mathbf{x}'$$

Each transition is in detailed balance with $\pi(\mathbf{x}) = P(\mathbf{x} \mid \mathbf{e})$:

$P(x_i' \mid \bar{\mathbf{x}}_i, \mathbf{e})\, P(\mathbf{x} \mid \mathbf{e}) =$ (expand $\mathbf{x}$ in t2)

$P(x_i' \mid \bar{\mathbf{x}}_i, \mathbf{e})\, P(x_i, \bar{\mathbf{x}}_i \mid \mathbf{e})$ (chain rule in t2)

$P(x_i' \mid \bar{\mathbf{x}}_i, \mathbf{e})\, P(x_i \mid \bar{\mathbf{x}}_i, \mathbf{e})\, P(\bar{\mathbf{x}}_i \mid \mathbf{e})$ (chain rule bwd in t1,t3)

$P(x_i \mid \bar{\mathbf{x}}_i, \mathbf{e})\, P(x_i', \bar{\mathbf{x}}_i \mid \mathbf{e})$ (compact $\mathbf{x}_i'$, $\bar{\mathbf{x}}_i' = \bar{\mathbf{x}}_i$)

$P(x_i \mid \bar{\mathbf{x}}_i', \mathbf{e})\, P(\mathbf{x}' \mid \mathbf{e})$ (detailed balance)

Unless MC contains 0,1 (can get disconnected) MC is ergodic

Gibbs-Sampling Algorithm

function Gibbs-Ask (X,**e**,bn,N) returns $\hat{P}(X|\mathbf{e})$

local

$\mathbf{Z} \subseteq$ bn // the nonevidence variables in bn

$\mathbf{x} =$ Initialize($\mathbf{Z}$,$\mathbf{e}$) // random in **Z**, fixed in **e**

$\mathbf{W} = \mathbf{0}$ // vector of counts for each value of X

foreach j = 1:N { // take N×|**Z**| samples

foreach $Z_i \in \mathbf{Z}$ { // one sample for each Z_i

$\mathbf{x}[Z_i] =$ Sample($P(Z_i \mid \text{mb}(Z_i,\text{bn}))$) // get sample

$\mathbf{W}[x] += 1$ }} // where x is value of X in **x**

return Normalize(**W**)

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.

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

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.

Gibbs Sampling

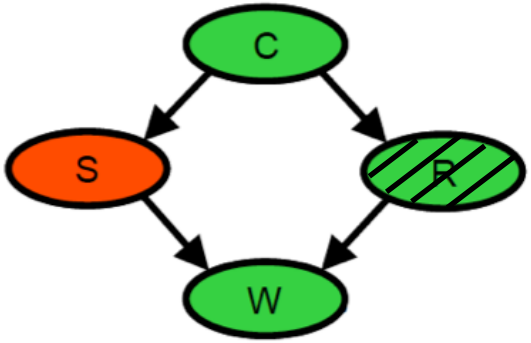
- *Procedure:* keep track of a full instantiation $x_1, x_2, \ldots, x_n$. Start with an arbitrary instantation consistent with the evidence. Sample one variable at a time, conditioned on all the rest, but keep evidence fixed. Keep repeating this for a long time.

- *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.

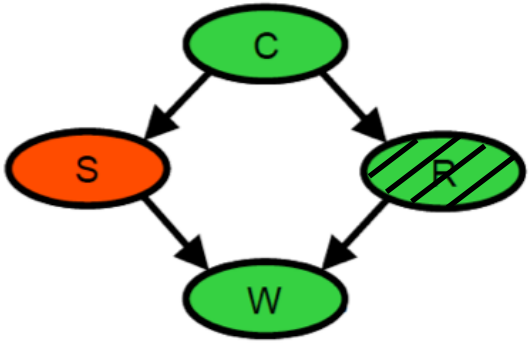
In contrast: likelihood weighting only conditions on upstream evidence, and hence weights obtained in likelihood weighting can sometimes be very small. Sum of weights over all samples is indicative of how many "effective" samples were obtained, so want high weight.

Gibbs Sampling



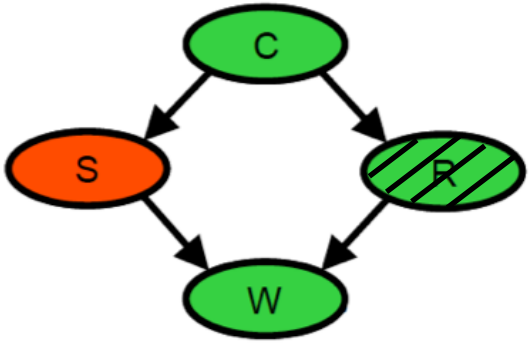
- Say we want to sample P(S | R = +r)

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)

Gibbs Sampling

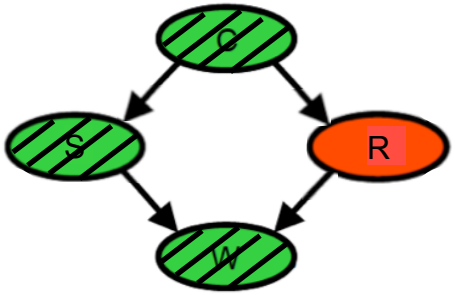


- Say we want to sample P(S | R = +r)
- Step 1: Initialize
  - Set evidence (R = +r)
  - Set all other variables (S, C, W) to random values (e.g. by prior sampling or just uniformly sampling; say S = s, W = +w, C = -c)
- Steps 2+: Repeat the following for some number of iterations
  - Choose a non-evidence 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)

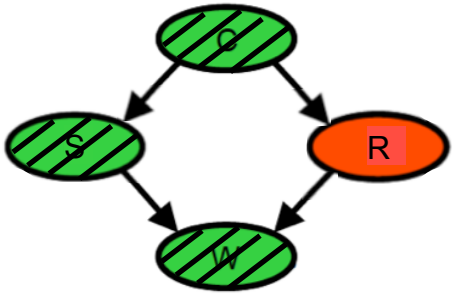
$$P(R|+s,-c,-w)$$



Gibbs Sampling Example

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

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

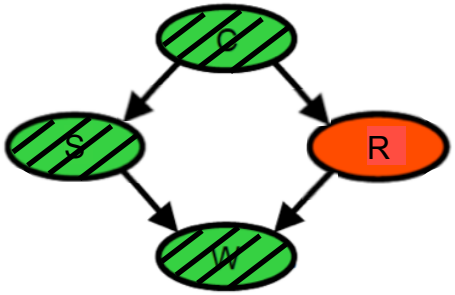


Gibbs Sampling Example

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

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

$$= \frac{P(R,+s,-c,-w)}{\sum_r P(R=r,+s,-c,-w)}$$



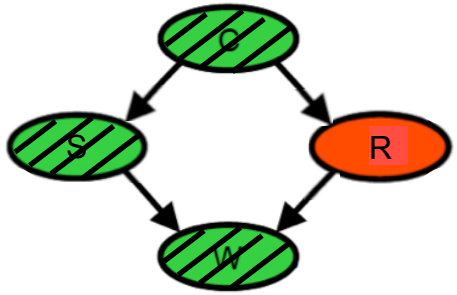
Gibbs Sampling Example

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

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

$$= \frac{P(R,+s,-c,-w)}{\sum_r P(R=r,+s,-c,-w)}$$

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



Gibbs Sampling Example

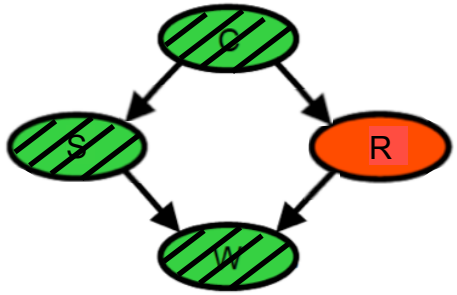
- Want to sample from P(R | +s,-c,-w)
  - Shorthand for P(R | S=+s,C=-c,W=-w)

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

$$= \frac{P(R,+s,-c,-w)}{\sum_r P(R=r,+s,-c,-w)}$$

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

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



Gibbs Sampling Example

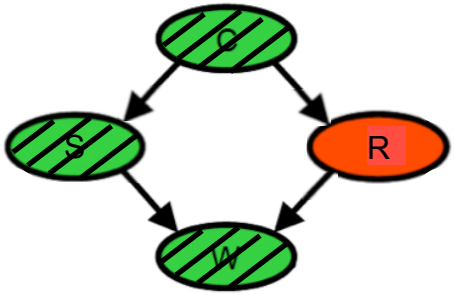
- Want to sample from P(R | +s,-c,-w)
  - Shorthand for P(R | S=+s,C=-c,W=-w)

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

$$= \frac{P(R,+s,-c,-w)}{\sum_r P(R=r,+s,-c,-w)}$$

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

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



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

Further Reading*

- Gibbs sampling is a special case of more general methods called Markov chain Monte Carlo (MCMC) methods

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)

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