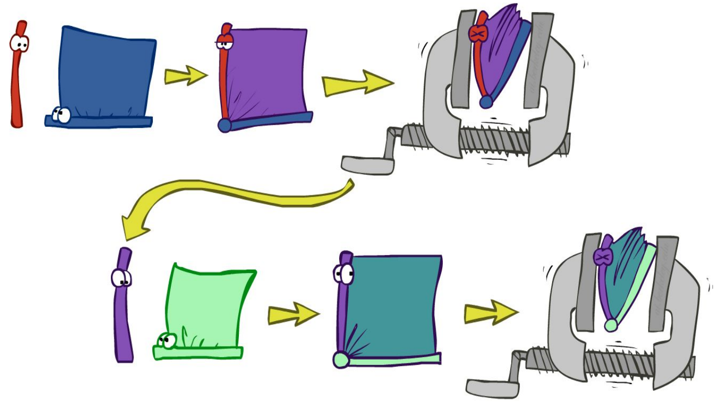
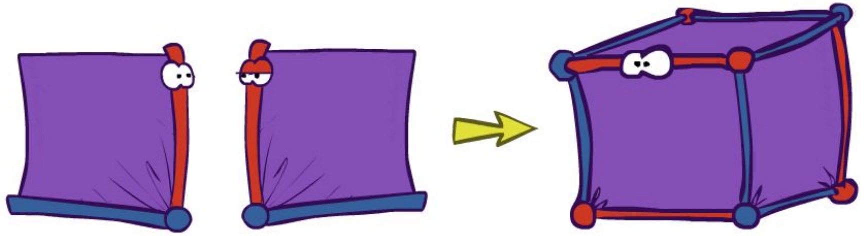
Variable elimination: The basic ideas

- Move summations inwards as far as possible
  - $P(B \mid j, m) = \alpha \sum_{e,a} P(B)\, P(e)\, P(a|B,e)\, P(j|a)\, P(m|a)$
  - $\qquad\qquad\quad = \alpha\, P(B) \sum_{e} P(e) \sum_{a} P(a|B,e)\, P(j|a)\, P(m|a)$
- Do the calculation from the inside out
  - I.e., sum over *a* first, then sum over *e*
  - Problem: *P(a|B,e)* isn't a single number, it's a bunch of different numbers depending on the values of *B* and *e*
  - Solution: use arrays of numbers (of various dimensions) with appropriate operations on them; these are called ***factors***



Operation 1: Pointwise product

- First basic operation: ***pointwise product*** of factors (similar to a ***database join***, ***not*** matrix multiply!)
  - New factor has ***union*** of variables of the two original factors
  - Each entry is the product of the corresponding entries from the original factors



- Example: $P(A,J)$ x $P(A,M)$ = $P(A,J,M)$

$P(A,J)$

A \ J	true	false
true	0.09	0.01
false	0.045	0.855

x

$P(A,M)$

A \ M	true	false
true	0.07	0.03
false	0.009	0.891

=

$P(A,J,M)$

J \ M	true	false
		18

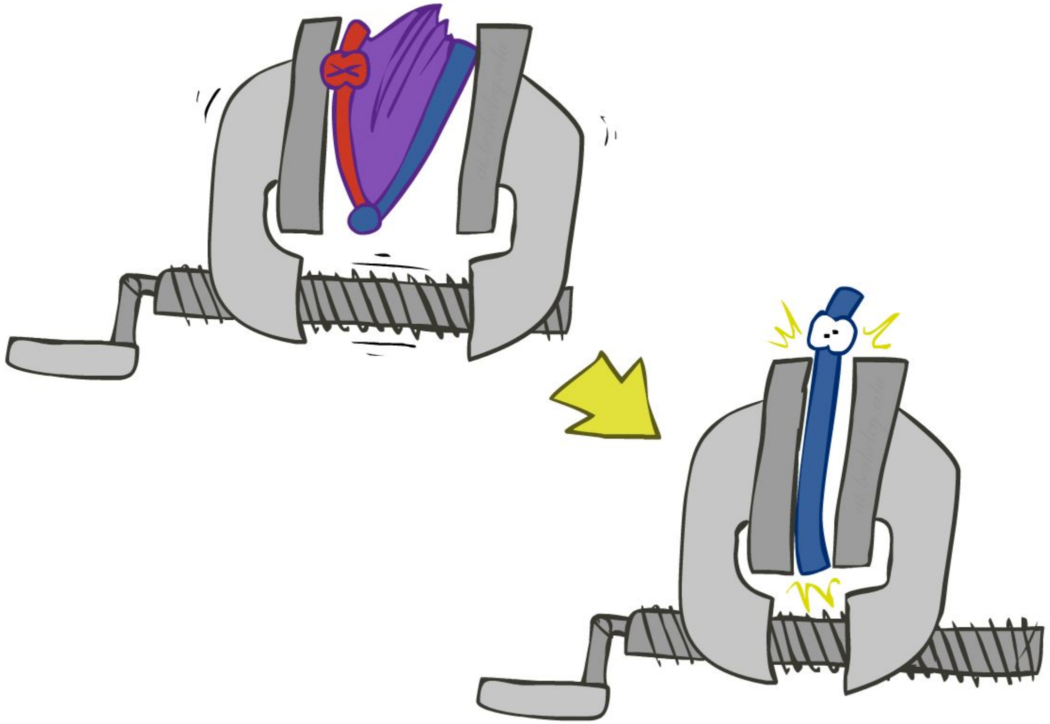
A=false

J \ M	true	false
true		
false		.0003

A=true

Operation 2: Summing out a variable

- Second basic operation: ***summing out*** (or eliminating) a variable from a factor
  - Shrinks a factor to a smaller one
- Example: $\sum_j P(A,J) = P(A,j) + P(A,\neg j) = P(A)$



$P(A,J)$

A \ J	true	false
true	0.09	0.01
false	0.045	0.855

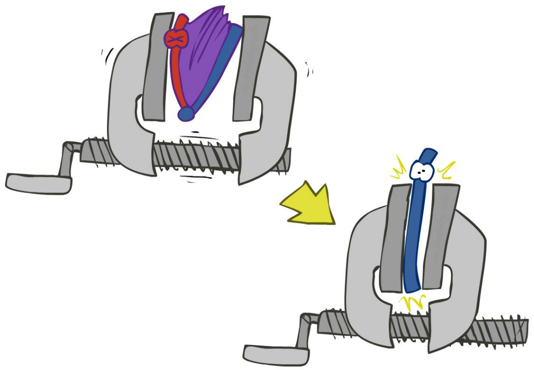
Sum out J

$P(A)$

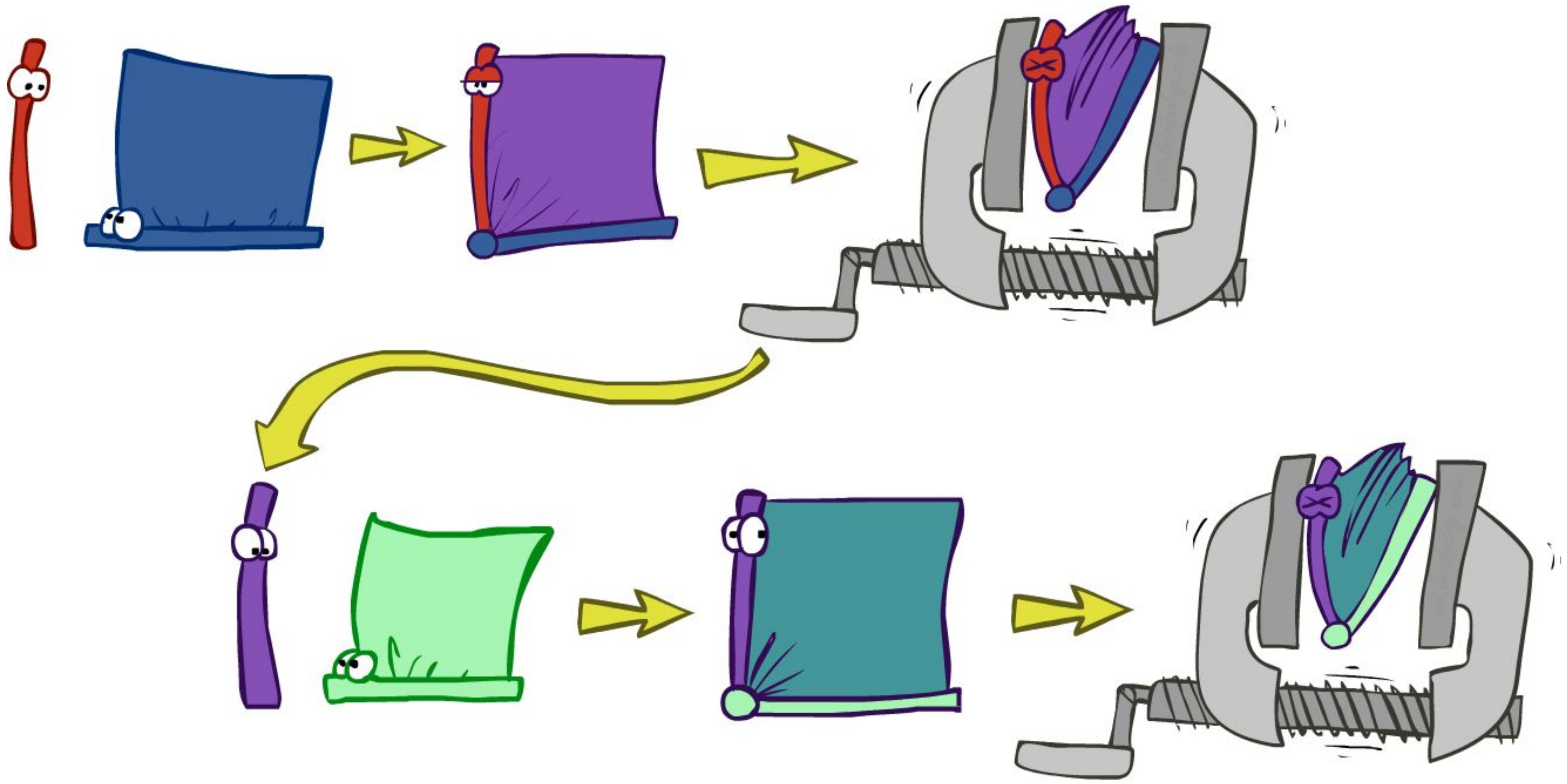
true	0.1
false	0.9

Summing out from a product of factors

- Project the factors each way first, then sum the products
- Example: $\sum_a P(a \mid B,e) \times P(j \mid a) \times P(m \mid a)$
- $= P(a \mid B,e) \times P(j \mid a) \times P(m \mid a) +$
- $P(\neg a \mid B,e) \times P(j \mid \neg a) \times P(m \mid \neg a)$

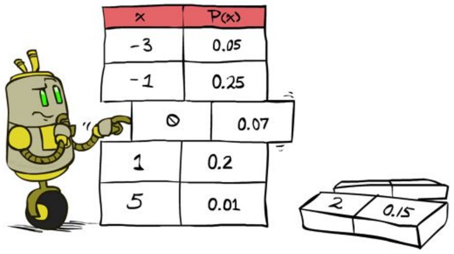


Variable Elimination

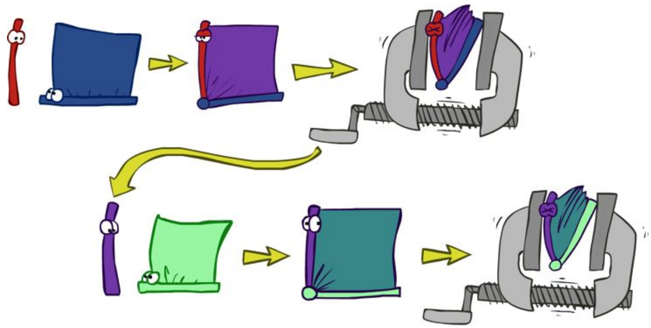


Variable Elimination

- Query: $P(\boldsymbol{Q}|\boldsymbol{e})$

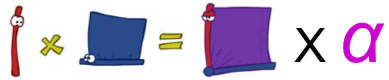


- Start with initial factors:
  - Local CPTs (but instantiated by evidence)



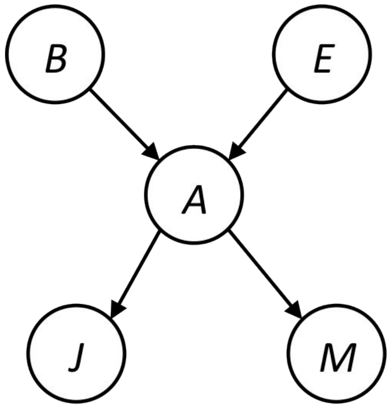
- For each hidden variable H_j
  - Sum out H_j from the product of all factors mentioning H_j

- Join all remaining factors and normalize



Example

Query $P(B \mid j,m)$



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

Choose A

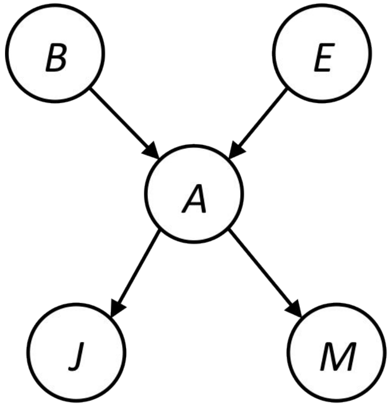
$P(A|B,E)$
$P(j|A)$
$P(m|A)$

$\times \;\rightarrow\; \sum \;\rightarrow\; P(j,m|B,E)$

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

Example

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



Choose E

$P(E)$
$P(j,m|B,E)$ $\xrightarrow{\times}$ $\xrightarrow{\sum}$ $P(j,m|B)$

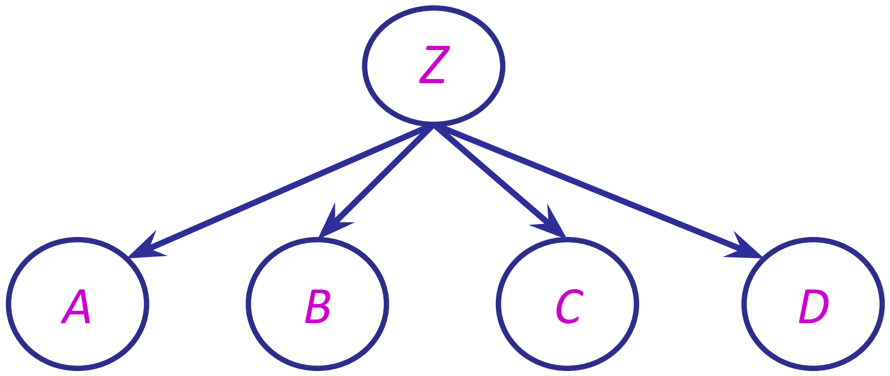
$P(B)$ $P(j,m|B)$

Finish with B

$P(B)$
$P(j,m|B)$ $\xrightarrow{\times}$ $P(j,m,B)$ $\xrightarrow{\text{Normalize}}$ $P(B \mid j,m)$

Order matters

- Order the terms *Z, A, B C, D*
  - $P(D) = \alpha \sum_{z,a,b,c} P(z)\, P(a|z)\, P(b|z)\, P(c|z)\, P(D|z)$
  - $= \alpha \sum_z P(z) \sum_a P(a|z) \sum_b P(b|z) \sum_c P(c|z)\, P(D|z)$
  - Largest factor has 2 variables (*D,Z*)
- Order the terms *A, B C, D, Z*
  - $P(D) = \alpha \sum_{a,b,c,z} P(a|z)\, P(b|z)\, P(c|z)\, P(D|z)\, P(z)$
  - $= \alpha \sum_a \sum_b \sum_c \sum_z P(a|z)\, P(b|z)\, P(c|z)\, P(D|z)\, P(z)$
  - Largest factor has 4 variables (*A,B,C,D*)
  - In general, with *n* leaves, factor of size 2^n



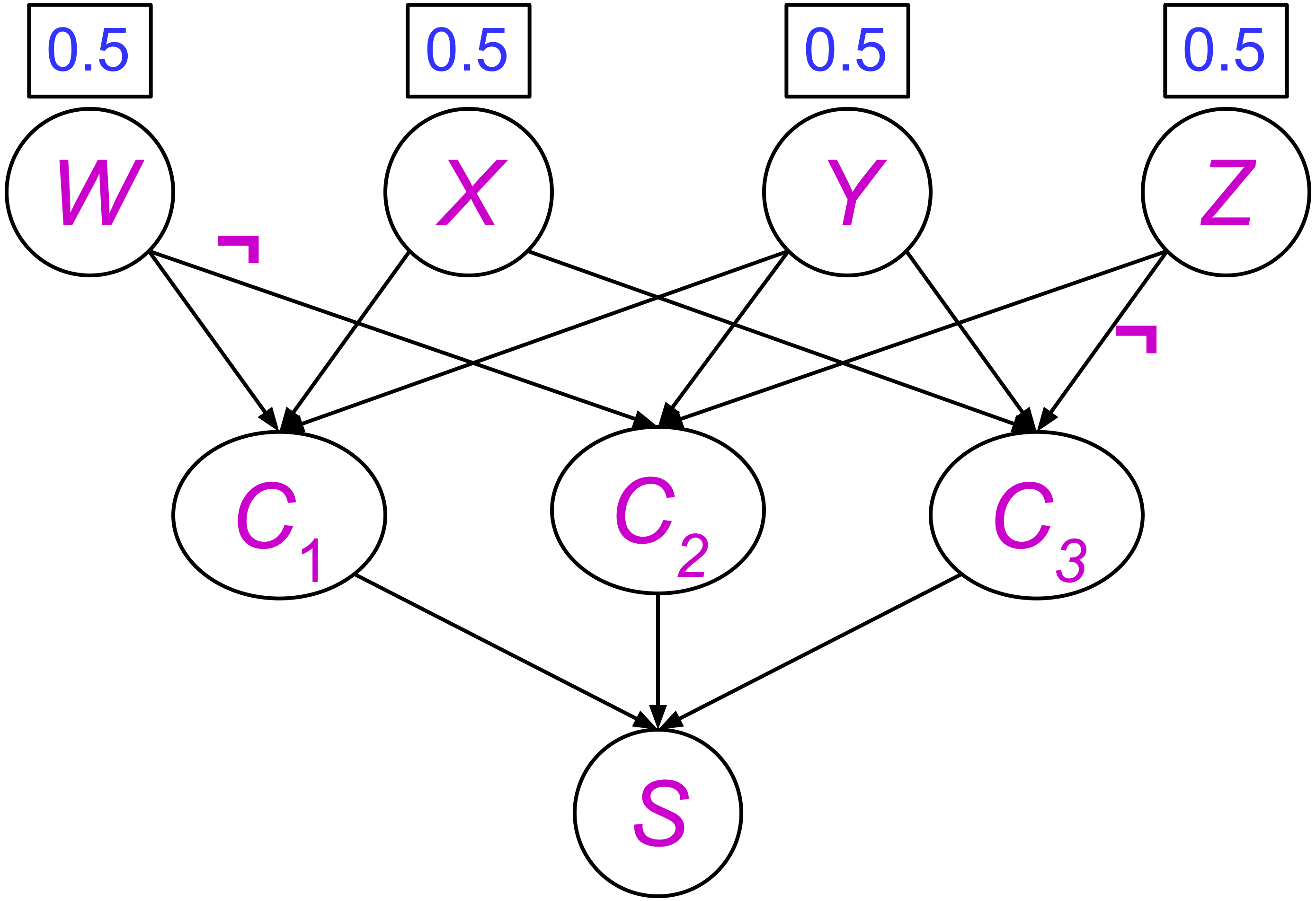
VE: Computational and Space Complexity

- The computational and space complexity of variable elimination is determined by the largest factor (and it's space that kills you)

- 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? Reduction from SAT

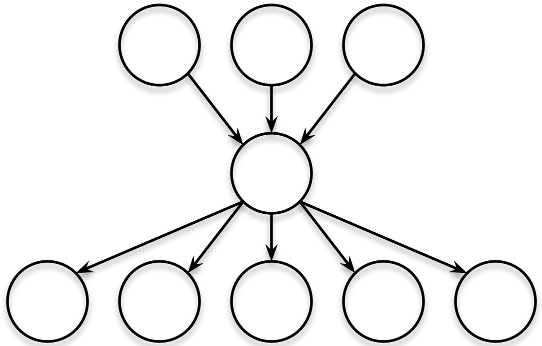


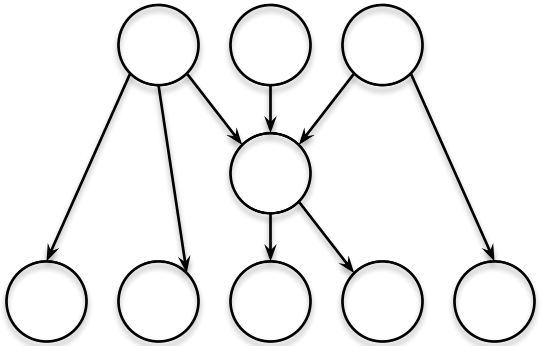
- Variables: *W, X, Y, Z*
- CNF clauses:
  - $C_1 = W \vee X \vee Y$
  - $C_2 = Y \vee Z \vee \neg W$
  - $C_3 = X \vee Y \vee \neg Z$
- Sentence $S = C_1 \wedge C_2 \wedge C_3$
- $P(S) > 0$ iff S is satisfiable
  - => ***NP-hard***
- $P(S) = K \times 0.5^n$ where K is the number of satisfying assignments for clauses
  - => ***#P-hard***

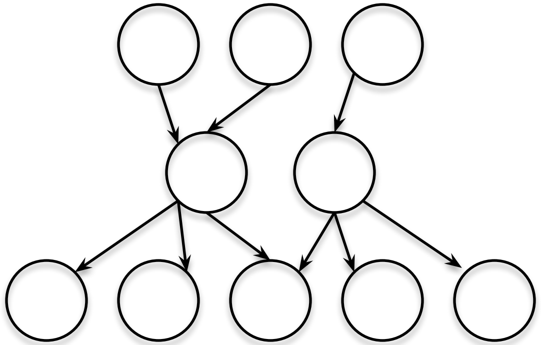
Polytrees

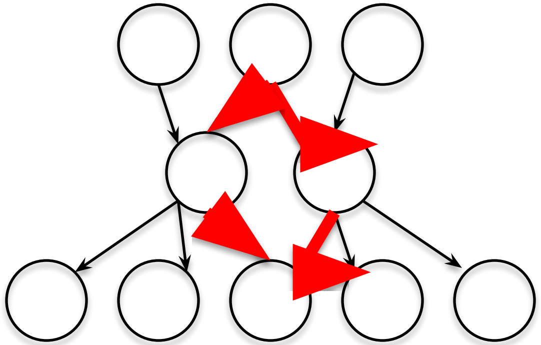
- A polytree is a directed graph with no undirected cycles

- For poly-trees the complexity of variable elimination is ***linear in the network size*** if you eliminate from the leaves towards the roots



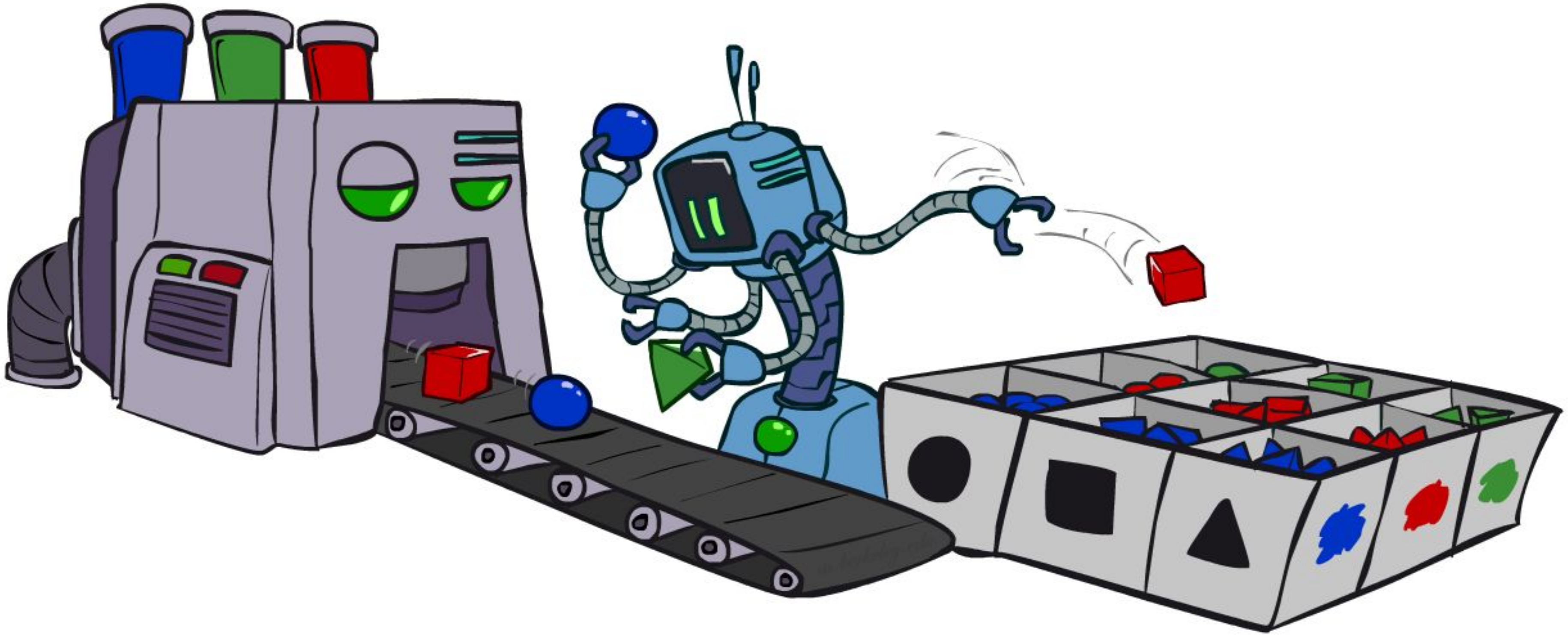






CS 188: Artificial Intelligence

Bayes Nets: Approximate Inference



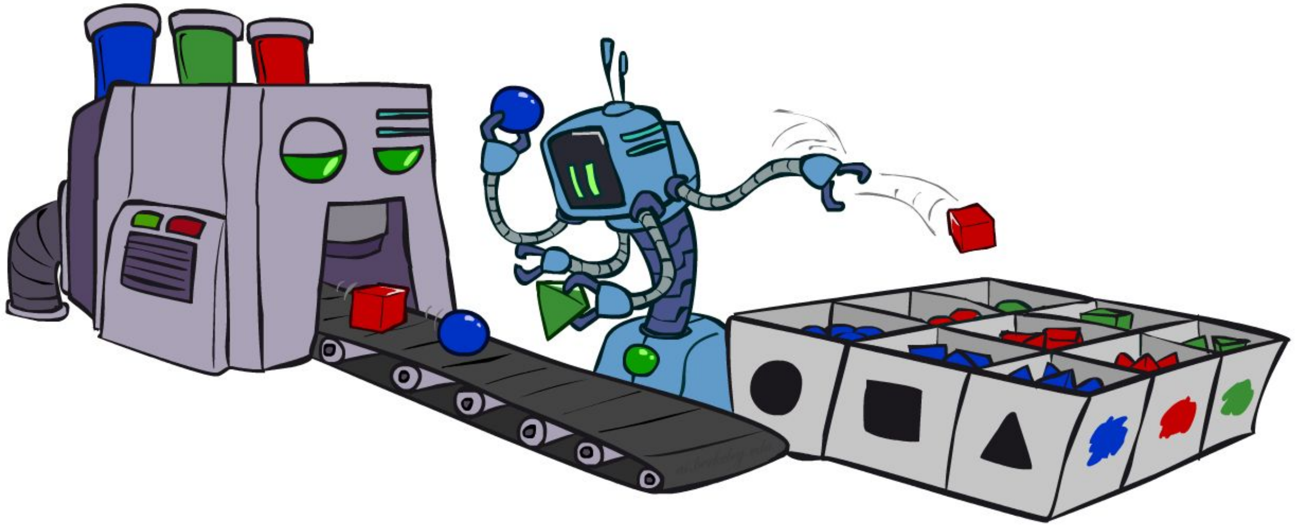
Instructors: Stuart Russell and Dawn Song

University of California, Berkeley

Sampling

- 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?
  - Often very fast to get a decent approximate answer
  - The algorithms are very simple and general (easy to apply to fancy models)
  - They require very little memory ($O(n)$)
  - They can be applied to large models, whereas exact algorithms blow up

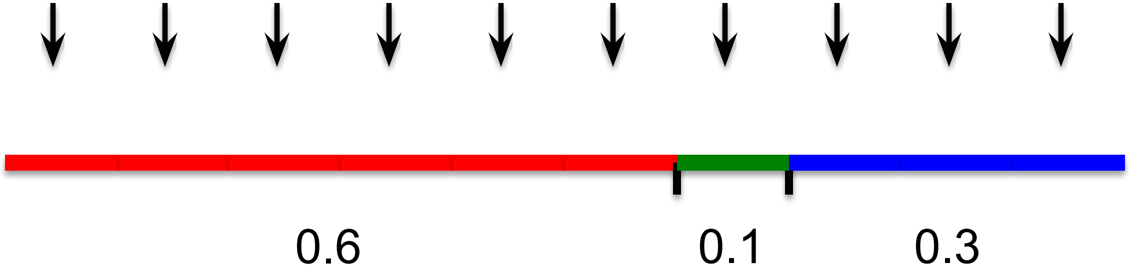


Example

- Suppose you have two agent programs ***A*** and ***B*** for Monopoly
- What is the probability that ***A*** wins?
  - Method 1:
    - Let s be a sequence of dice rolls and Chance and Community Chest cards
    - Given s, the outcome $V(s)$ is determined (1 for a win, 0 for a loss)
    - Probability that ***A*** wins is $\sum_s P(s)\ V(s)$
    - Problem: infinitely many sequences s !
  - Method 2:
    - Sample N sequences from $P(s)$, play N games (maybe 100)
    - Probability that ***A*** wins is roughly $1/N \sum_i V(s_i)$ i.e., fraction of wins in the sample

Sampling basics: discrete (*categorical*) distribution

- To simulate a biased d-sided coin:
  - Step 1: Get sample *u* from uniform distribution over [0, 1)
    - E.g. random() in python
  - Step 2: Convert this sample *u* into an outcome for the given distribution by associating each outcome *x* with a *P*(*x*)-sized sub-interval of [0,1)

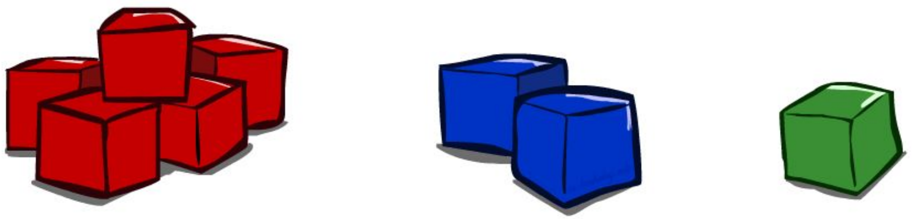


- Example

C	*P*(*C*)
red	0.6
green	0.1
blue	0.3

$0.0 \le u < 0.6, \rightarrow C = red$
$0.6 \le u < 0.7, \rightarrow C = green$
$0.7 \le u < 1.0, \rightarrow C = blue$

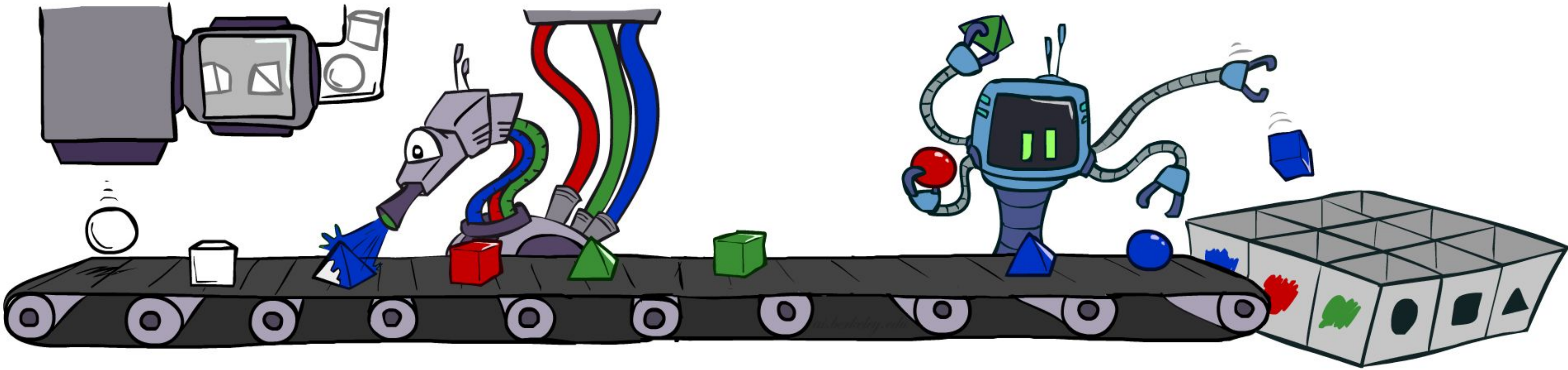
- If random() returns *u* = 0.83, then the sample is *C* = *blue*
- E.g, after sampling 8 times:



Sampling in Bayes Nets

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

Prior Sampling



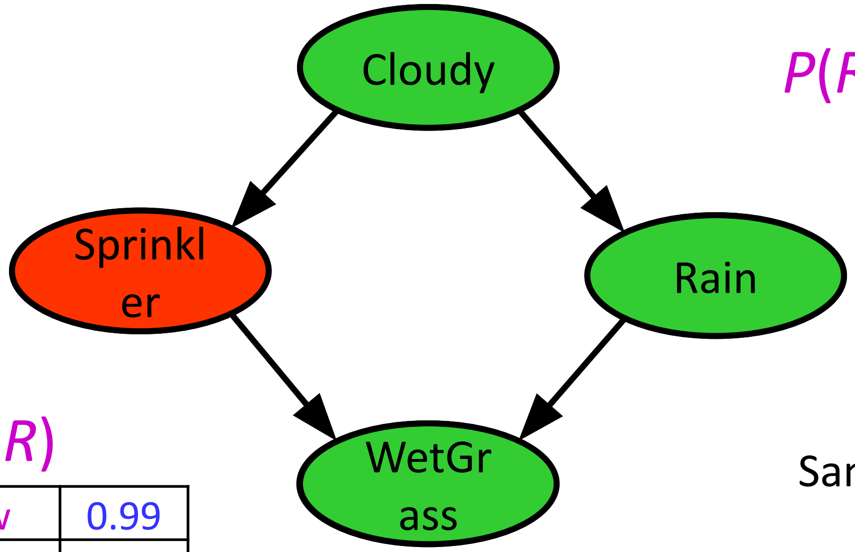
Prior Sampling

$P(C)$

c	0.5
¬c	0.5

$P(S \mid C)$

c	s	0.1
	¬s	0.9
¬c	s	0.5
	¬s	0.5



$P(R \mid C)$

c	r	0.8
	¬r	0.2
¬c	r	0.2
	¬r	0.8

$P(W \mid S,R)$

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:

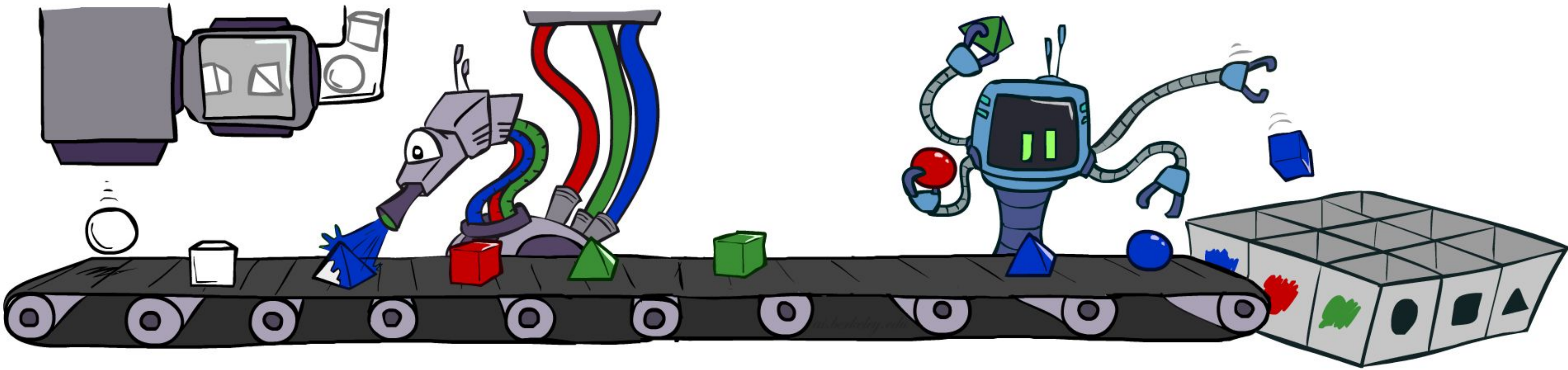
c, ¬s, r, w

¬c, s, ¬r, w

…

Prior Sampling

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



Prior Sampling

- This process generates samples with probability:

  $S_{PS}(x_1,\ldots,x_n) = \prod_i P(x_i \mid parents(X_i)) = P(x_1,\ldots,x_n)$

  …i.e. the BN's joint probability

- Let the number of samples of an event be $N_{PS}(x_1,\ldots,x_n)$
- Estimate from N samples is $Q_N(x_1,\ldots,x_n) = N_{PS}(x_1,\ldots,x_n)/N$
- Then $\lim_{N\to\infty} Q_N(x_1,\ldots,x_n) = \lim_{N\to\infty} N_{PS}(x_1,\ldots,x_n)/N$

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

  $= P(x_1,\ldots,x_n)$
- I.e., the sampling procedure is ***consistent***

Example

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

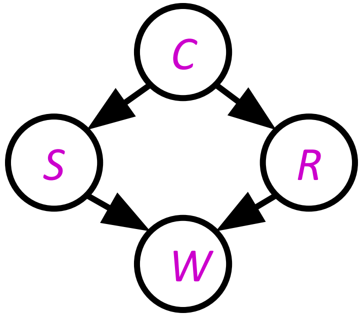
  c, ¬s, r, w

  c, s, r, w

  ¬c, s, r, ¬w

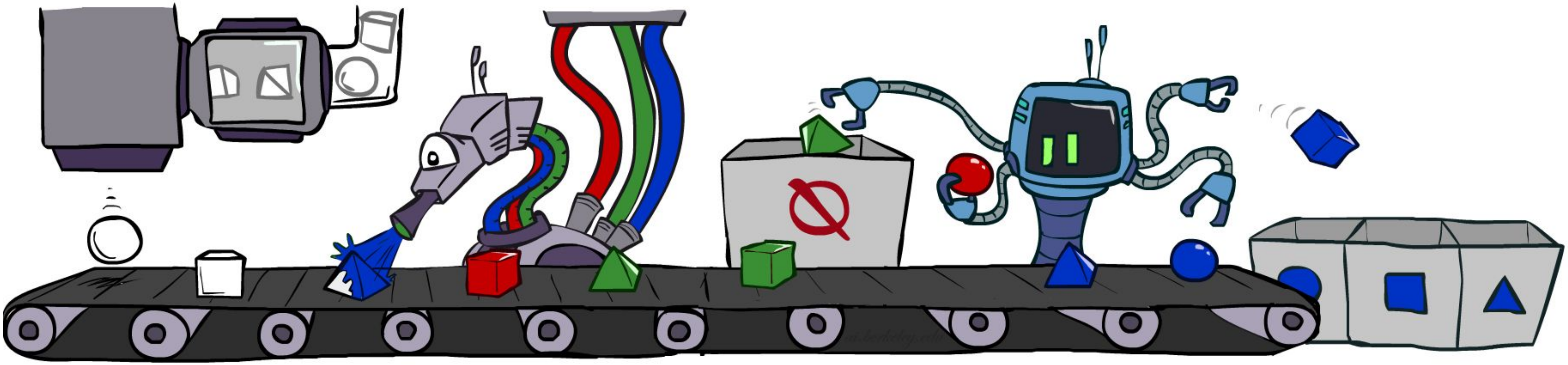
  c, ¬s, r, w

  ¬c, ¬s, ¬r, w



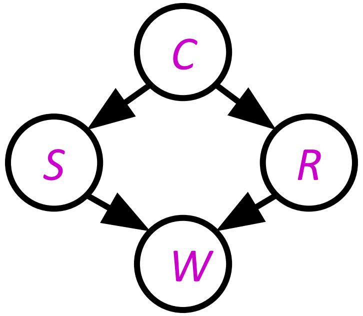
- If we want to know $P(W)$
  - We have counts <w:4, ¬w:1>
  - Normalize to get $P(W)$ = <w:0.8, ¬w:0.2>
  - This will get closer to the true distribution with more samples
  - Can estimate anything else, too
    - E.g., for query $P(C|\ r, w)$ use $P(C|\ r, w) = \alpha\, P(C, r, w)$

Rejection Sampling



Rejection Sampling

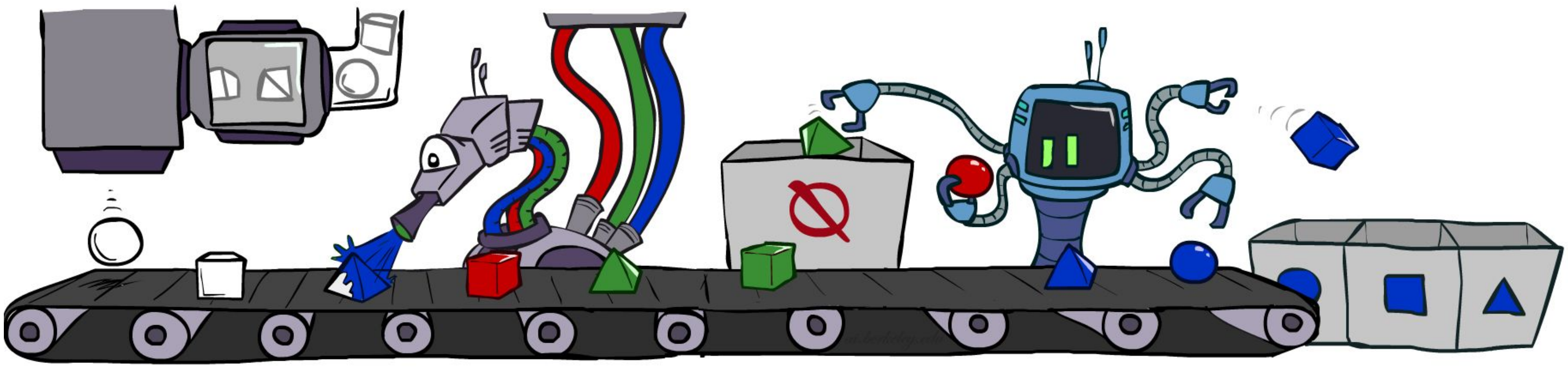
- A simple modification of prior sampling for conditional probabilities
- Let's say we want P(*C*| *r, w*)
- Count the *C* outcomes, but ignore (reject) samples that don't have *R*=true, *W*=true
  - This is called ***rejection sampling***
  - It is also consistent for conditional probabilities (i.e., correct in the limit)



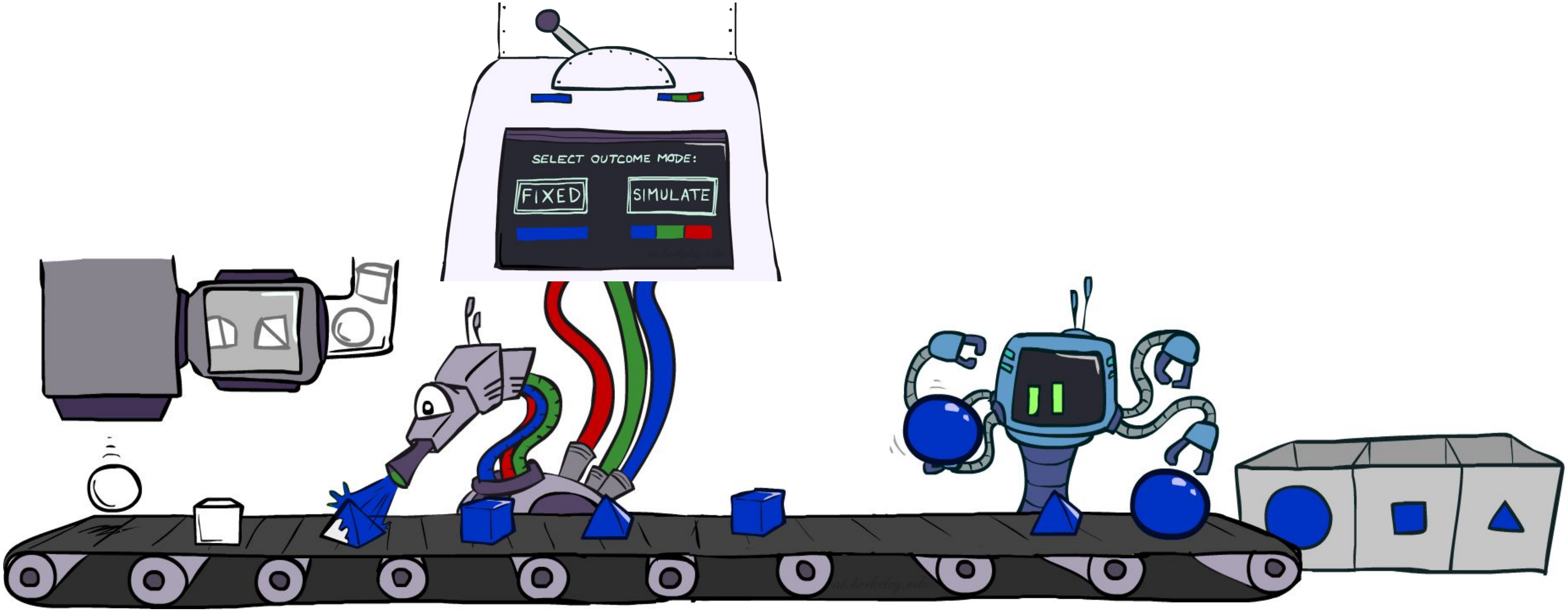
c, ¬s, r, w
~~c, s, ¬r~~
~~¬c, s, r, ¬w~~
~~c, ¬s, ¬r~~
¬c, ¬s, r, w

Rejection Sampling

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



Likelihood Weighting

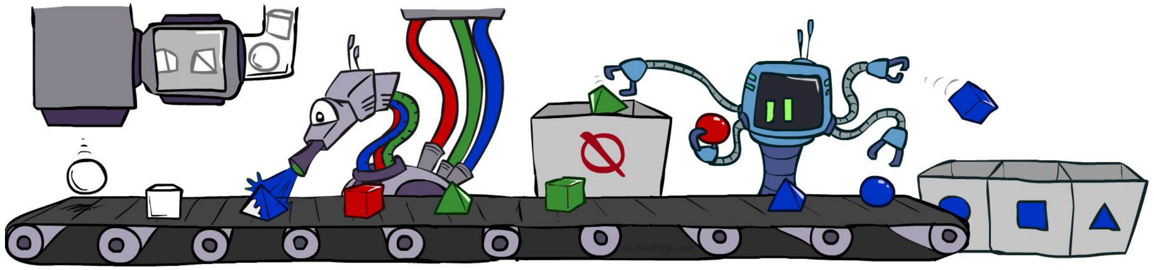


Likelihood Weighting

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

Shape → Color

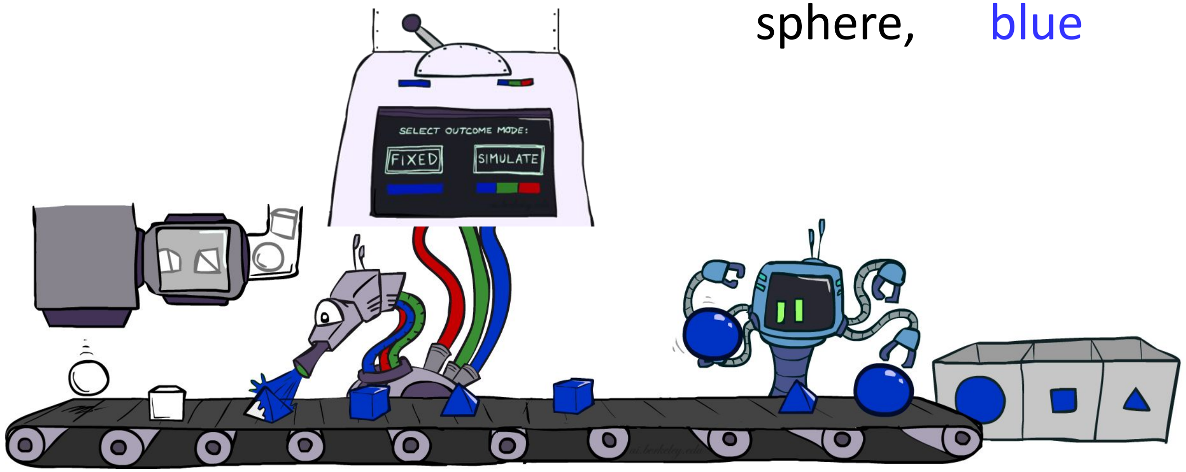
~~pyramid, green~~
~~pyramid, red~~
sphere, blue
~~cube, red~~
~~sphere, green~~



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

Shape → ~~Color~~

pyramid, blue
pyramid, blue
sphere, blue
cube, blue
sphere, blue



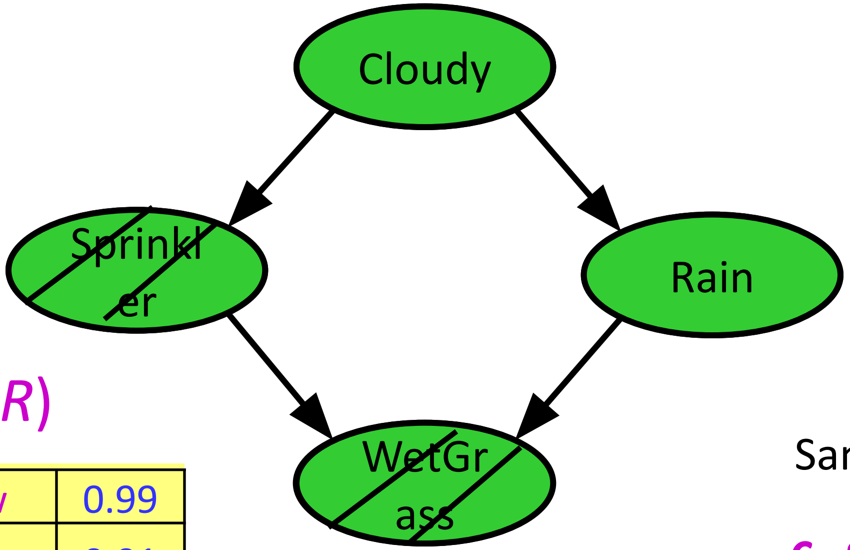
Likelihood Weighting

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*)

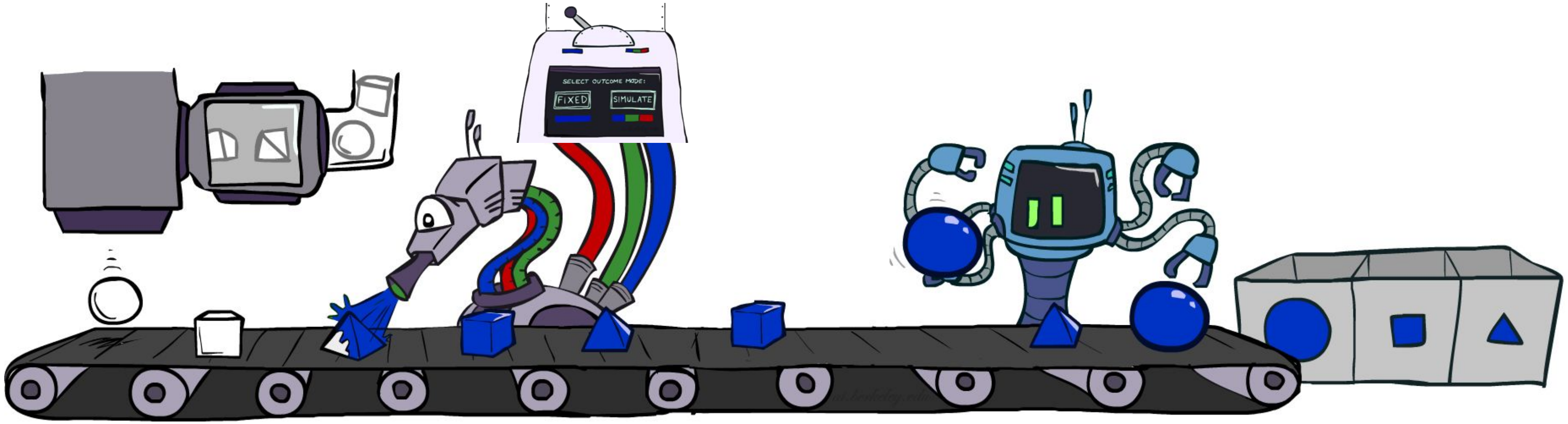
s	r	w	0.99
		¬w	0.01
	¬r	w	0.90
		¬w	0.10
¬s	r	w	0.90
		¬w	0.10
	¬r	w	0.01
		¬w	0.99

Samples:

c*, *s*, *r*, *w **_w_ = 1.0 x 0.1 x 0.99**

Likelihood Weighting

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



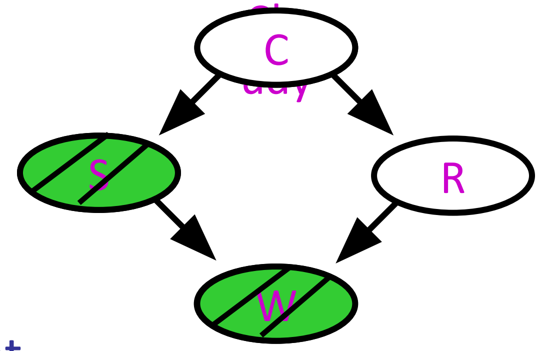
Likelihood Weighting

- Sampling distribution if **Z** sampled and **e** fixed evidence

  $$S_{WS}(\mathbf{z},\mathbf{e}) = \prod_j P(z_j \mid parents(Z_j))$$

- Now, samples have weights

  $$w(\mathbf{z},\mathbf{e}) = \prod_k P(e_k \mid parents(E_k))$$



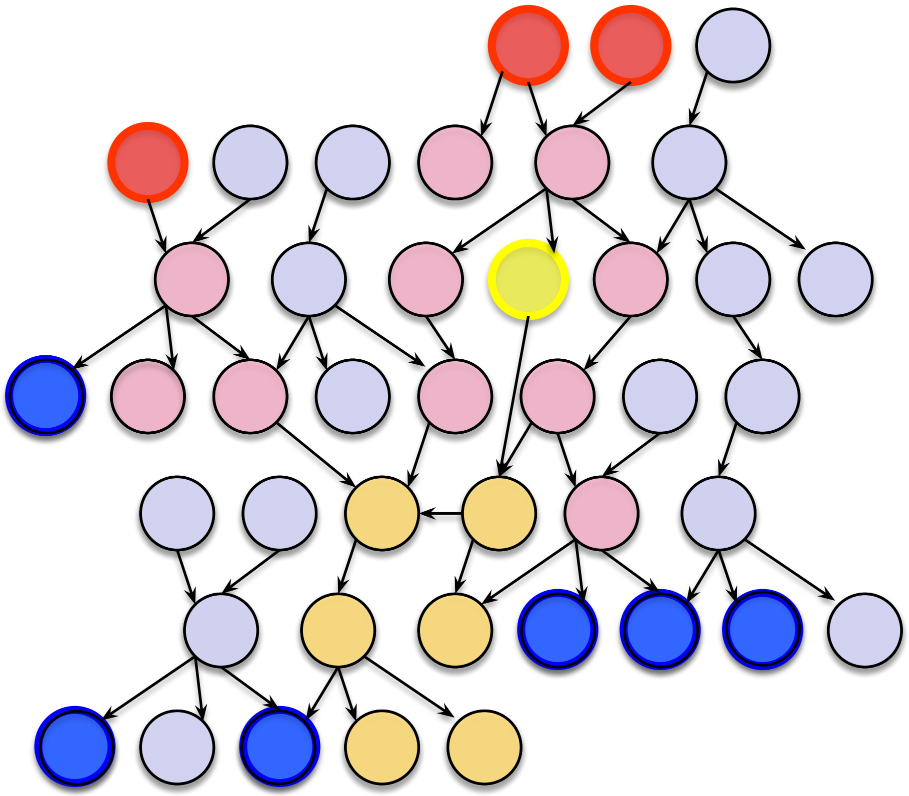
- Together, weighted sampling distribution is consistent

  $$S_{WS}(\mathbf{z},\mathbf{e}) \cdot w(\mathbf{z},\mathbf{e}) = \prod_j P(z_j \mid parents(Z_j)) \prod_k P(e_k \mid parents(E_k))$$
  $$= P(\mathbf{z},\mathbf{e})$$

- Likelihood weighting is an example of ***importance sampling***
  - Would like to estimate some quantity based on samples from P
  - P is hard to sample from, so use Q instead
  - Weight each sample x by $P(x)/Q(x)$

Likelihood Weighting

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



- Likelihood weighting still has weaknesses
  - The values of ***upstream*** variables are unaffected by ***downstream*** evidence
    - E.g., suppose evidence is a video of a traffic accident
  - With evidence in k leaf nodes, weights will be $O(2^{-k})$
  - With high probability, one lucky sample will have much larger weight than the others, dominating the result

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

Quiz

- Suppose I perform a random walk on a graph, following the arcs out of a node ***uniformly at random***. In the infinite limit, what fraction of time do I spend at each node?
  - Consider these two examples: