UBC Department of Computer Science
Undergraduate Events
More details @ https://my.cs.ubc.ca/students/development/events

Resume Editing Drop-in Session

Mon., Sept 28
10 am – 2 pm (sign up at 9 am)
ICCS 253

Facebook Crush Your Code Workshop

Mon., Sept 28
6 – 8 pm
DMP 310

UBC Careers Day & Professional School Fair

Wed., Sept 30 & Thurs., Oct 1
10 am – 4 pm
AMS Nest

ACM Programming Competition Try Out

Sat., Oct 3
8:45 am – 2 pm
ICCS 005

IBM Info/Networking Session

Mon., Oct 5
6 – 8 pm
DMP 310

Invoke Media Info Session

Thurs., Oct 8
5:30 – 6:30 pm
DMP 110

Intelligent Systems (AI-2)

Computer Science cpsc422, Lecture 12

Oct, 5, 2015

Slide credit: some slides adapted from Stuart Russell (Berkeley)

Lecture Overview

- Recap of Forward and Rejection Sampling
- Likelihood Weighting
- Monte Carlo Markov Chain (MCMC) – Gibbs Sampling
- Application Requiring Approx. reasoning

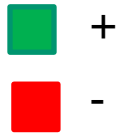
Sampling

The building block on any sampling algorithm **is the generation of samples from a known (or easy to compute, like in Gibbs) distribution**

We then **use these samples to derive estimates of probabilities hard-to-compute exactly**

And you want **consistent sampling methods…. More samples…. Closer to….**

Prior Sampling

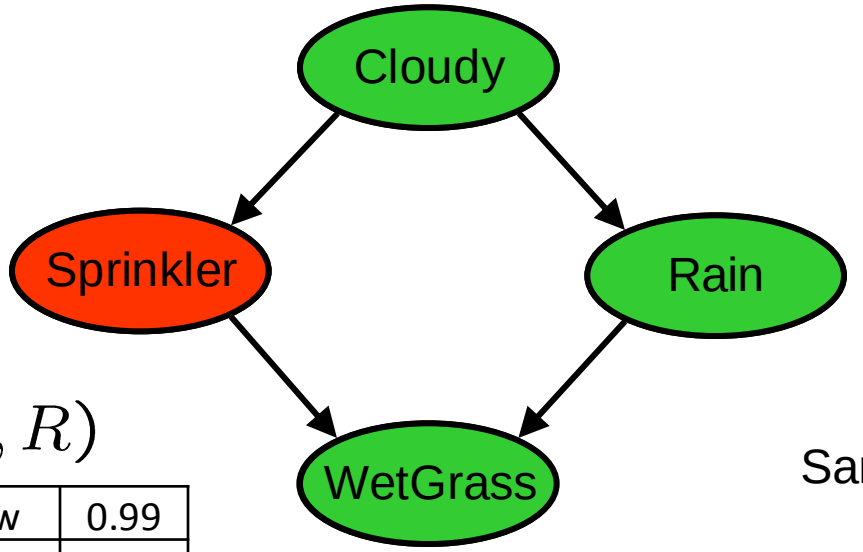


$P(C)$

+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:

+c, -s, +r, +w

-c, +s, -r, +w

…

Example

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

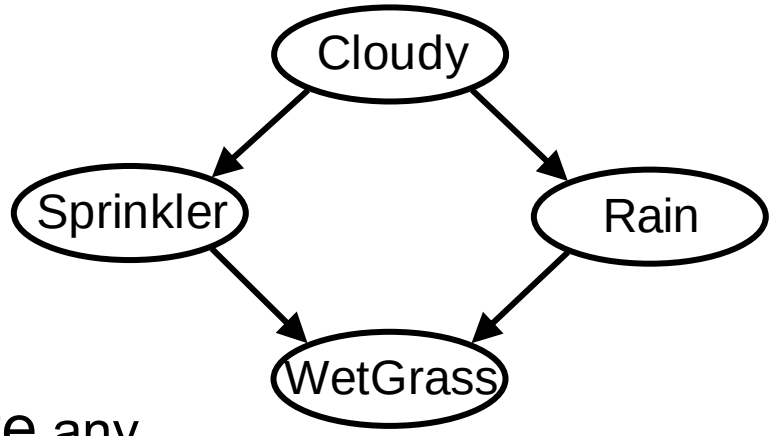
+c, -s, +r, +w

+c, +s, +r, +w

-c, +s, +r, -w

+c, -s, +r, +w

-c, -s, -r, +w



From these samples you can compute any distribution involving the five vars….

Example

Can estimate anything else from the samples, besides P(W), P(R) , etc:

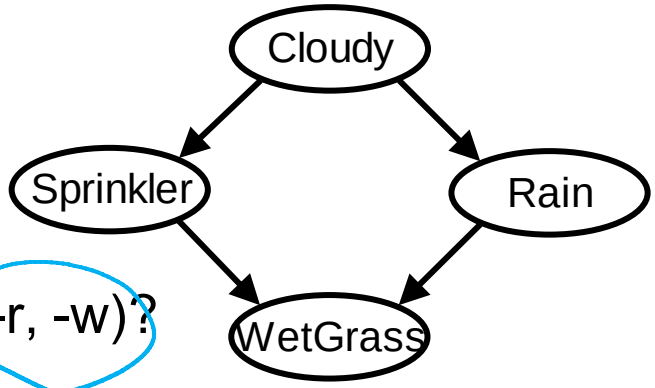
+c, -s, +r, +w

+c, +s, +r, +w

-c, +s, +r, -w

+c, -s, +r, +w

-c, -s, -r, +w



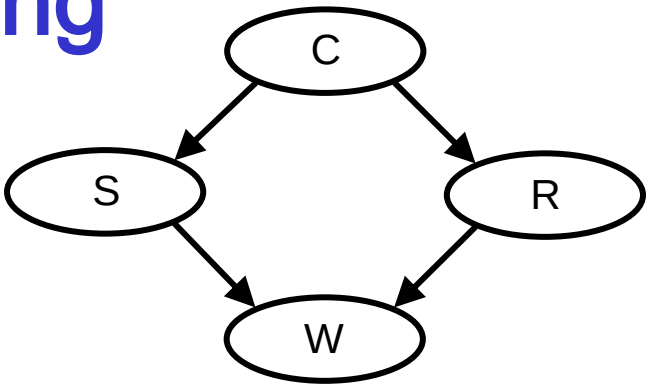
- What about P(C| +w)? P(C| +r, +w)? P(C| -r, -w)?

A. [+c 0, -c 1] B. [+c .5, -c .5] C. [+c 1, -c 0]

D. None of the above

Can use/generate fewer samples when we want to estimate a probability conditioned on evidence?

Rejection Sampling



Let's say we want P(W| +s)

- 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

But what happens if +s is rare?

And if the number of evidence vars grows……

i-clicker

A. Less samples will be rejected

B. More samples will be rejected

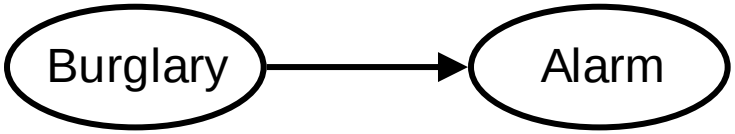
C. The same number of samples will be rejected

Likelihood Weighting

Problem with rejection sampling:

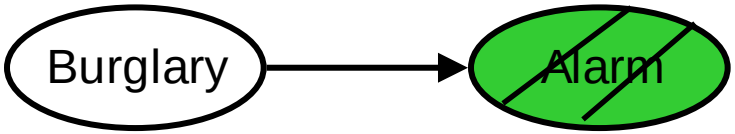
- If evidence is unlikely, you reject a lot of samples
- You don't exploit your evidence as you sample
- Consider P(B|+a)

-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



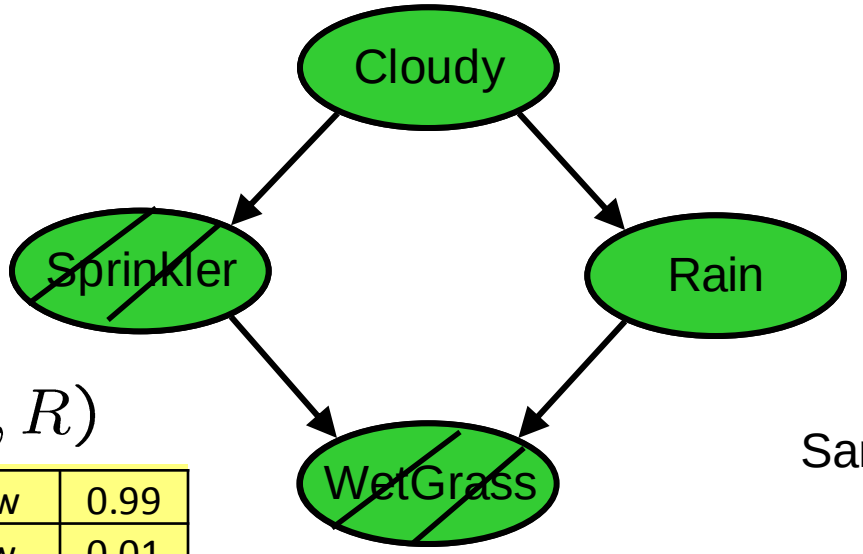
Problem?:

Solution: weight by probability of evidence given parents

Likelihood Weighting

$P(C)$

+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:

+c +s +r +w

…

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

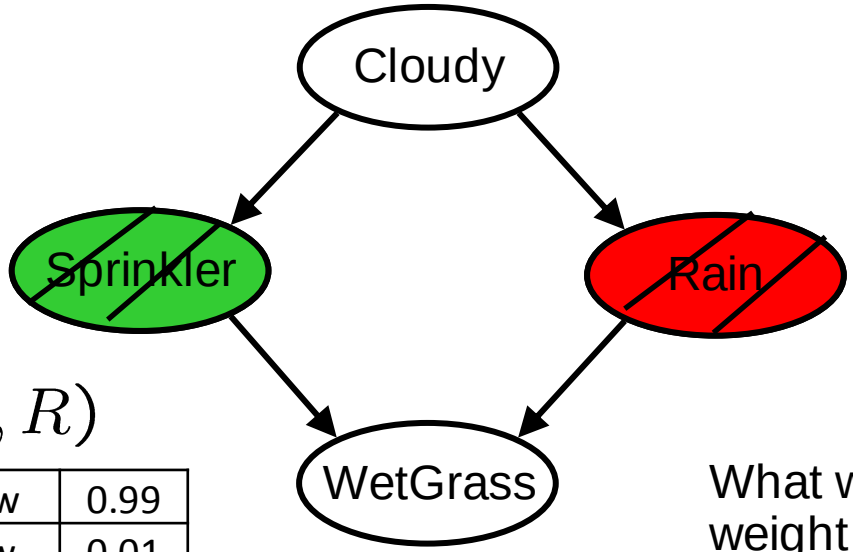
Likelihood Weighting

$P(C)$

+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

What would be the weight for this sample?

+c, +s, -r, +w

i-clicker

A 0.08

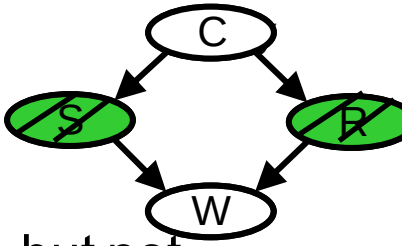
B 0.02

C. 0.005

Likelihood Weighting

Likelihood weighting is good

- We have taken evidence into account as we generate the sample
- All our samples will reflect the state of the world suggested by the evidence
- Uses all samples that it generates (much more efficient than rejection sampling)



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*)
- Degradation in performance with large number of evidence vars -> each sample small weight

We would like to consider evidence when we sample *every* variable

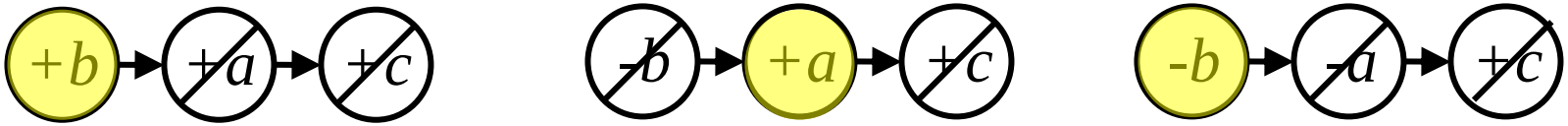
Lecture Overview

- Recap of Forward and Rejection Sampling
- Likelihood Weighting
- Monte Carlo Markov Chain (MCMC) – Gibbs Sampling
- Application Requiring Approx. reasoning

Markov Chain Monte Carlo

Idea: instead of sampling from scratch, create samples that are each like the last one (only randomly change one var).

Procedure: resample one variable at a time, conditioned on all the rest, but keep evidence fixed. E.g., for P(B|+c):



+b, +a, +c

Sample b

- b, +a, +c

Sample a

- b, -a, +c

Sample b

- b, -a, +c

Sample a

- b, -a, +c

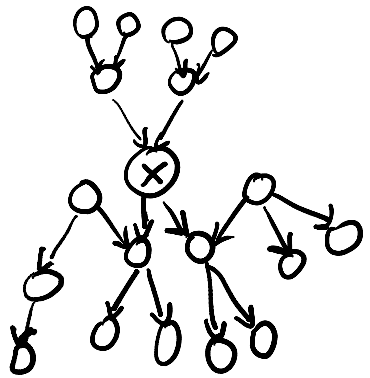
Sample b

+ b, -a, +c

Markov Chain Monte Carlo

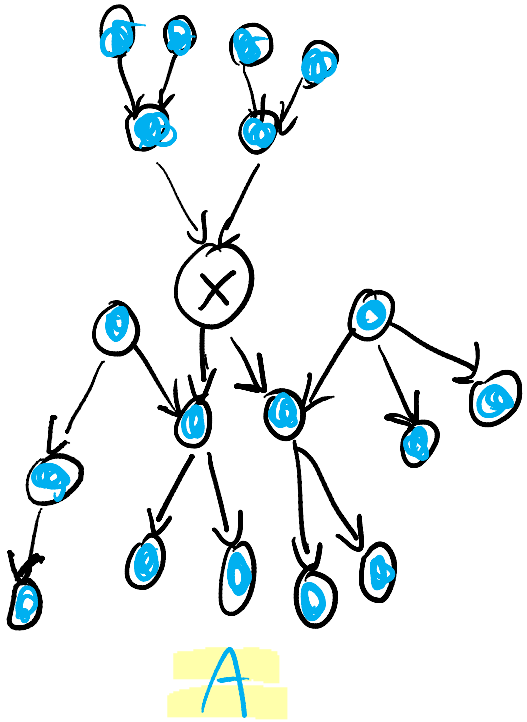
Properties: Now samples are not independent (in fact they're nearly identical), but sample averages are still consistent estimators! And can be computed efficiently

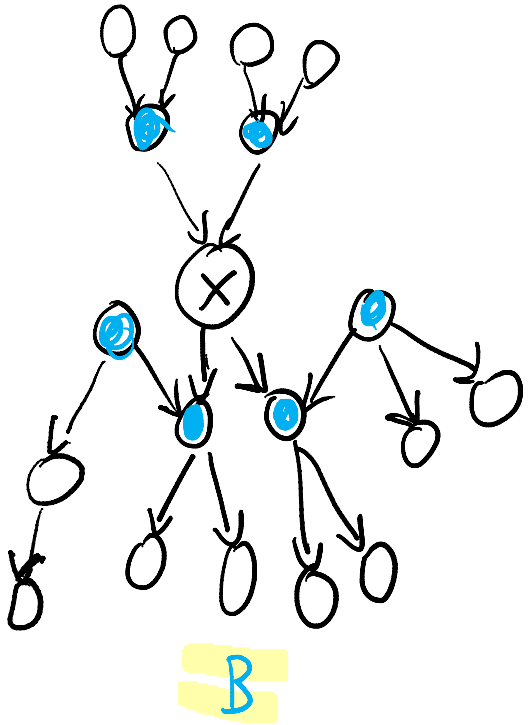
What's the point: when you sample a variable conditioned on all the rest, both upstream and downstream variables condition on evidence.

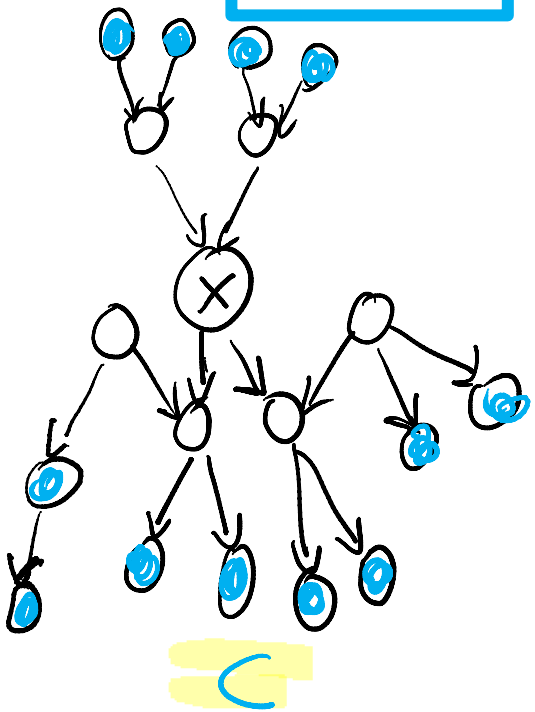


Open issue: what does it mean to sample a variable conditioned on all the rest ?

Sample for X is conditioned on all the rest





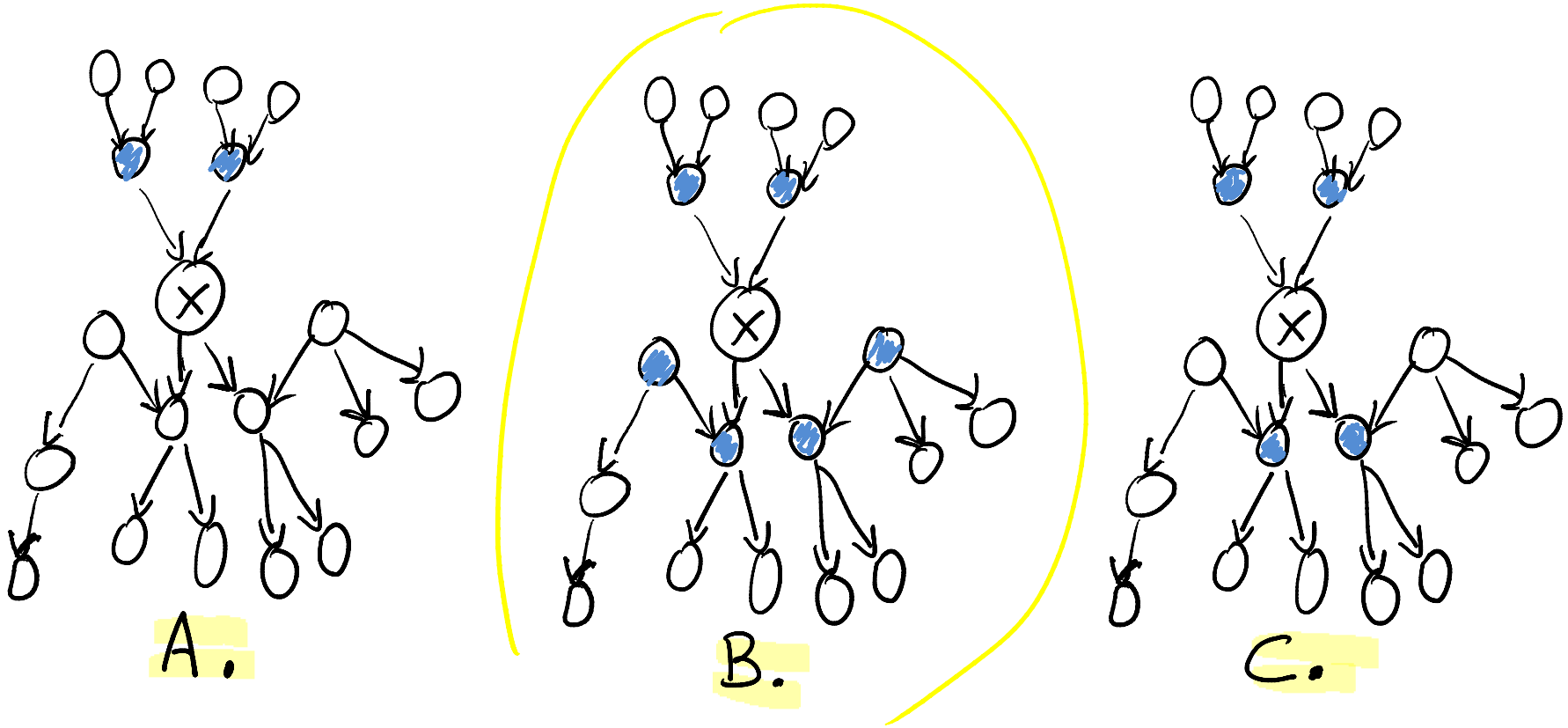


A. I need to consider all the other nodes

B. I only need to consider its Markov Blanket

C. I only need to consider all the nodes not in the Markov Blanket

Sample conditioned on all the rest



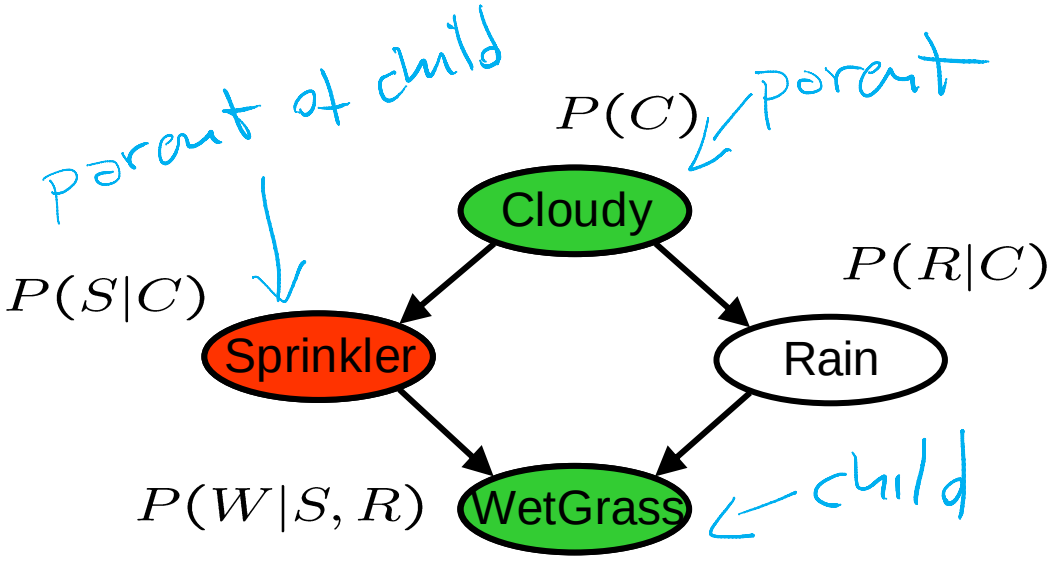
A node is conditionally independent from all the other nodes in the network, given its parents, children, and children's parents (i.e., its **Markov Blanket**) Configuration B

Probability given the Markov blanket is calculated as follows:

$$P(x_i'|mb(X_i)) = \alpha P(x_i'|parents(X_i)) \prod_{Z_j \in Children(X_i)} P(z_j|parents(Z_j))$$

We want to sample Rain

Rain's Markov Blanket is ?



$$P(r|c^+, s^-, w^+) = \alpha P(r|c^+) P(w^+|r, s^-)$$

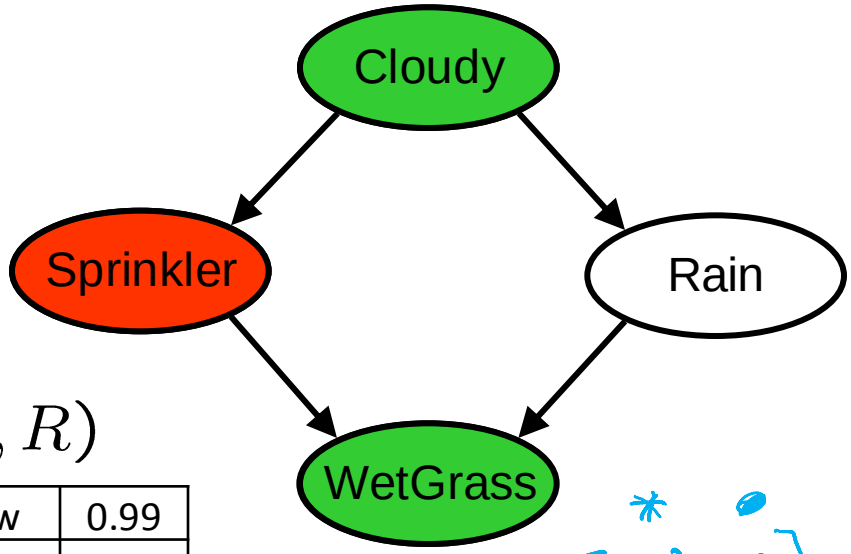
Markov blanket of *Cloudy* is
Sprinkler and *Rain*
Markov blanket of *Rain* is
Cloudy, *Sprinkler*, and *WetGrass*

$P(r|c^+, s^-, w^+) = \alpha P(r|c^+) P(w^+|r, s^-)$

We want to sample Rain

$P(C)$

+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

$= \alpha [.8, .2] \cdot [.9, .01]$

$= \alpha [.72, .002] = [.997, .003]$

sample this

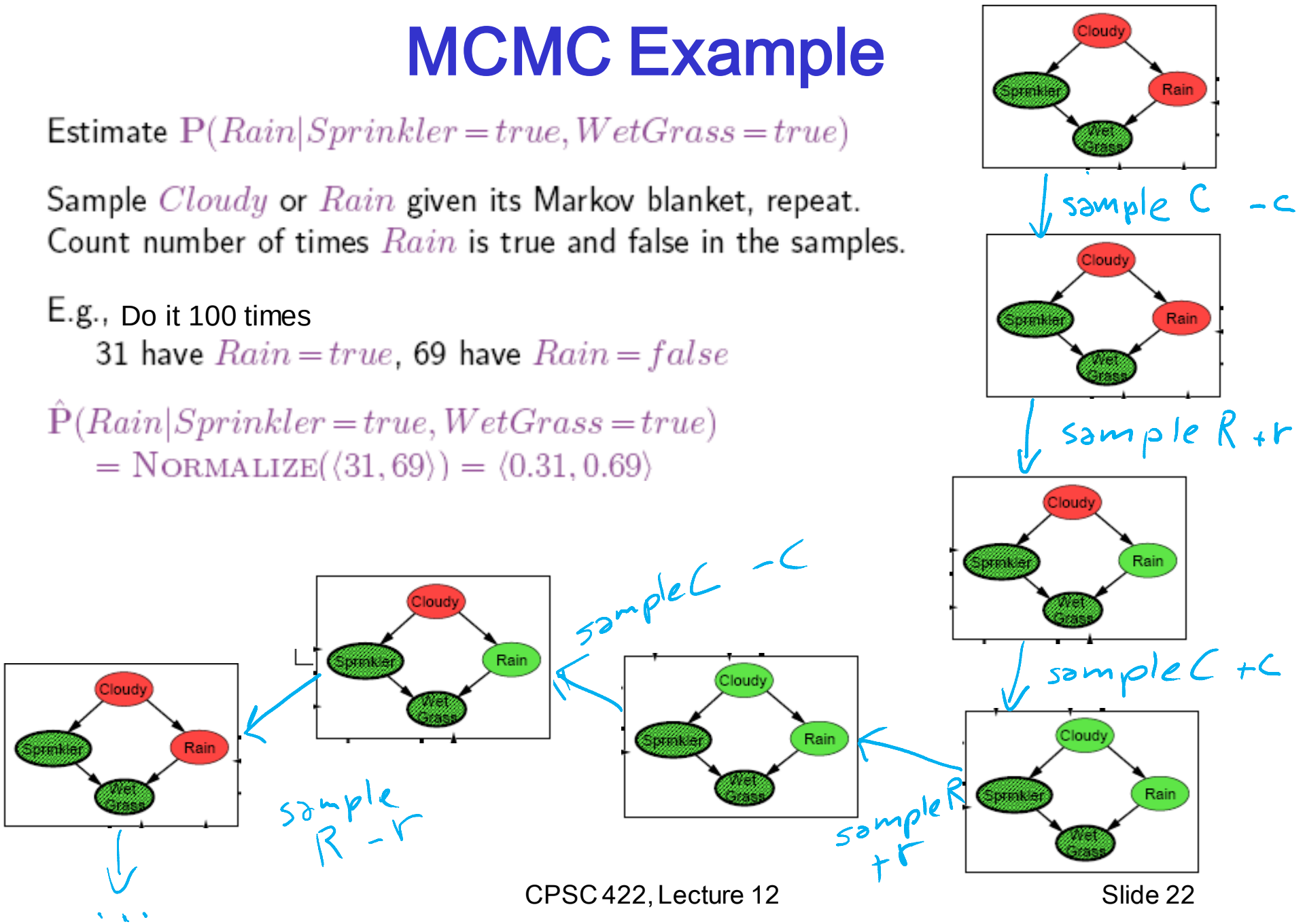
MCMC Example

Estimate $\mathbf{P}(Rain|Sprinkler = true, WetGrass = true)$

Sample $Cloudy$ or $Rain$ given its Markov blanket, repeat.
Count number of times $Rain$ is true and false in the samples.

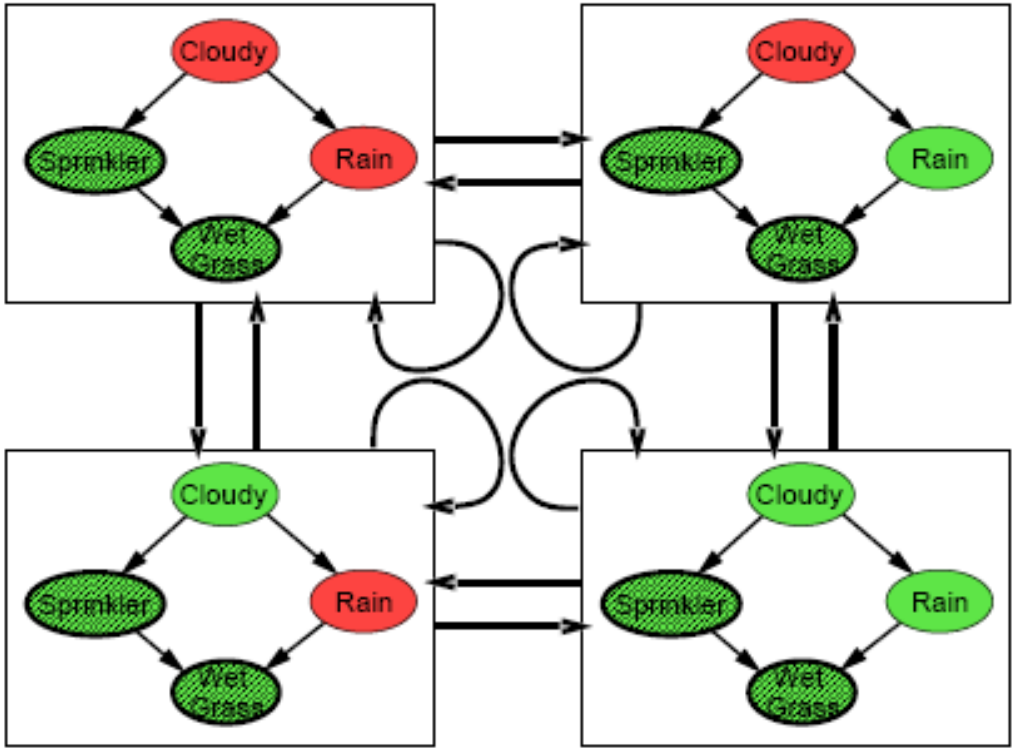
E.g., Do it 100 times
31 have $Rain = true$, 69 have $Rain = false$

$\hat{\mathbf{P}}(Rain|Sprinkler = true, WetGrass = true)$
$= \text{NORMALIZE}(\langle 31, 69 \rangle) = \langle 0.31, 0.69 \rangle$



Why it is called Markov Chain MC

With $Sprinkler = true, WetGrass = true$, there are four states:



States of the chain are possible samples (fully instantiated Bnet)

Wander about for a while, average what you see

Theorem: chain approaches stationary distribution:
long-run fraction of time spent in each state is exactly
proportional to its posterior probability ..given the evidence

Learning Goals for today's class

➢ **You can:**

- Describe and justify the Likelihood Weighting sampling method
- Describe and justify Markov Chain Monte Carlo sampling method

TODO for Wed

- **Next research paper:** Using Bayesian Networks to Manage Uncertainty in Student Modeling. *Journal of User Modeling and User-Adapted Interaction* **2002** _ Dynamic BN *(required only up to page 400)*

- **Follow instructions on course WebPage <Readings>**

- Keep working on assignment-2 (due on Fri, Oct 16)

Not Required

a. There are several ways to prove this. Probably the simplest is to work directly from the global semantics. First, we rewrite the required probability in terms of the full joint:

$$P(x_i|x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n) = \frac{P(x_1,\ldots,x_n)}{P(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)}$$

$$= \frac{P(x_1,\ldots,x_n)}{\sum_{x_i} P(x_1,\ldots,x_n)}$$

$$= \frac{\prod_{j=1}^{n} P(x_j|parentsX_j)}{\sum_{x_i} \prod_{j=1}^{n} P(x_j|parentsX_j)}$$

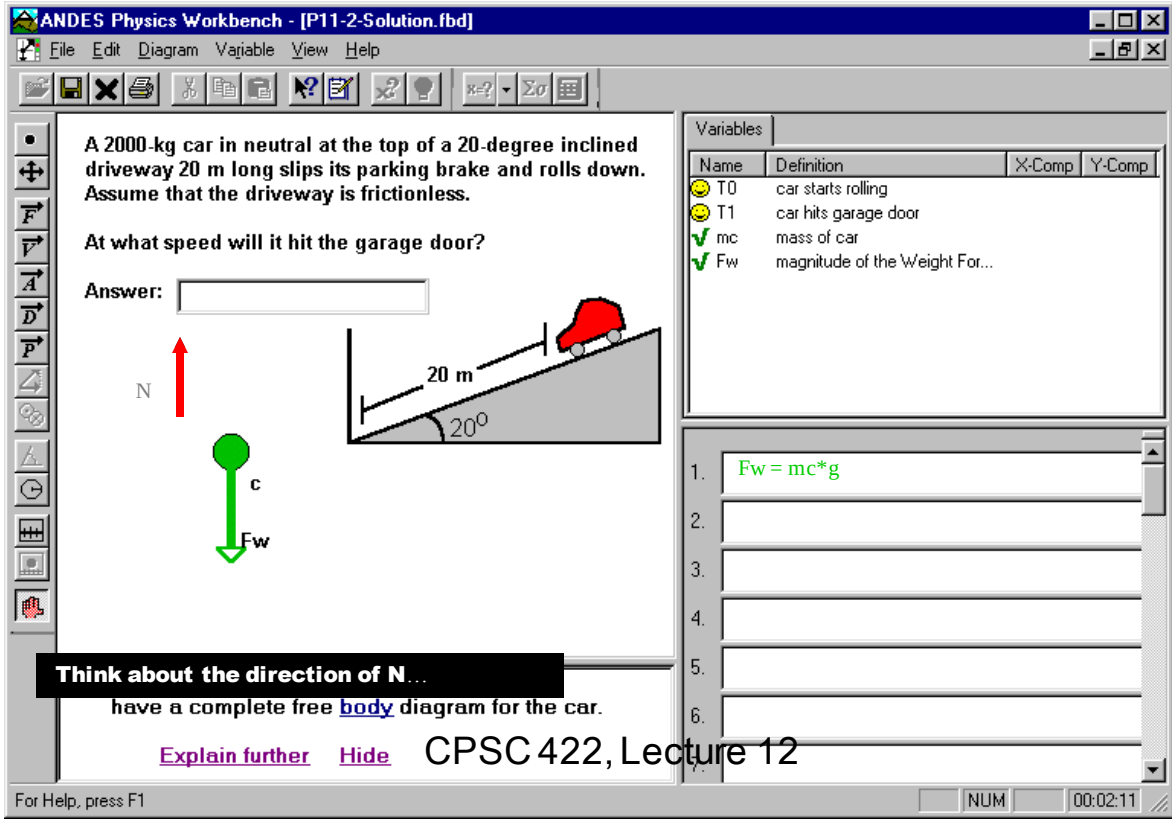
Now, all terms in the product in the denominator that do not contain x_i can be moved outside the summation, and then cancel with the corresponding terms in the numerator. This just leaves us with the terms that do mention x_i, i.e., those in which X_i is a child or a parent. Hence, $P(x_i|x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)$ is equal to

$$\frac{P(x_i|parentsX_i)\prod_{Y_j\in Children(X_i)} P(y_j|parents(Y_j))}{\sum_{x_i} P(x_i|parentsX_i)\prod_{Y_j\in Children(X_i)} P(y_j|parents(Y_j))}$$

Now, by reversing the argument in part (b), we obtain the desired result.

ANDES: an ITS for Coached problem solving

- The tutor monitors the student's solution and intervenes when the student needs help.
  - Gives feedback on correctness of student solution entries
  - Provides hints when student is stuck



Student Model for Coached Problem Solving

Three main functions

- Assess from the student's actions her domain knowledge, to decide which concepts the student needs help on
- Infer from student's actions the solution being followed, to understand what the student is trying to do
- Predict what further actions should be suggested to the student, to provide meaningful suggestions

Several sources of uncertainty

Same action can belong to different solutions

Often much of the reasoning behind the student's actions is hidden from the tutor

Correct answers can be achieved through guessing

Errors can be due to slips

System's help affects learning

In many domains, there is flexible solution step order

Andes deals with this uncertainty by using

Case Study: LW on Andes

Conati C., Gertner A., VanLehn K., Druzdzel M. (1997). On-Line Student Modeling for Coached Problem Solving Using Bayesian Networks . In Jameson A., Paris C., Tasso C., (eds.) *User Modeling; Proceedings of the sixth International Conference UM97*.

- Andes' networks include anywhere between 100 and 1000 nodes
  - (You'll know more about it after reading the paper for next class)
- Update needs to happen in real time
  - Starts each time a student performs a new action
  - Needs to be done when the student asks for help
- Exact algorithms would often not be done when needed.
  - Everything would stop until the algorithm was done
  - Very intrusive for the student
- Sampling algorithms have the advantage of being *anytime algorithms*
  - They can give you an answer anytime
  - The answer gets better the longer you wait
- So they seemed a good alternative for Andes

Case Study: LW on Andes

Precision	Number of samples	Run time (seconds)
±0.1	1,374,000	400
±0.2	362,000	140
±0.3	5,000	30

- Tested on a network with 110 nodes
  - Run exact algorithm to get "true" probabilities
  - Checked the number of samples and running times to get all nodes in the network within 0.1, 0.2, and 0.3 of the exact probability with all actions in the solution as evidence
- Many networks in Andes have 5 to 10 times the nodes of our test network, and running time of LW increases linearly with the number of nodes
  - It may take several minutes to update nodes in larger networks to a high precision

Case Study: LW on Andes

- Can still be OK when students think before asking for help after an action.
- Also, LW reaches
  - 0.3 precision for all nodes when 98% of the nodes where already at 0.2 precision, and 66% of the nodes where at 0.1 precision
  - 0.2 precision for all nodes when 98% of the nodes where already at 0.1 precision
- Could have still been acceptable in most cases – we were planning to run studies to compute the average waiting time
- But then we found an exact algorithm that works well for most of our networks…

Next Tuesday

- First discussion-based class
- Paper (available on-line from class schedule):
  - Conati C., Gertner A., VanLehn K., 2002. Using Bayesian Networks to Manage Uncertainty in Student Modeling. *User Modeling and User-Adapted Interaction. 12(4) p. 371-417.*
- Make sure to have at least two questions on this reading to discuss in class.

  - See syllabus for more details on what questions should look like
- Send your questions to *both* conati@cs.ubc.ca and ssuther@cs.ubc.ca by 9am on Tuesday.
  - **Please use "questions for 422" as subject**