Lecture 10: Bayes Nets: Inference

Shuai Li

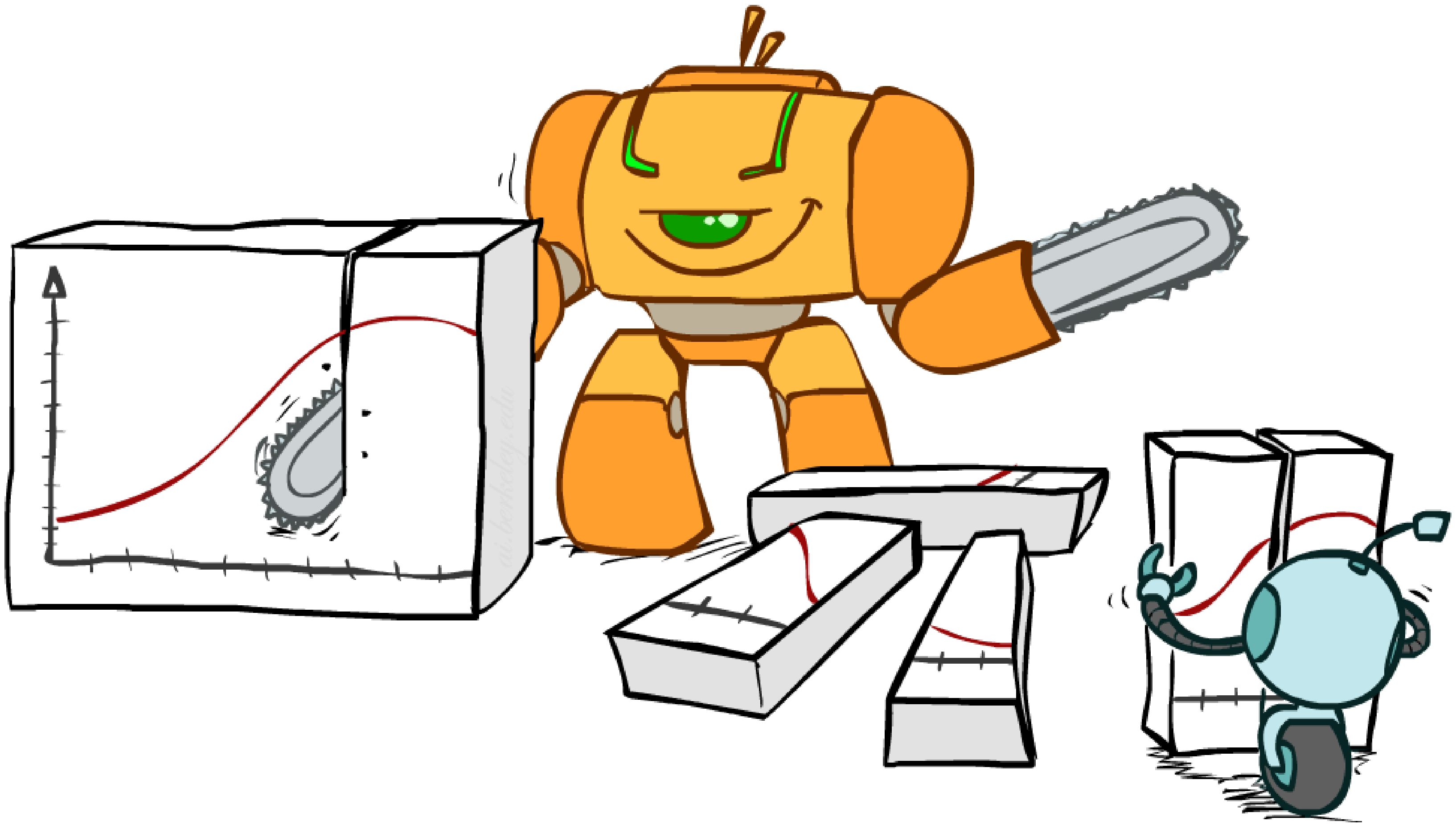
John Hopcroft Center, Shanghai Jiao Tong University

https://shuaili8.github.io

https://shuaili8.github.io/Teaching/CS3317/index.html

Part of slide credits: CMU AI & http://ai.berkeley.edu

Bayes Rule



Bayes' Rule

- Two ways to factor a joint distribution over two variables:

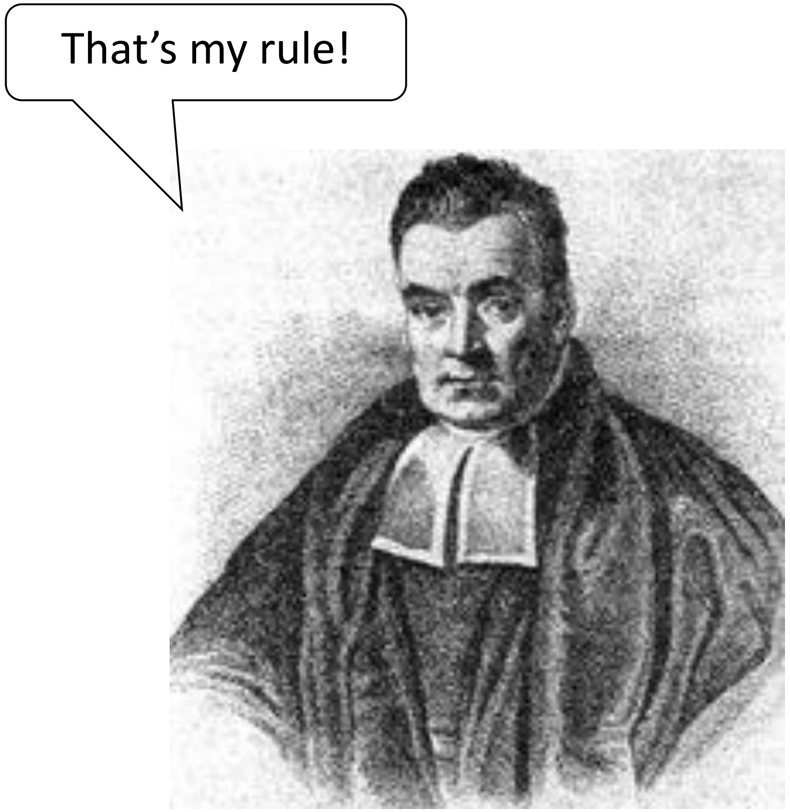
$$P(x,y) = P(x|y)P(y) = P(y|x)P(x)$$

- Dividing, we get:

$$P(x|y) = \frac{P(y|x)}{P(y)}P(x)$$

- Why is this at all helpful?
  - Lets us build one conditional from its reverse
  - Often one conditional is tricky but the other one is simple
  - Foundation of many systems (e.g. ASR, MT)

- In the running for most important AI equation!



Inference with Bayes' Rule

- Example: Diagnostic probability from causal probability:

$$P(\text{cause}|\text{effect}) = \frac{P(\text{effect}|\text{cause})P(\text{cause})}{P(\text{effect})}$$

- Example:
  - M: meningitis, S: stiff neck

$$\left.\begin{array}{r} P(+m) = 0.0001 \\ P(+s|+m) = 0.8 \\ P(+s|-m) = 0.01 \end{array}\right\} \text{Example givens}$$

$$P(+m|+s) = \frac{P(+s|+m)P(+m)}{P(+s)} = \frac{P(+s|+m)P(+m)}{P(+s|+m)P(+m) + P(+s|-m)P(-m)} = \frac{0.8 \times 0.0001}{0.8 \times 0.0001 + 0.01 \times 0.999}$$

  - Note: posterior probability of meningitis still very small
  - Note: you should still get stiff necks checked out! Why?

Quiz: Bayes' Rule

- Given:

$P(W)$

R	P
sun	0.8
rain	0.2

$P(D|W)$

D	W	P
wet	sun	0.1
dry	sun	0.9
wet	rain	0.7
dry	rain	0.3

- What is P(W | dry) ?

Quiz: Bayes' Rule 2

- Given:

$P(W)$

R	P
sun	0.8
rain	0.2

$P(D|W)$

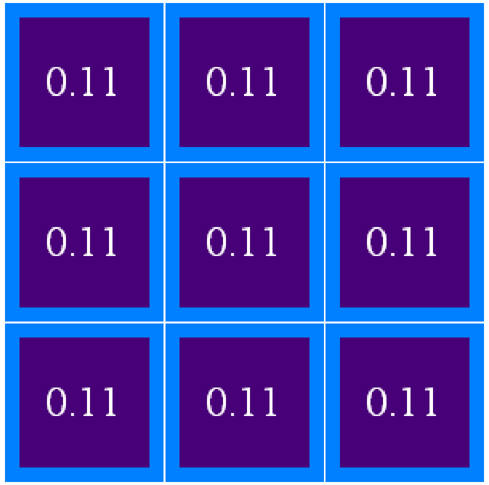
D	W	P
wet	sun	0.1
dry	sun	0.9
wet	rain	0.7
dry	rain	0.3

- What is P(W | dry) ?

P(sun|dry) ∝ P(dry|sun)P(sun) = .9*.8 = .72
P(rain|dry) ∝ P(dry|rain)P(rain) = .3*.2 = .06
P(sun|dry)=12/13
P(rain|dry)=1/13

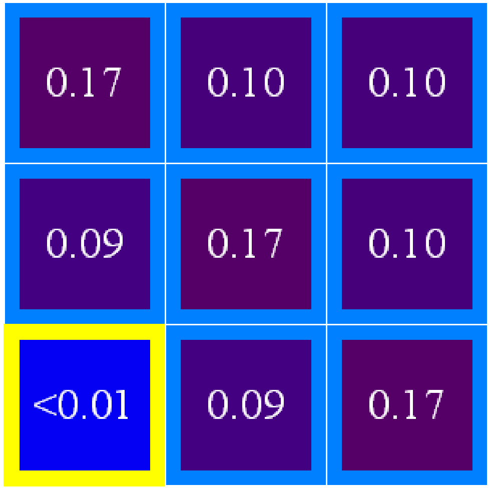
Ghostbusters, Revisited

- Let’s say we have two distributions:
  - Prior distribution over ghost location: P(G)
    - Let’s say this is uniform
  - Sensor reading model: P(R | G)
    - Given: we know what our sensors do
    - R = reading color measured at (1,1)
    - E.g. P(R = yellow | G=(1,1)) = 0.1



- We can calculate the posterior distribution P(G|r) over ghost locations given a reading using Bayes’ rule:

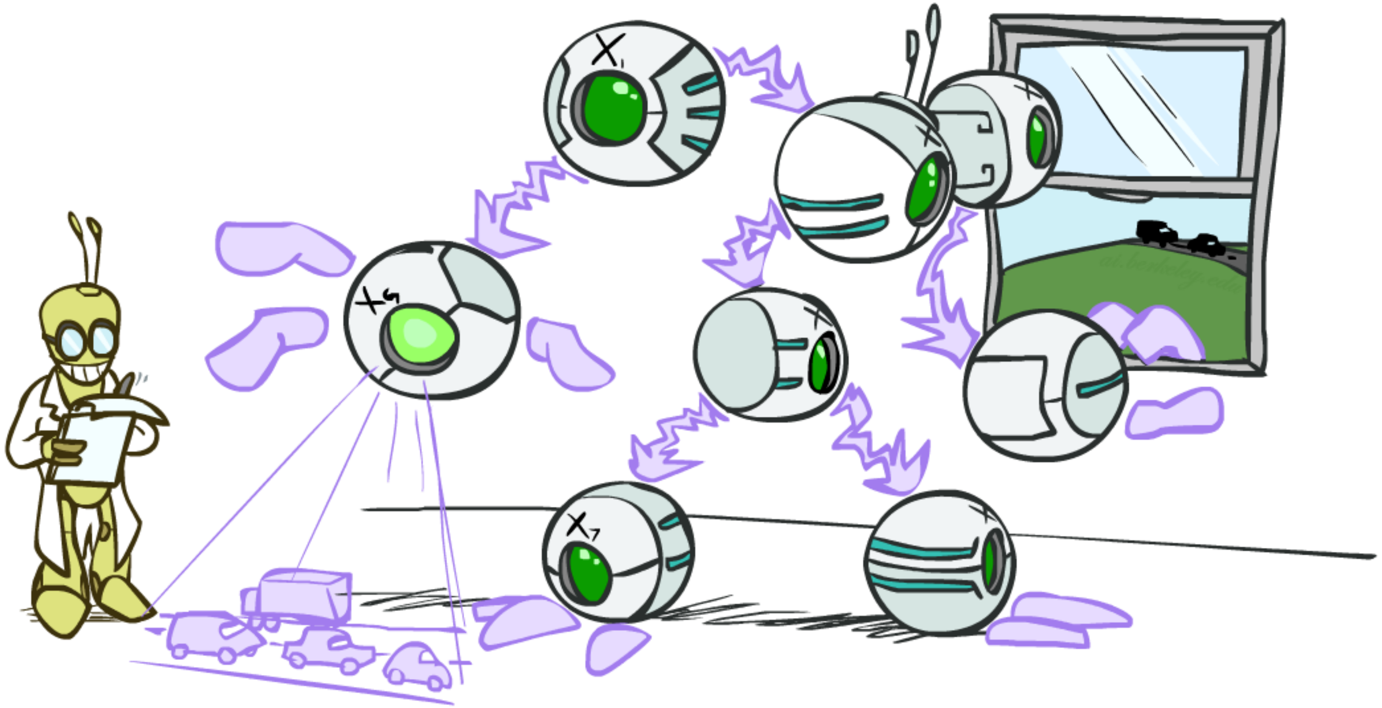
$$P(g|r) \propto P(r|g)P(g)$$



[Demo: Ghostbuster – with probability (L12D2)]

Video of Demo Ghostbusters with Probability

Inference



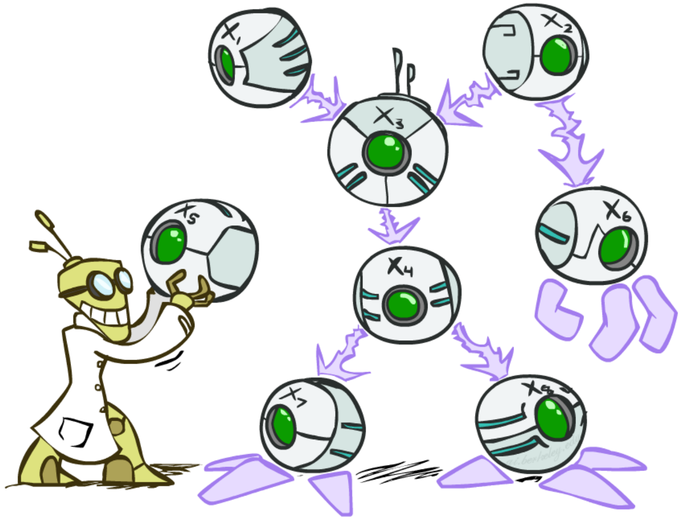
Recall: Bayes' Net Representation

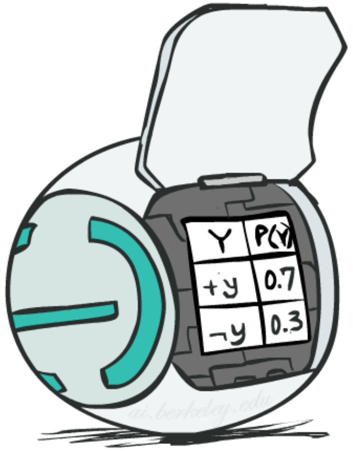
- A directed, acyclic graph, one node per random variable
- A conditional probability table (CPT) for each node
  - A collection of distributions over X, one for each combination of parents' values

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

- Bayes' nets implicitly encode joint distributions
  - As a product of local conditional distributions
  - To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together:

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





Inference

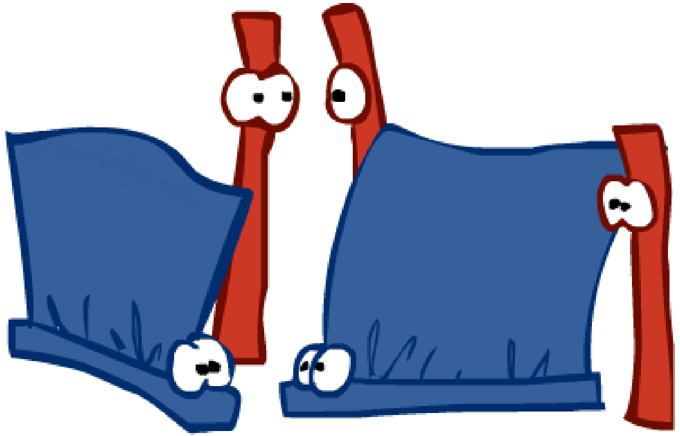
- Inference: calculating some useful quantity from a joint probability distribution

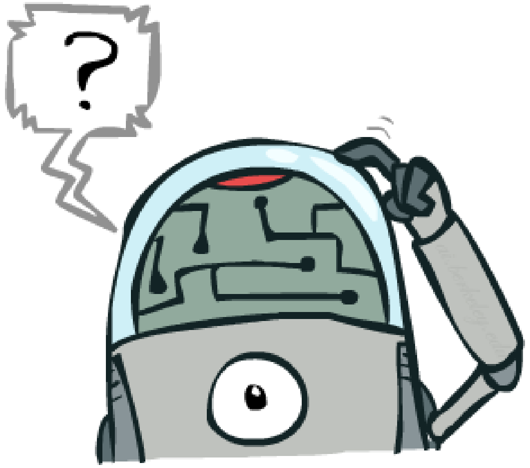
- **Examples:**
  - Posterior probability

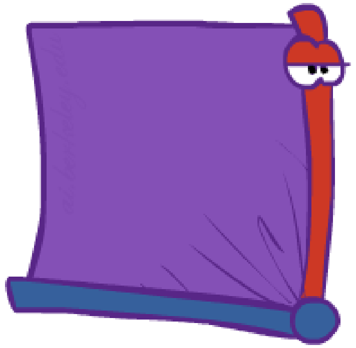
    $$P(Q|E_1 = e_1, \ldots E_k = e_k)$$

  - Most likely explanation:

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







Queries

- What is the probability of *this* given what I know?

$$P(\, q \mid e\,) = \frac{P(q,\ \ e)}{P(e)} = \frac{\sum_{h_1} \sum_{h_2} P(q,\ \ h_1,\ \ h_2,\ \ e)}{P(e)}$$

- What are the probabilities of all the possible outcomes (given what I know)?

$$P(\, Q \mid e\,) = \frac{P(Q,\ \ e)}{P(e)} = \frac{\sum_{h_1} \sum_{h_2} P(Q,\ \ h_1,\ \ h_2,\ \ e)}{P(e)}$$

- Which outcome is the most likely outcome (given what I know)?

$$\operatorname{argmax}_{q \in Q} P(\, q \mid e\,) = \operatorname{argmax}_{q \in Q} \frac{P(q,\ \ e)}{P(e)}$$

$$= \operatorname{argmax}_{q \in Q} \frac{\sum_{h_1} \sum_{h_2} P(q,\ \ h_1,\ \ h_2,\ \ e)}{P(e)}$$

Inference by Enumeration in Joint Distributions

- General case:
  - Evidence variables: $E_1 \ldots E_k = e_1 \ldots e_k$
  - Query* variable: Q
  - Hidden variables: $H_1 \ldots H_r$

$X_1, X_2, \ldots X_n$

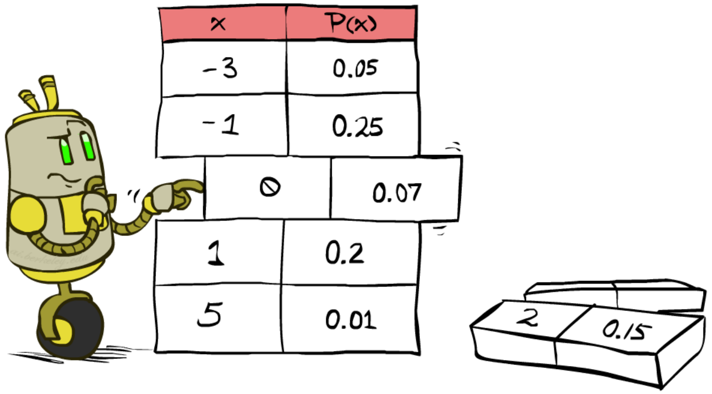
All variables

- We want:

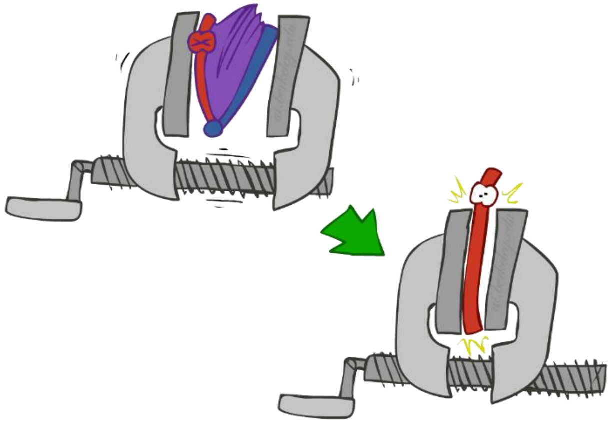
** Works fine with multiple query variables, too*

$$P(Q|e_1 \ldots e_k)$$

- Step 1: Select the entries consistent with the evidence



- Step 2: Sum out H to get joint of Query and evidence



$$P(Q, e_1 \ldots e_k) = \sum_{h_1 \ldots h_r} P(\underbrace{Q, h_1 \ldots h_r, e_1 \ldots e_k}_{X_1, X_2, \ldots X_n})$$

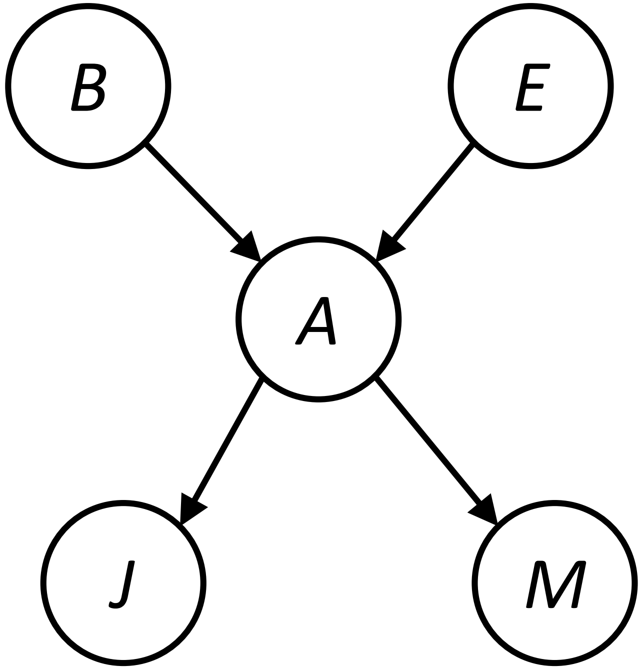
- Step 3: Normalize

$$\times \frac{1}{Z}$$

$$Z = \sum_q P(Q, e_1 \cdots e_k)$$

$$P(Q|e_1 \cdots e_k) = \frac{1}{Z} P(Q, e_1 \cdots e_k)$$

Inference by Enumeration in Bayes' Net



- Given unlimited time, inference in BNs is easy

$$P(B \mid +j, +m) \propto_B P(B, +j, +m)$$

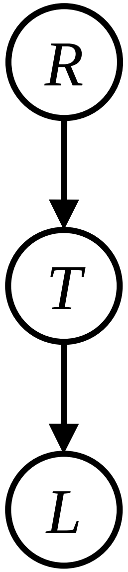
$$= \sum_{e,a} P(B, e, a, +j, +m)$$

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

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

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

$$P(L) = ?$$

$$= \sum_{r,t} P(r, t, L)$$

$$= \sum_{r,t} P(r)P(t|r)P(L|t)$$

Inference by Enumeration: Procedural 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

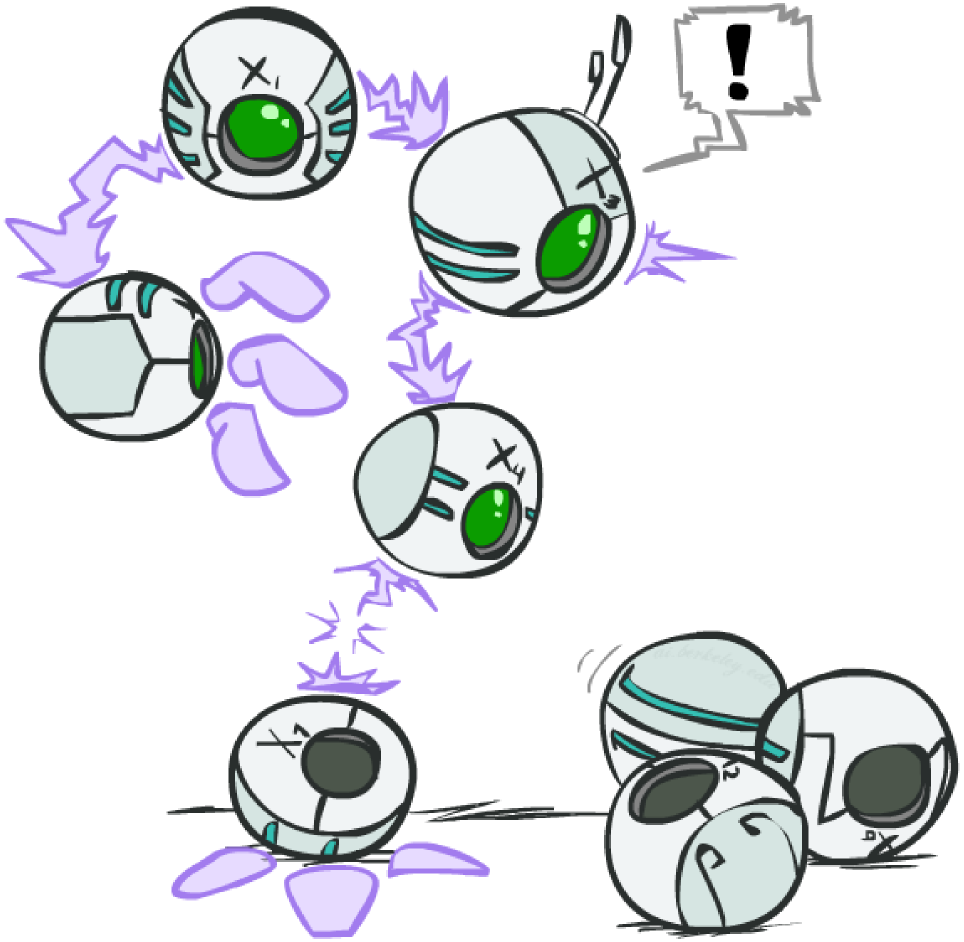
$P(T|R)$

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

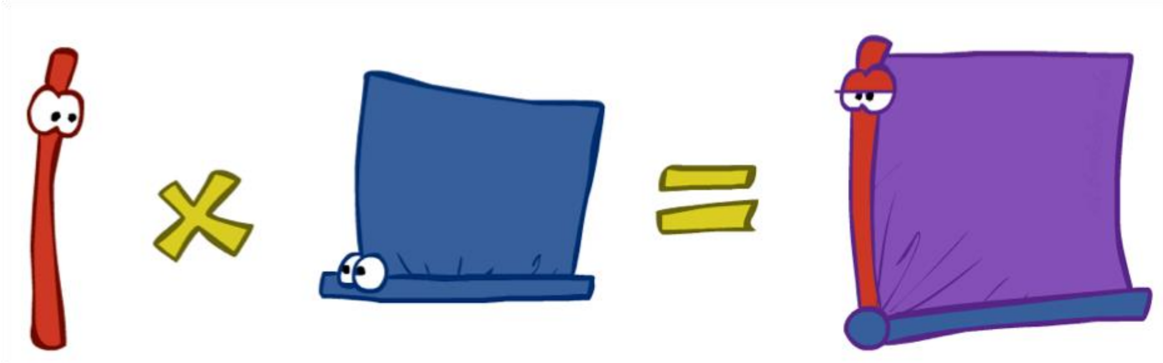
$P(+\ell|T)$

+t	+l	0.3
-t	+l	0.1

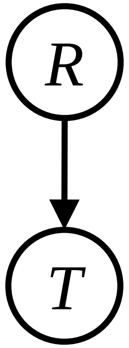
- Procedure: Join all factors, then sum out all hidden variables



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



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

$P(R)$

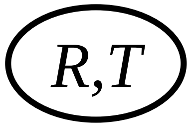
+r	0.1
-r	0.9

$P(T|R)$

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

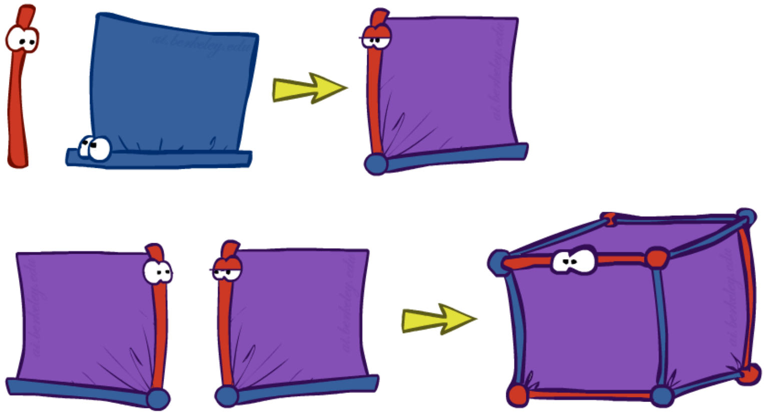
$P(R,T)$

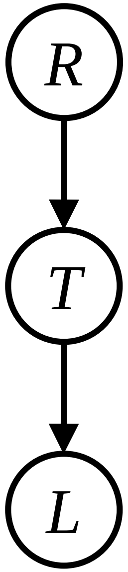
+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81



- Computation for each entry: pointwise products $\forall r, t: \quad P(r,t) = P(r) \cdot P(t|r)$

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

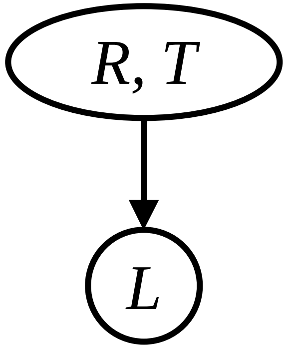
Join R

$P(R, T)$

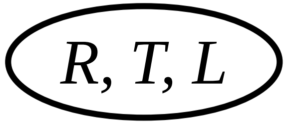
+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

$P(L|T)$

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



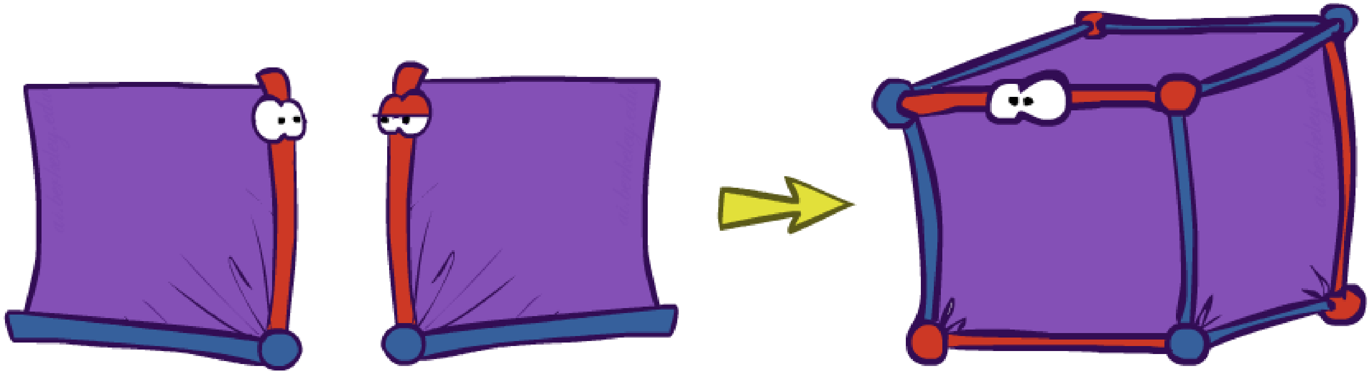
Join T



$P(R, T, L)$

+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

Example: Joining two conditional factors



- Example: $P(J|A)\ \text{x}\ P(M|A) = P(J,M|A)$

$P(J|A)$

A \ J	true	false
true	0.99	0.01
false	0.145	0.855

x

$P(M|A)$

A \ M	true	false
true	0.97	0.03
false	0.019	0.891

=

$P(J,M|A)$

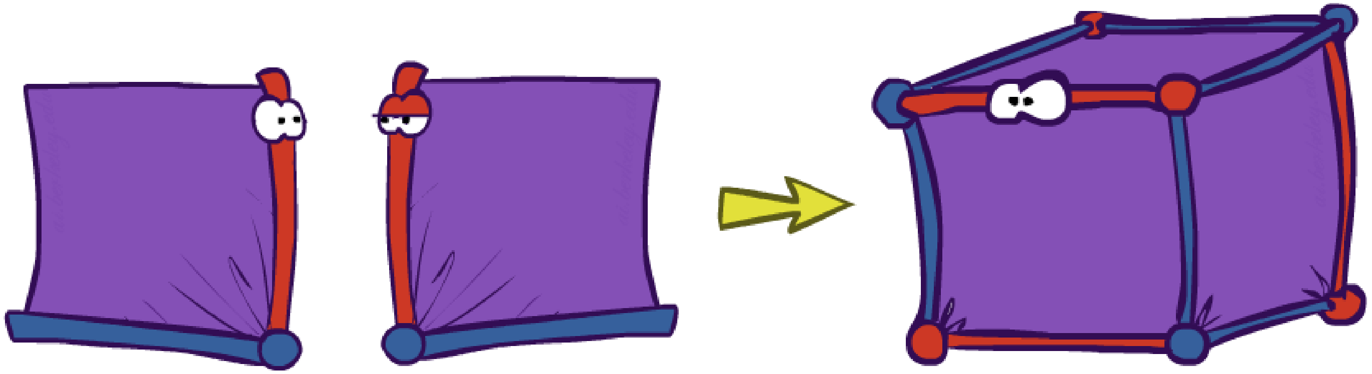
A=false

J \ M	true	false
		18

A=true

J \ M	true	false
true		
false		.0003

Example: Making larger factors



- Example: $f_1(U,V)$ x $f_2(V,W)$ x $f_3(W,X)$ = $f_4(U,V,W,X)$
- Sizes: [10,10] x [10,10] x [10,10] = [10,10,10,10]
- i.e., 300 numbers blows up to 10,000 numbers!
- Factor blowup can make joining very expensive

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:

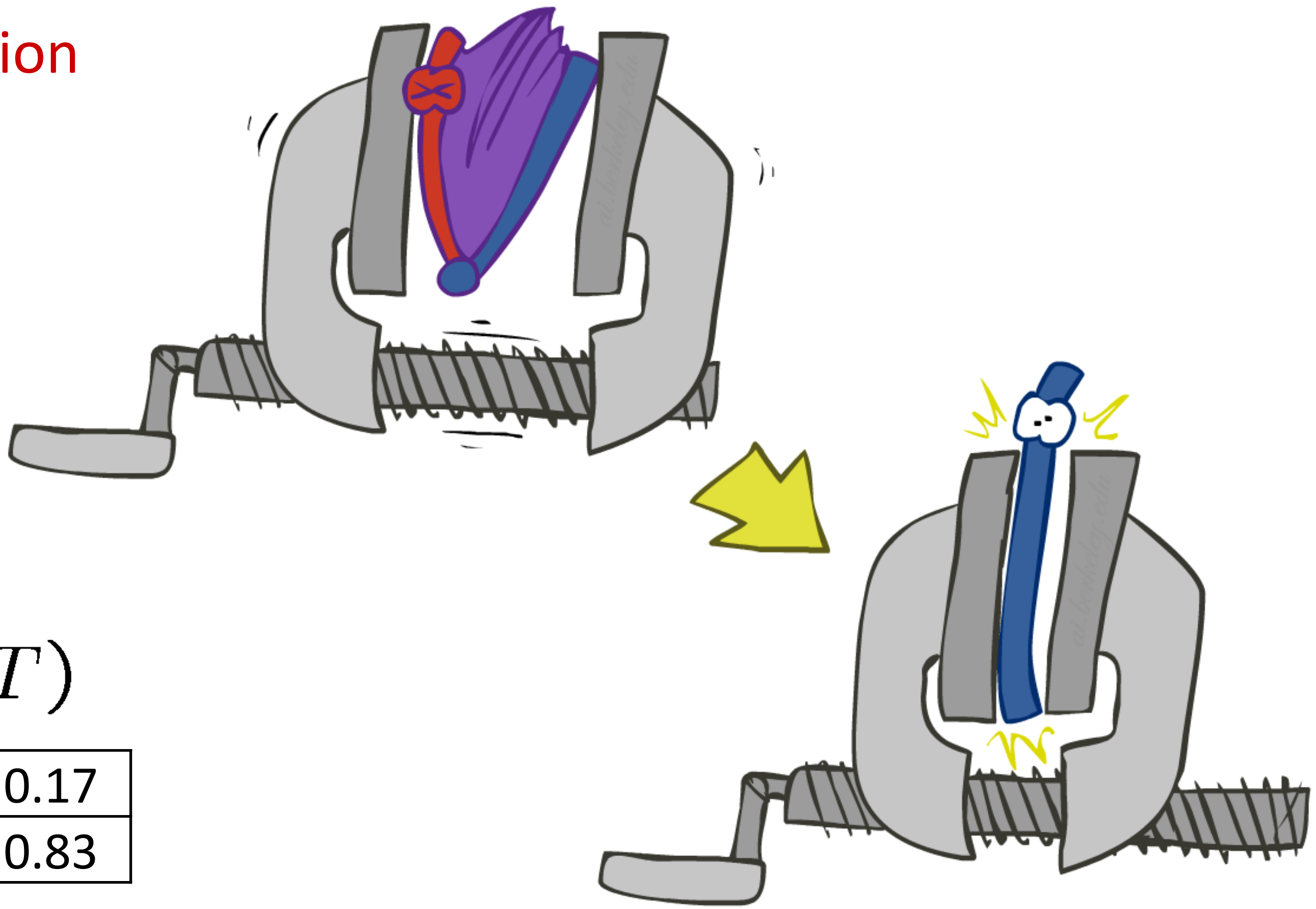
$P(R,T)$

+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

R, T, L

$P(R, T, L)$

+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

T, L

$P(T, L)$

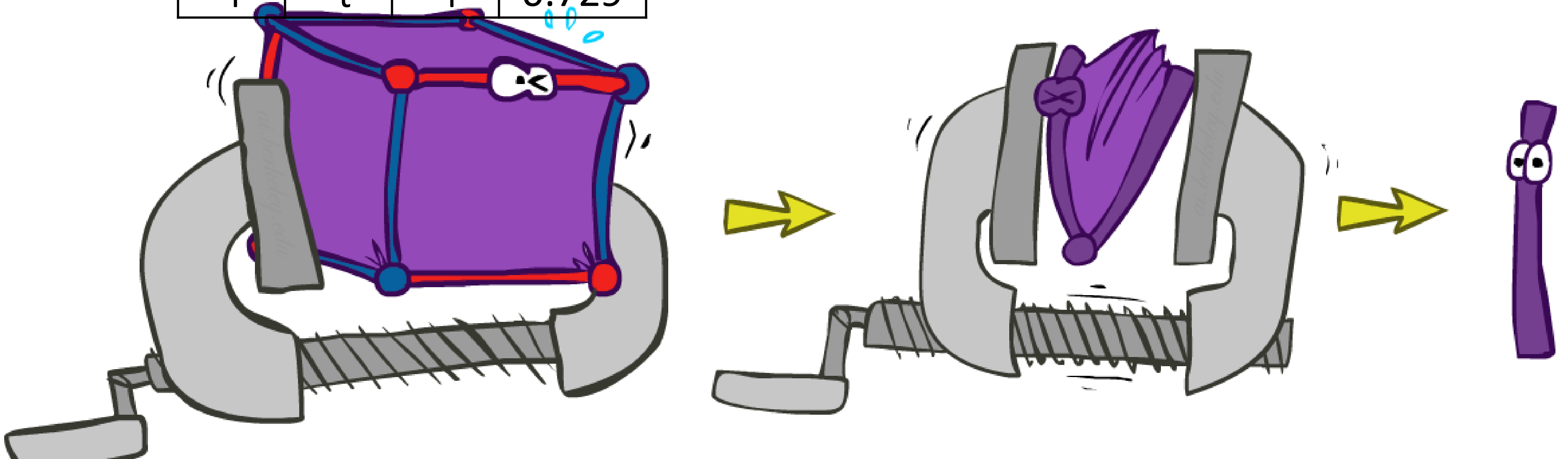
+t	+l	0.051
+t	-l	0.119
-t	+l	0.083
-t	-l	0.747

Sum out T

L

$P(L)$

+l	0.134
-l	0.866



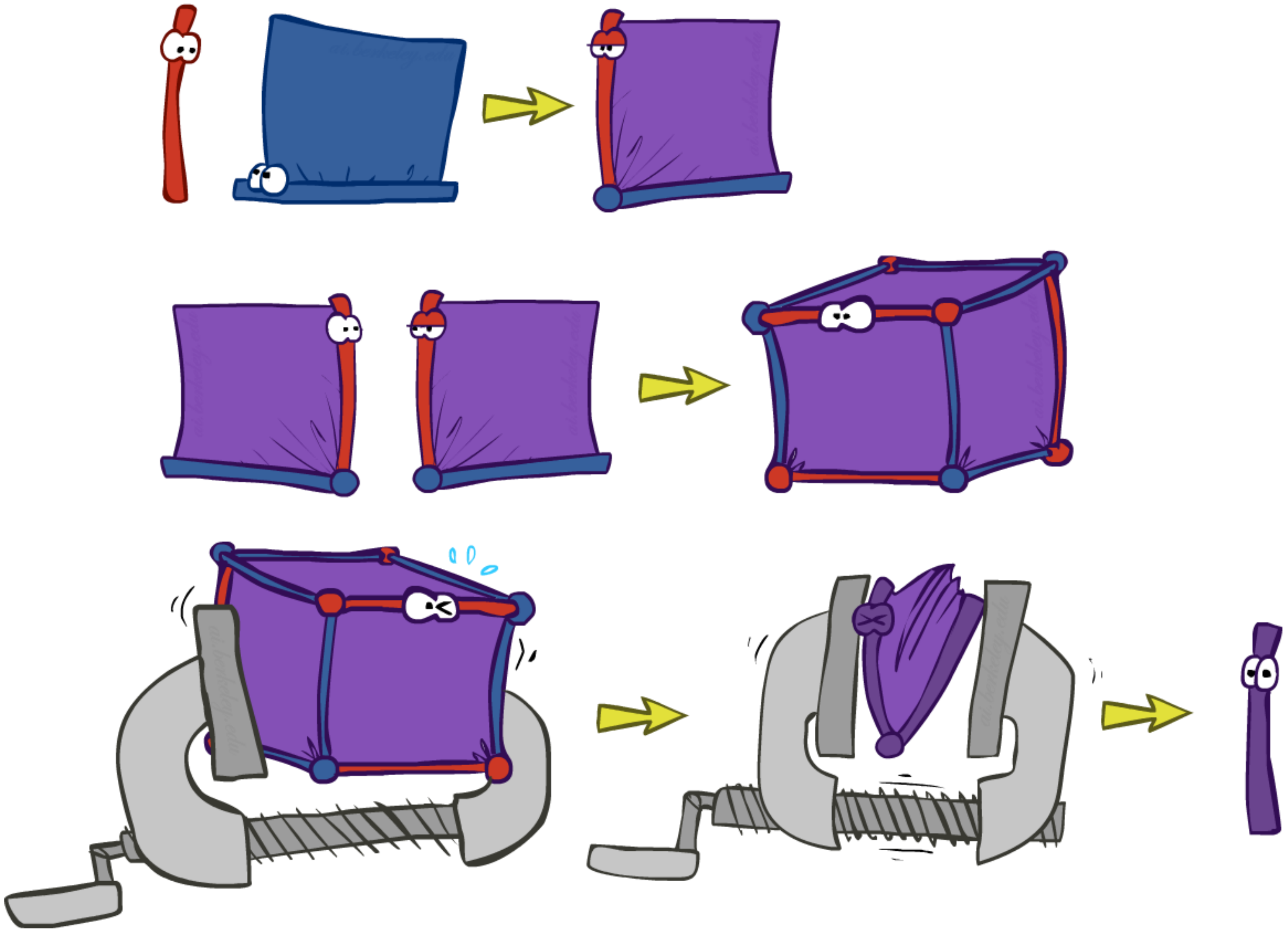
Thus Far: Multiple Join, Multiple Eliminate (= Inference by Enumeration)

$$P(R)$$

$$P(T|R) \quad \Longrightarrow \quad P(R,T,L) \quad \Longrightarrow \quad P(L)$$

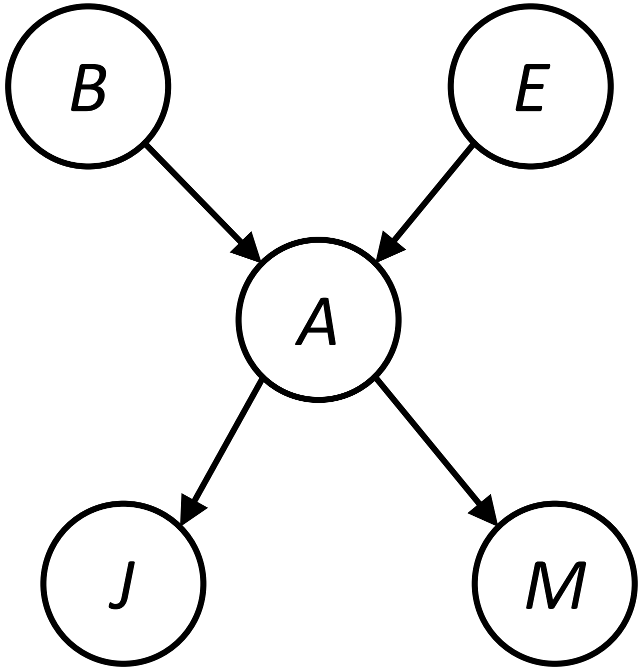
$$P(L|T)$$

Thus Far: Multiple Join, Multiple Eliminate (= Inference by Enumeration)



Inference by Enumeration in Bayes Net

- Reminder of inference by enumeration:
  - Any probability of interest can be computed by summing entries from the joint distribution
  - Entries from the joint distribution can be obtained from a BN by multiplying the corresponding conditional probabilities



$$
\begin{aligned}
P(B \mid j, m) &= \alpha\, P(B, j, m) \\
&= \alpha \textstyle\sum_{e,a} P(B, e, a, j, m) \\
&= \alpha \textstyle\sum_{e,a} P(B)\, P(e)\, P(\mathrm{a} \mid B, e)\, P(j \mid a)\, P(m \mid a)
\end{aligned}
$$

- So inference in Bayes nets means computing sums of products of numbers: sounds easy!!
- Problem: sums of *exponentially many* products!

Can we do better?

- Consider
  - $x_1y_1z_1 + x_1y_1z_2 + x_1y_2z_1 + x_1y_2z_2 + x_2y_1z_1 + x_2y_1z_2 + x_2y_2z_1 + x_2y_2z_2$
  - 16 multiplies, 7 adds
  - Lots of repeated subexpressions!
- Rewrite as
  - $(x_1 + x_2)(y_1 + y_2)(z_1 + z_2)$
  - 2 multiplies, 3 adds

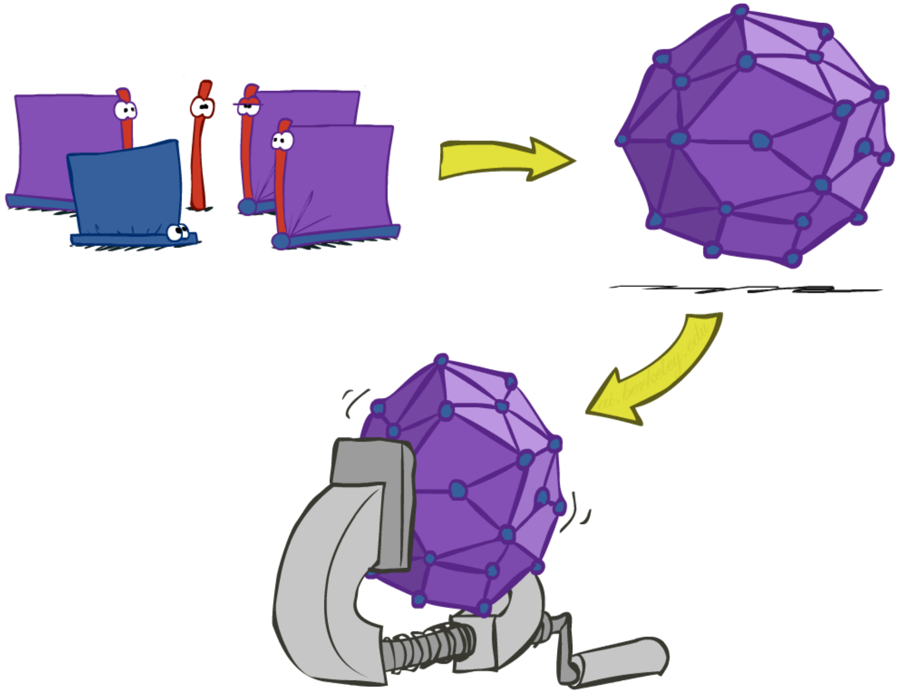
$$\begin{aligned}\sum_e \sum_a P(B)\,P(e)\,P(a \mid B,e)\,P(j \mid a)\,P(m \mid a) &= P(B)\,P(+e)\,P(+a \mid B,+e)\,P(j \mid +a)\,P(m \mid +a) \\ &+ P(B)\,P(-e)\,P(+a \mid B,-e)\,P(j \mid +a)\,P(m \mid +a) \\ &+ P(B)\,P(+e)\,P(-a \mid B,+e)\,P(j \mid -a)\,P(m \mid -a) \\ &+ P(B)\,P(-e)\,P(-a \mid B,-e)\,P(j \mid -a)\,P(m \mid -a)\end{aligned}$$

  - Lots of repeated subexpressions!

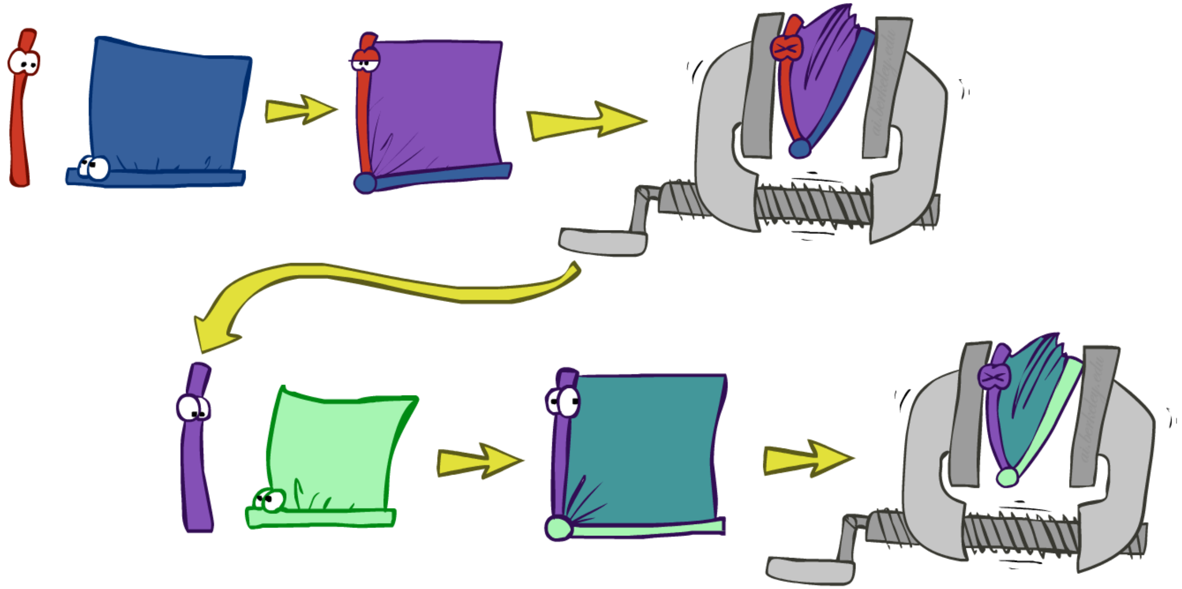
Variable Elimination

Inference by Enumeration vs. Variable Elimination

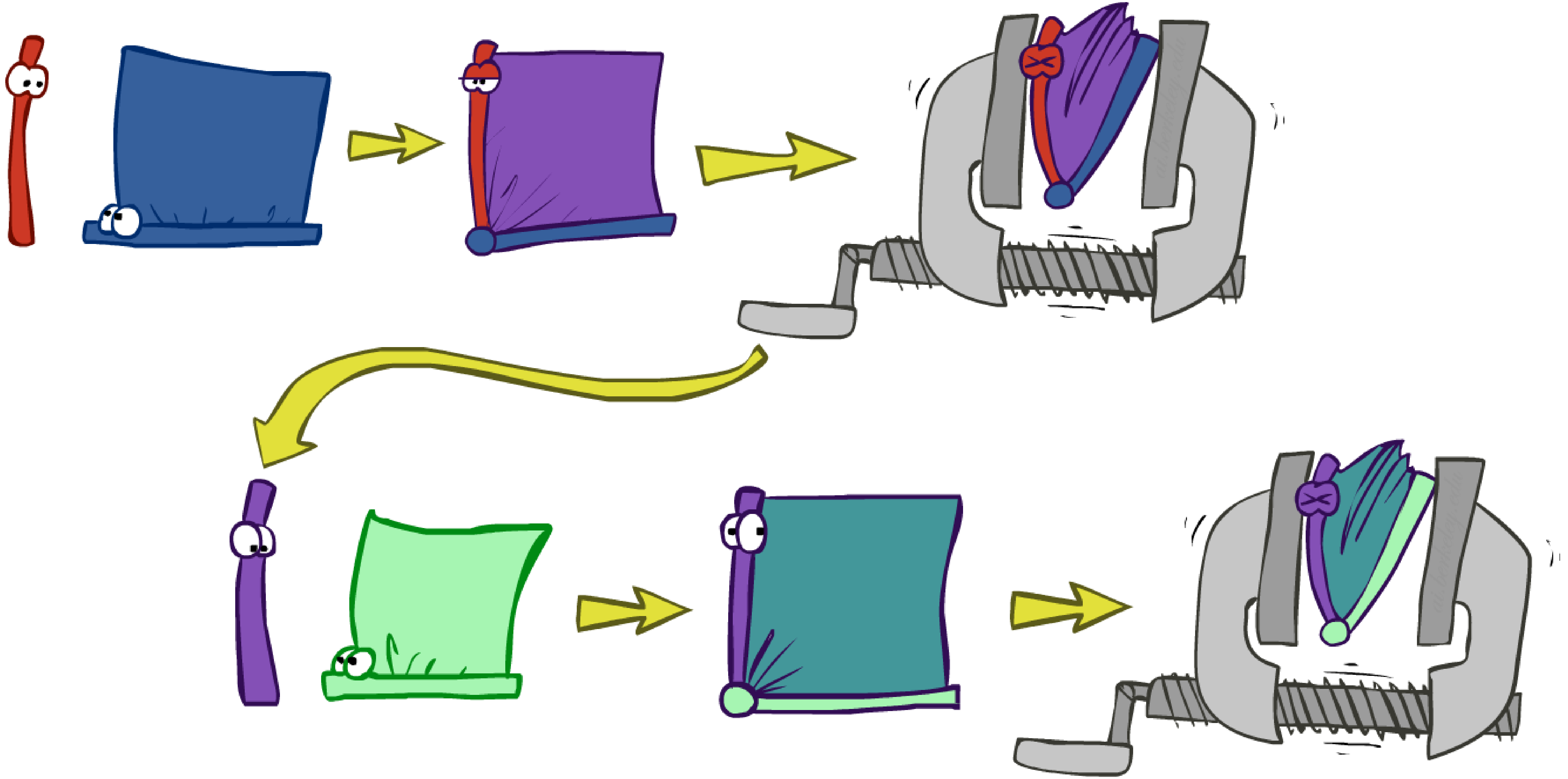
- Why is inference by enumeration so slow?
  - You join up the whole joint distribution before you sum out the hidden variables



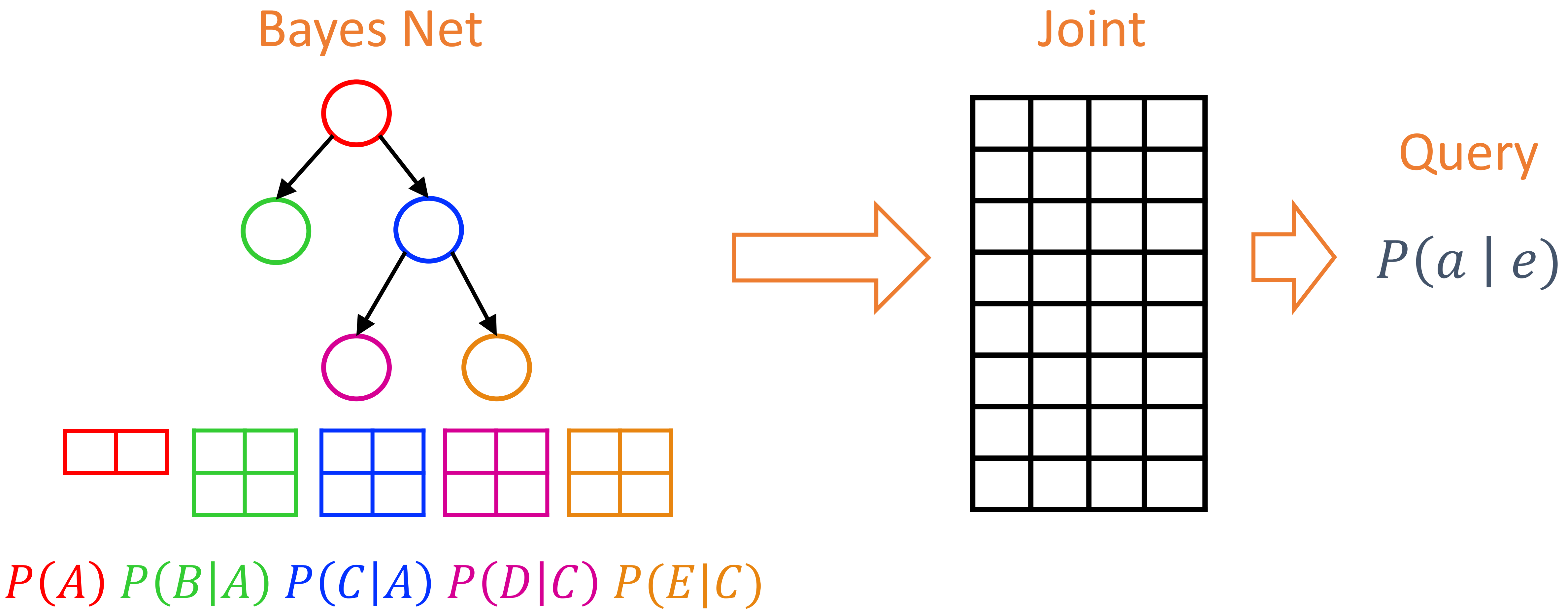
- Idea: interleave joining and marginalizing!
  - Called "Variable Elimination"
  - Still NP-hard, but usually much faster than inference by enumeration



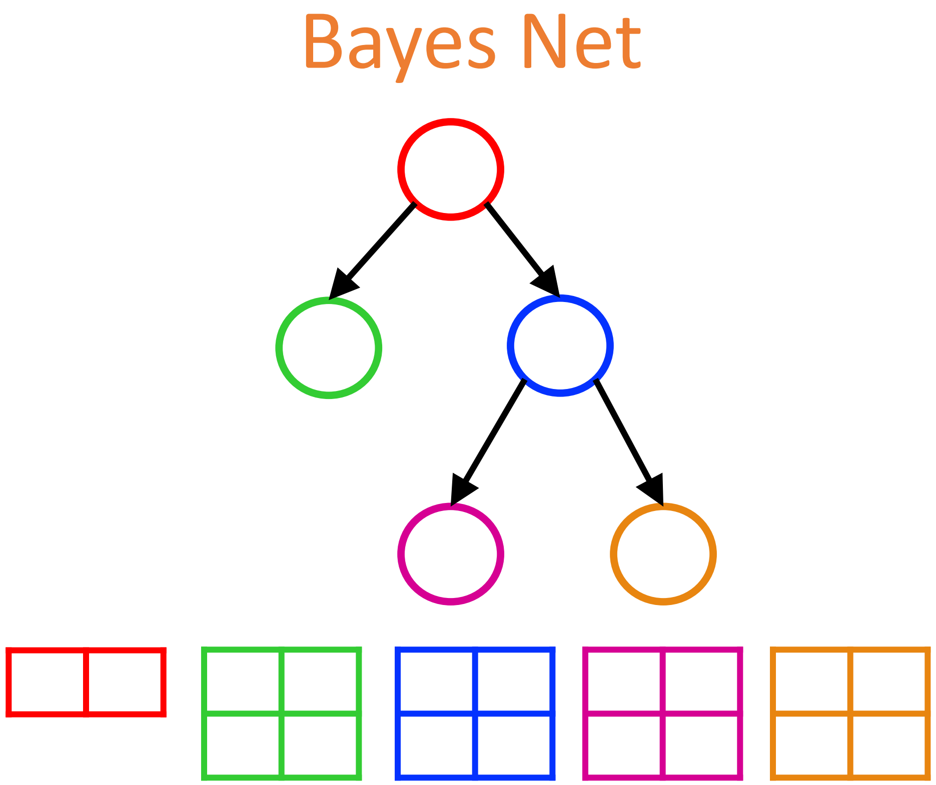
Marginalizing Early (= Variable Elimination)



Answer Any Query from Bayes Net (Previous)



Next: Answer Any Query from Bayes Net



$P(A)$ $P(B|A)$ $P(C|A)$ $P(D|C)$ $P(E|C)$

Query

$P(a \mid e)$

Inference by Enumeration

- General case:
  - Evidence variables: $E_1 \dots E_k = e_1 \dots e_k$
  - Query* variable: Q
  - Hidden variables: $H_1 \dots H_r$

$\left.\begin{matrix} E_1 \dots E_k = e_1 \dots e_k \\ Q \\ H_1 \dots H_r \end{matrix}\right\} X_1, X_2, \dots X_n$

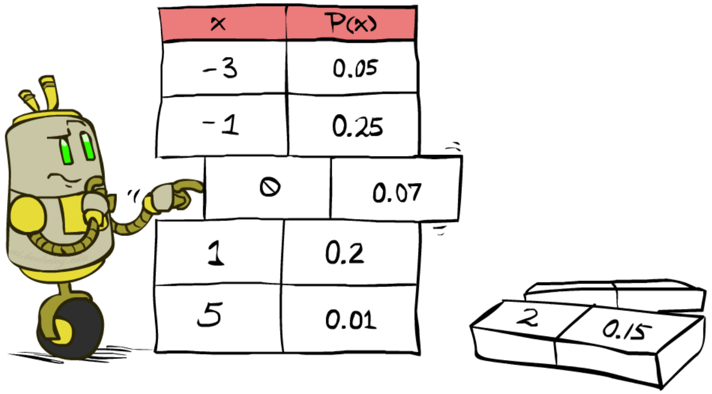
All variables

- We want:

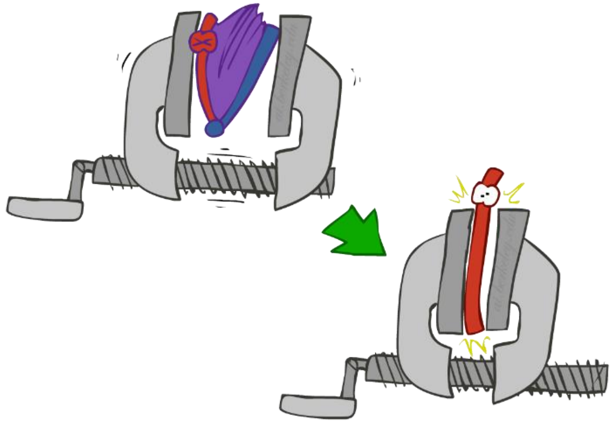
** Works fine with multiple query variables, too*

$$P(Q|e_1 \dots e_k)$$

- Step 1: Select the entries consistent with the evidence



- Step 2: Sum out H to get joint of Query and evidence



- Compute joint

$$P(Q, e_1 \dots e_k) = \sum_{h_1 \dots h_r} \underbrace{P(Q, h_1 \dots h_r, e_1 \dots e_k)}_{X_1, X_2, \dots X_n}$$

- Sum out hidden variables

- Step 3: Normalize

$$\times \frac{1}{Z}$$

$$Z = \sum_q P(Q, e_1 \cdots e_k)$$

$$P(Q|e_1 \cdots e_k) = \frac{1}{Z} P(Q, e_1 \cdots e_k)$$

Variable Elimination

- General case:
  - Evidence variables: $E_1 \ldots E_k = e_1 \ldots e_k$
  - Query* variable: Q
  - Hidden variables: $H_1 \ldots H_r$

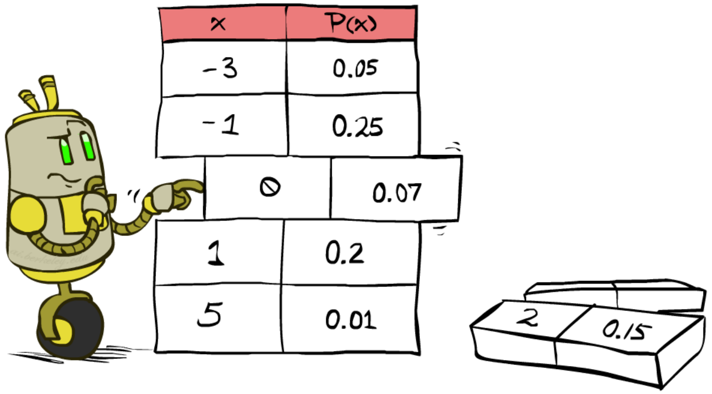
$$\left.\begin{array}{l} E_1 \ldots E_k = e_1 \ldots e_k \\ Q \\ H_1 \ldots H_r \end{array}\right\} \begin{array}{l} X_1, X_2, \ldots X_n \\ \textit{All variables} \end{array}$$

- We want:

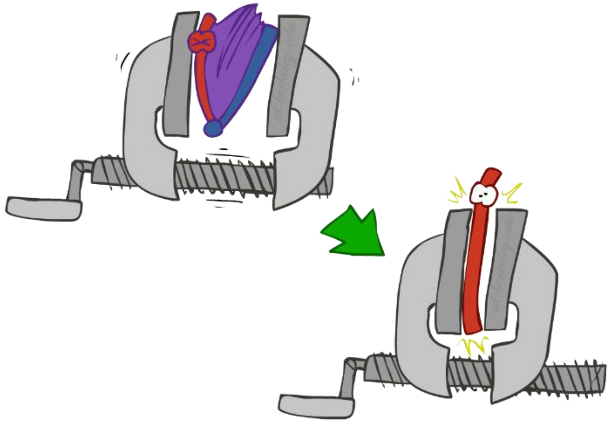
** Works fine with multiple query variables, too*

$$P(Q|e_1 \ldots e_k)$$

- Step 1: Select the entries consistent with the evidence



- Step 2: Sum out H to get joint of Query and evidence



$$P(Q, e_1 \ldots e_k) = \sum_{h_1 \ldots h_r} \underbrace{P(Q, h_1 \ldots h_r, e_1 \ldots e_k)}_{X_1, X_2, \ldots X_n}$$

- Interleave joining and summing out $X_1, X_2, \ldots X_n$

- Step 3: Normalize

$$\times \frac{1}{Z}$$

$$Z = \sum_q P(Q, e_1 \cdots e_k)$$

$$P(Q|e_1 \cdots e_k) = \frac{1}{Z} P(Q, e_1 \cdots e_k)$$

Marginalizing Early! (aka VE)

$P(R)$

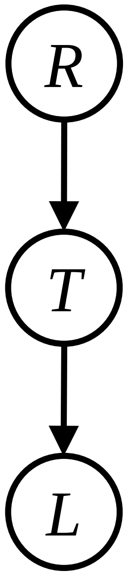
+r	0.1
-r	0.9

$P(T|R)$

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

$P(L|T)$

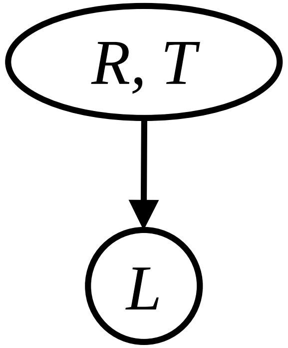
+t	+l	0.3
+t	-l	0.7
-t	+l	0.1
-t	-l	0.9



Join R

$P(R, T)$

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



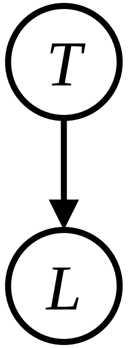
$P(L|T)$

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

Sum out R

$P(T)$

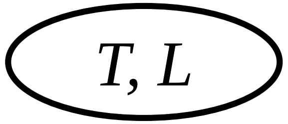
+t	0.17
-t	0.83



$P(L|T)$

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

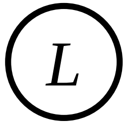
Join T



$P(T, L)$

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

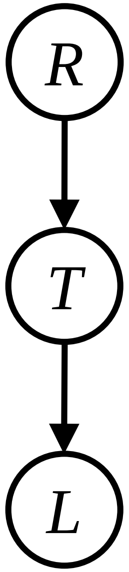
Sum out T



$P(L)$

+l	0.134
-l	0.866

Traffic Domain



$$P(L) = ?$$

- Inference by Enumeration

$$= \underbrace{\sum_t \underbrace{\sum_r \underbrace{P(L|t)\underbrace{P(r)P(t|r)}_{\text{Join on r}}}_{\text{Join on t}}}_{\text{Eliminate r}}}_{\text{Eliminate t}}$$

- Variable Elimination

$$= \underbrace{\sum_t \underbrace{P(L|t)\underbrace{\sum_r \underbrace{P(r)P(t|r)}_{\text{Join on r}}}_{\text{Eliminate r}}}_{\text{Join on t}}}_{\text{Eliminate t}}$$

Inference Overview

- Given random variables Q, H, E (query, hidden, evidence)

Enumeration

- We know how to do inference on a joint distribution

$$P(q|e) = \alpha\, P(q,e)$$
$$= \alpha \textstyle\sum_{h \in \{h_1, h_2\}} P(q,h,e)$$

- We know Bayes nets can break down joint in to CPT factors

$$H \rightarrow Q \rightarrow E$$

$$P(q|e) = \alpha \textstyle\sum_{h \in \{h_1, h_2\}} P(h)\, P(q|h)\, P(e|q)$$
$$= \alpha\, [P(h_1)\, P(q|h_1)\, P(e|q) + P(h_2)\, P(q|h_2)\, P(e|q)]$$

Variable Elimination

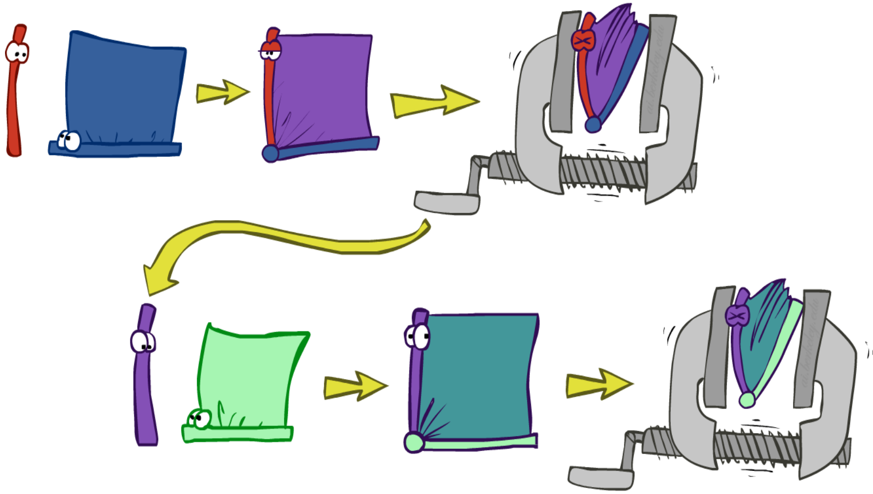
- But we can be more efficient

$$P(q|e) = \alpha\, P(e|q) \textstyle\sum_{h \in \{h_1, h_2\}} P(h) P(q|h)$$
$$= \alpha\, P(e|q)\, [P(h_1) P(q|h_1) + P(h_2) P(q|h_2)]$$
$$= \alpha\, P(e|q)\, P(q)$$

- Now just extend to larger Bayes nets and a variety of queries

Variable Elimination: The basic ideas

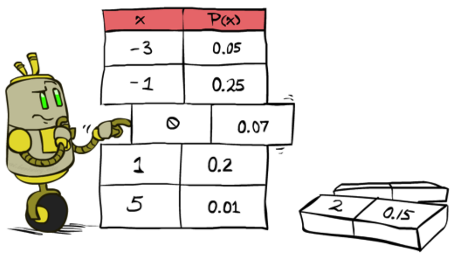
- Move summations inwards as far as possible
  - $P(B \mid j, m) = \alpha \sum_e \sum_a P(B)\, P(e)\, P(a|B,e)\, P(j|a)\, P(m|a)$
    $= \alpha\, P(B) \sum_e P(e) \sum_a P(a|B,e)\, P(j|a)\, P(m|a)$



- 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 factors

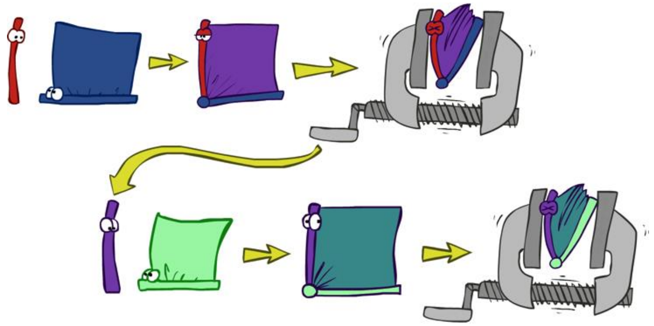
General Variable Elimination

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



- 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

$\times \frac{1}{Z}$

Variable Elimination

```
function VariableElimination(Q , e, bn) returns a distribution over Q
    factors ← [ ]
    for each var in ORDER(bn.vars) do
        factors ← [MAKE-FACTOR(var, e)|factors]
        if var is a hidden variable then
                factors ← SUM-OUT(var,factors)
    return NORMALIZE(POINTWISE-PRODUCT(factors))
```

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:

$P(+r)$

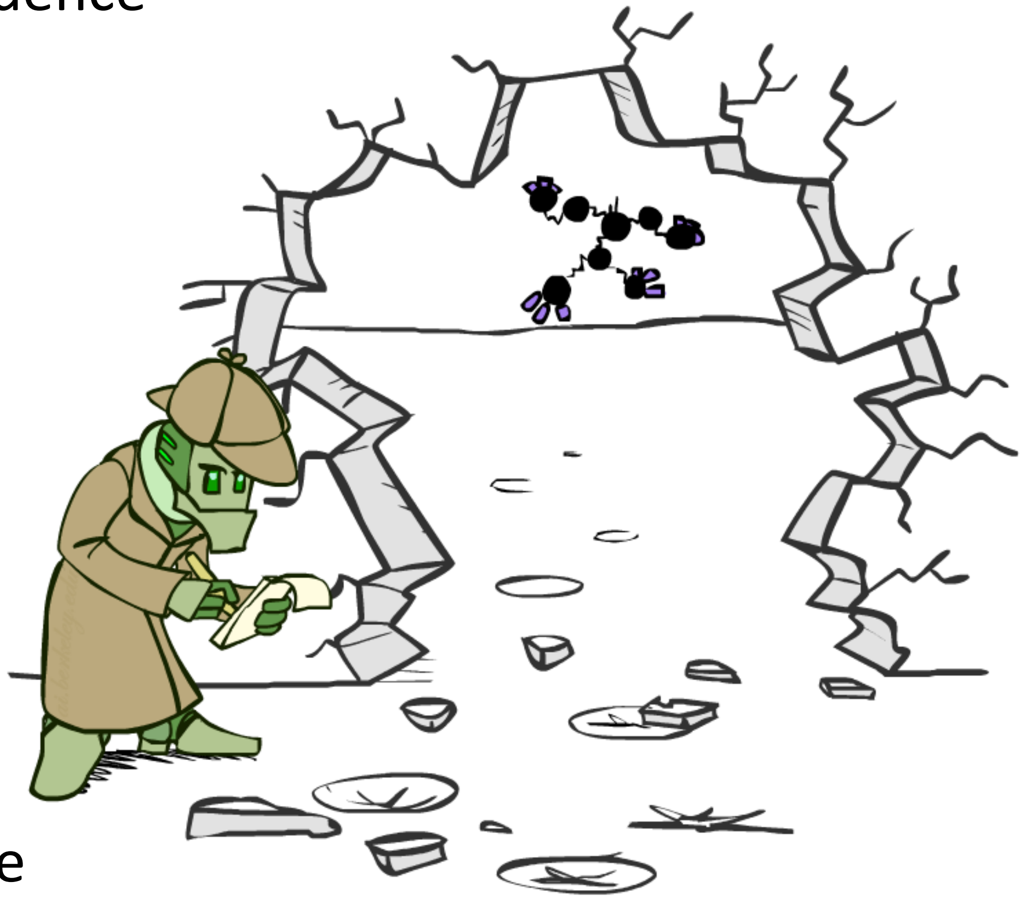
+r	0.1

$P(T|+r)$

+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 would end up with:

$$P(+r, L)$$

+r	+l	0.026
+r	-l	0.074

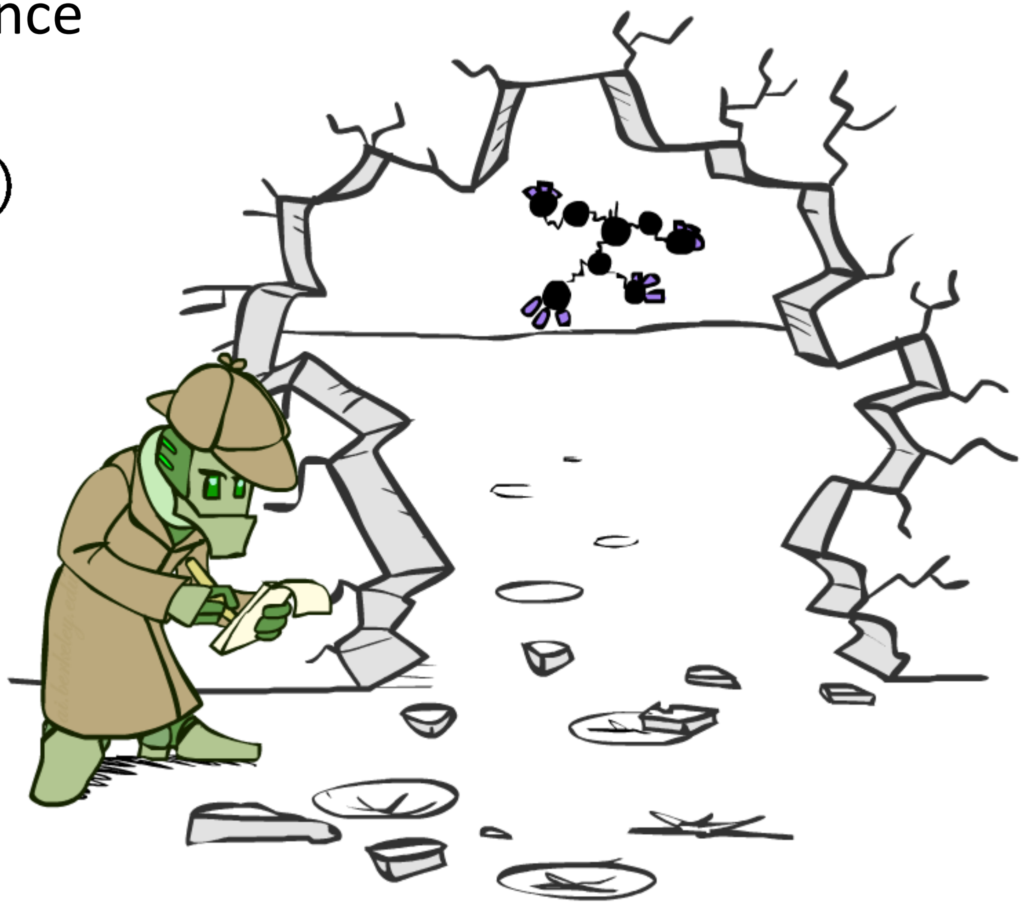
Normalize

$$P(L \mid +r)$$

+l	0.26
-l	0.74

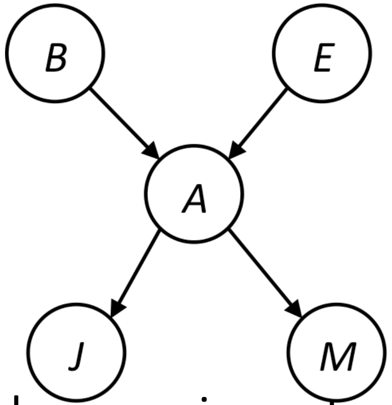
- To get our answer, just normalize this!

- That 's it!



Example

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



$$\boxed{P(B) \qquad P(E) \qquad P(A|B,E) \qquad P(j|A) \qquad P(m|A)}$$

$$
\begin{aligned}
P(B|j,m) &\propto P(B,j,m) \\
&= \sum_{e,a} P(B,j,m,e,a) \\
&= \sum_{e,a} P(B)P(e)P(a|B,e)P(j|a)P(m|a) \\
&= \sum_{e} P(B)P(e) \sum_{a} P(a|B,e)P(j|a)P(m|a) \\
&= \sum_{e} P(B)P(e) f_1(j,m|B,e) \\
&= P(B) \sum_{e} P(e) f_1(j,m|B,e) \\
&= P(B) f_2(j,m|B)
\end{aligned}
$$

marginal can be obtained from joint by summing out

use Bayes' net joint distribution expression

use x*(y+z) = xy + xz

joining on a, and then summing out gives f_1

use x*(y+z) = xy + xz

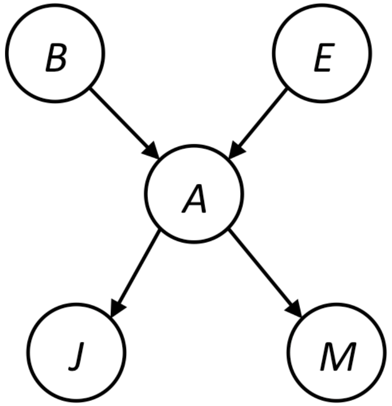
joining on e, and then summing out gives f_2

All we are doing is exploiting uwy + uwz + uxy + uxz + vwy + vwz + vxy +vxz = (u+v)(w+x)(y+z) to improve computational efficiency!

Example (cont'd)

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

$$\boxed{P(B) \qquad P(E) \qquad P(A|B,E) \qquad P(j|A) \qquad P(m|A)}$$



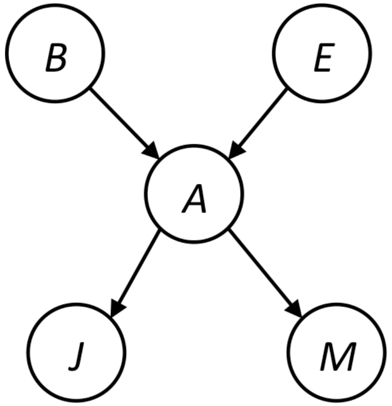
Choose A

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

$$\boxed{P(B) \qquad P(E) \qquad P(j,m|B,E)}$$

Example (cont'd)

$$\boxed{P(B) \qquad P(E) \qquad P(j, m|B, E)}$$



Choose E

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

$$\boxed{P(B) \qquad P(j, m|B)}$$

Finish with B

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

Another 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 $f_1(y_1|Z) = \sum_{x_1} P(x_1|Z)P(y_1|x_1)$, and we are left with:

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

Eliminate X_2, this introduces the factor $f_2(y_2|Z) = \sum_{x_2} P(x_2|Z)P(y_2|x_2)$, and we are left with:

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

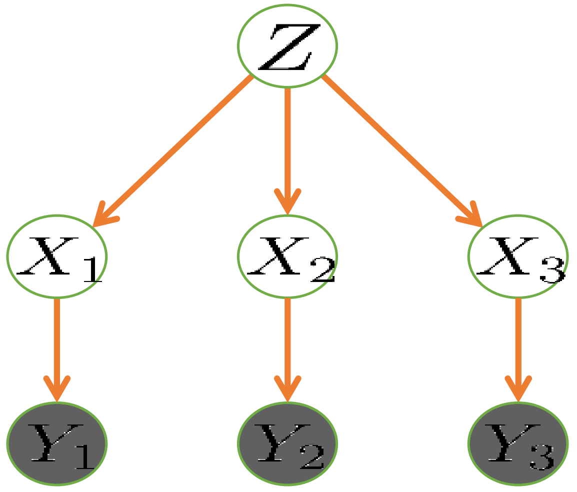
Eliminate Z, this introduces the factor $f_3(y_1, y_2, X_3) = \sum_z P(z)P(X_3|z)f_1(y_1|Z)f_2(y_2|Z)$, and we are left with:

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

No hidden variables left. Join the remaining factors to get:

$$f_4(y_1, y_2, y_3, X_3) = P(y_3|X_3)\ \ f_3(y_1, y_2, X_3)$$

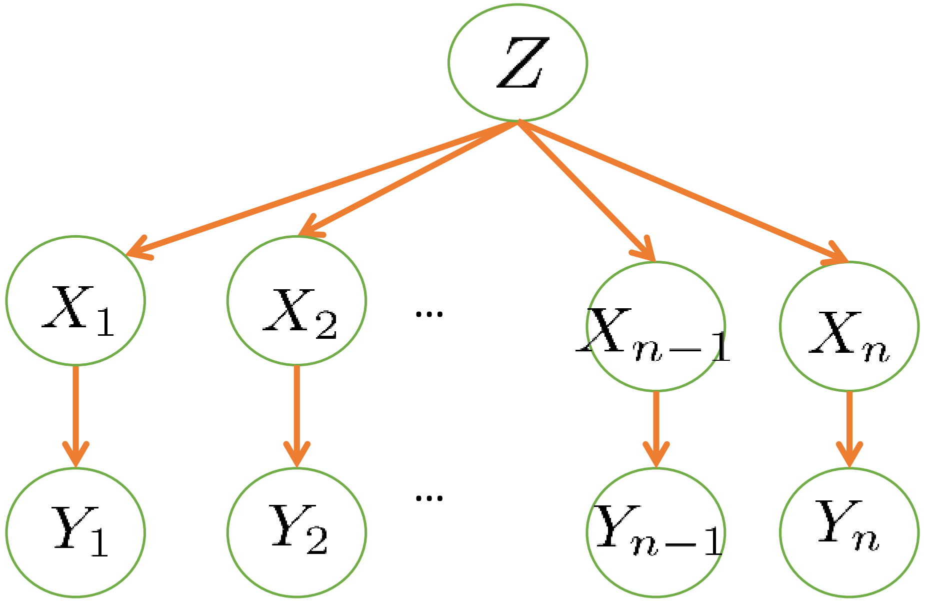
Normalizing over X_3 gives $P(X_3|y_1, y_2, y_3) = f_4(y_1, y_2, y_3, X_3)/\sum_{x_3} f_4(y_1, y_2, y_3, x_3)$



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 X_3 respectively).

Variable Elimination Ordering

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

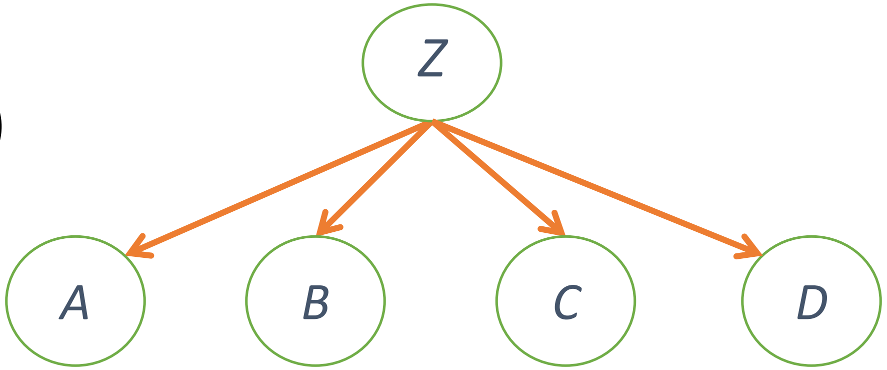


- Answer: 2^n versus 2 (assuming binary)

- In general: the ordering can greatly affect efficiency

Detail of size 4

- Elimination order: C, B, A, Z
  - $P(D) = \alpha \sum_{z,a,b,c} P(D|z)\, \mathrm{P}(z)\, P(a|z)\, P(b|z)\, P(c|z)$
  - $= \alpha \sum_z P(D|z)\, P(z) \sum_a P(a|z) \sum_b P(b|z) \sum_c P(c|z)$
  - Largest factor has 2 variables (D,Z)



- Elimination order: Z, C, B, A
  - $P(D) = \alpha \sum_{a,b,c,z} P(a|z)\, P(b|z)\, P(c|z)\, P(D|z)\, P(z)$
  - $= \alpha \sum_a \sum_b \sum_c \sum_z P(a|z)\, P(b|z)\, P(c|z)\, P(D|z)\, P(z)$
  - 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

- 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?

- CSP:

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

$$P(X_i = 0) = P(X_i = 1) = 0.5$$

$$Y_1 = X_1 \vee X_2 \vee \neg X_3$$

...

$$Y_8 = \neg X_5 \vee X_6 \vee X_7$$

$$Y_{1,2} = Y_1 \wedge Y_2$$

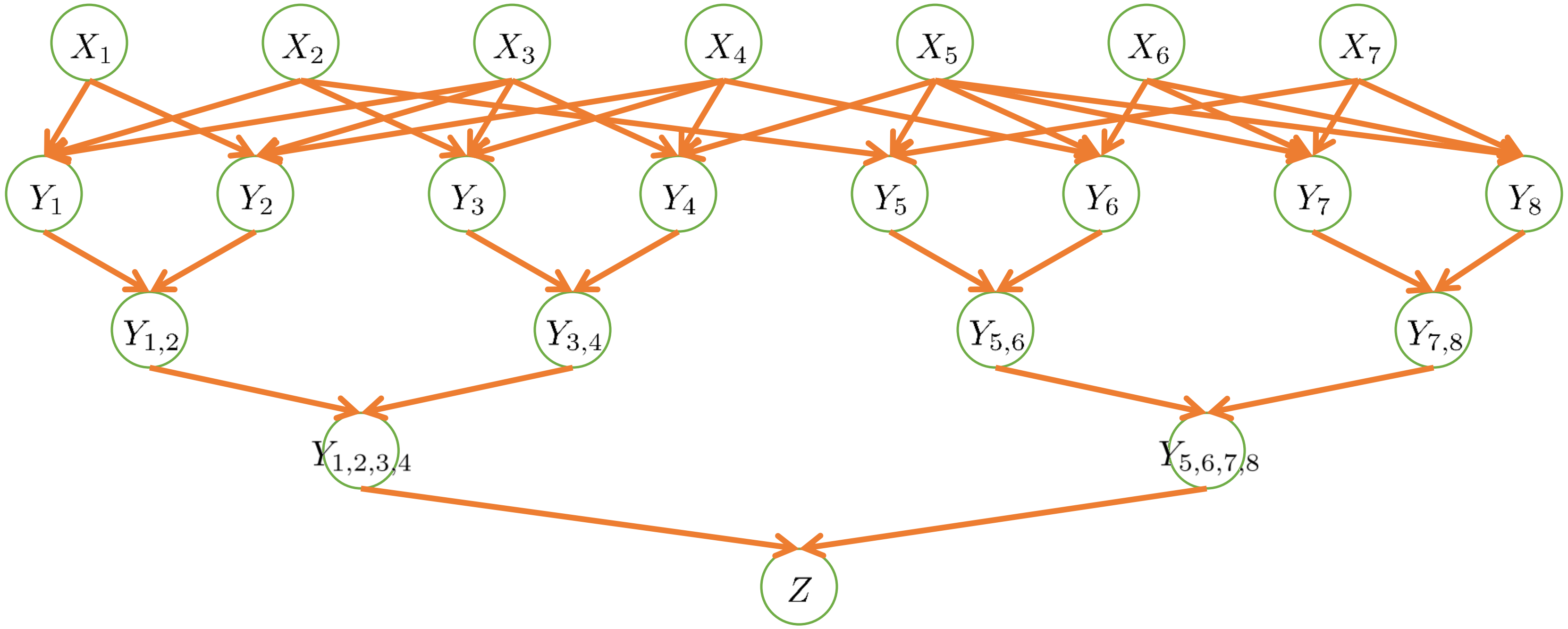
...

$$Y_{7,8} = Y_7 \wedge Y_8$$

$$Y_{1,2,3,4} = Y_{1,2} \wedge Y_{3,4}$$

$$Y_{5,6,7,8} = Y_{5,6} \wedge Y_{7,8}$$

$$Z = Y_{1,2,3,4} \wedge Y_{5,6,7,8}$$



- If we can answer P(z) equal to zero or not, we answered whether the 3-SAT problem has a solution
- Hence inference in Bayes' nets is NP-hard. No known efficient probabilistic inference in general

“Easy” Structures: Polytrees

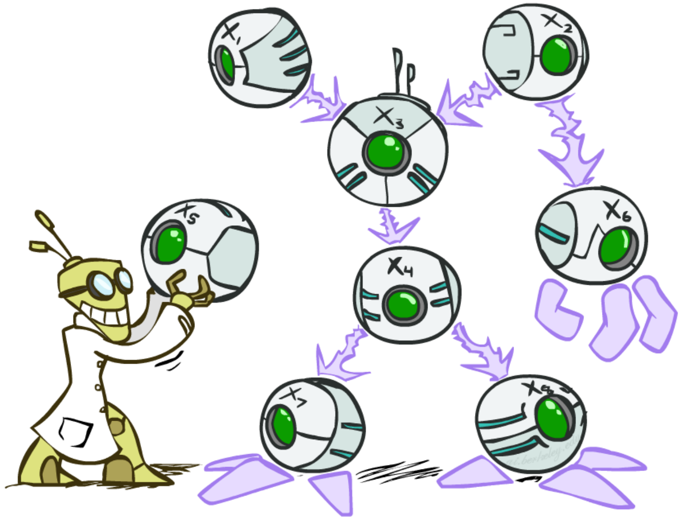
- A polytree is a directed graph with no undirected cycles

- 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
    - (Exercise) Think about how the specifics would work out!

Sampling

Recall: Bayes' Net Representation

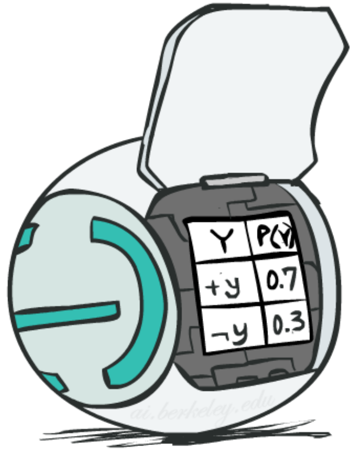


- A directed, acyclic graph, one node per random variable
- A conditional probability table (CPT) for each node
  - A collection of distributions over X, one for each combination of parents'' values

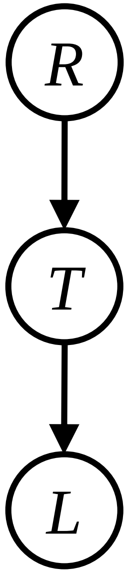
$$P(X|a_1 \dots a_n)$$

- Bayes' nets implicitly encode joint distributions
  - As a product of local conditional distributions
  - To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together:

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



Recap: Bayesian Inference (Exact)



$$P(L) = ?$$

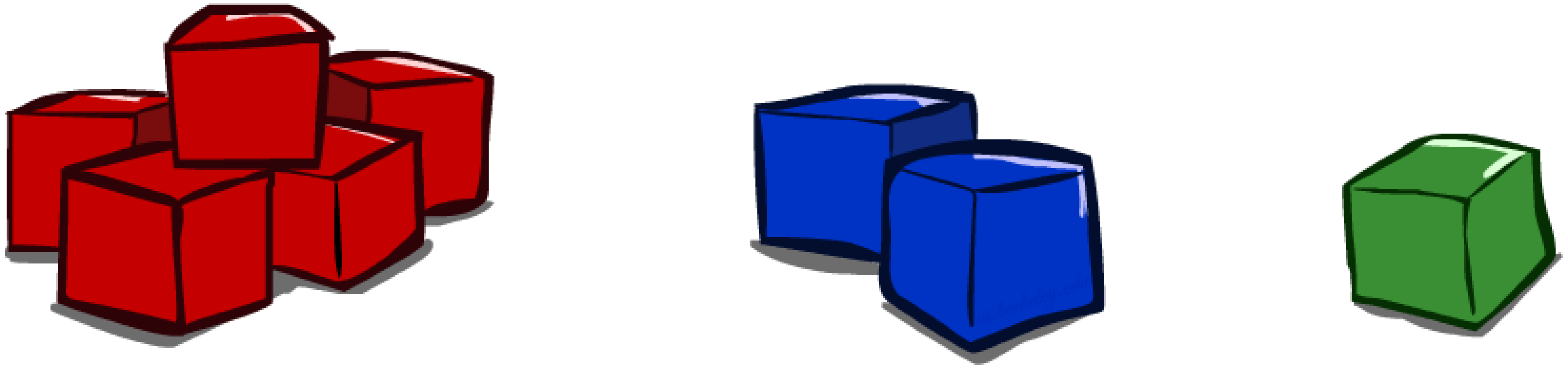
- Inference by Enumeration

$$= \underbrace{\sum_t \underbrace{\sum_r \underbrace{P(L|t)\underbrace{P(r)P(t|r)}_{\text{Join on r}}}_{\text{Join on t}}}_{\text{Eliminate r}}}_{\text{Eliminate t}}$$

- Variable Elimination

$$= \underbrace{\sum_t \underbrace{P(L|t) \underbrace{\sum_r \underbrace{P(r)P(t|r)}_{\text{Join on r}}}_{\text{Eliminate r}}}_{\text{Join on t}}}_{\text{Eliminate t}}$$

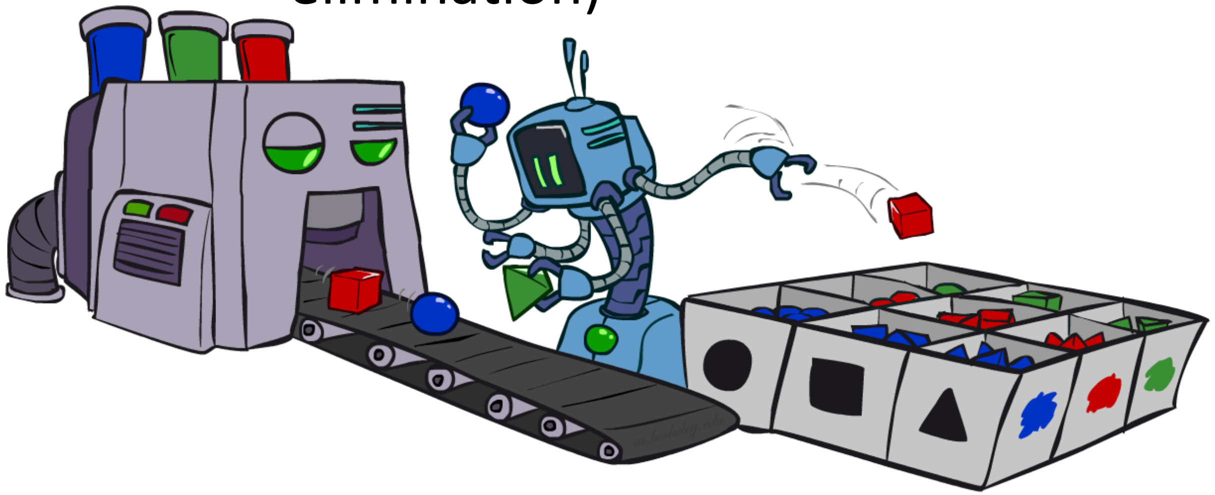
Approximate Inference: Sampling



Sampling

- Sampling is a lot like repeated simulation
  - 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 2

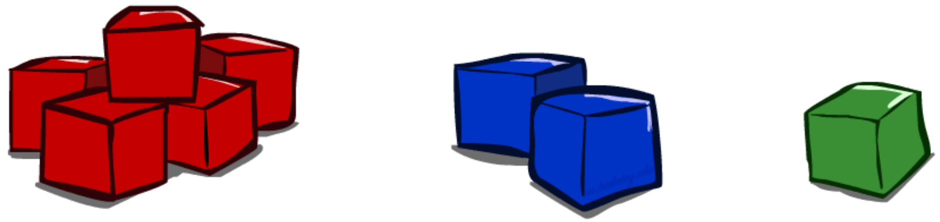
- Sampling from given distribution
  - Step 1: Get sample u from uniform distribution over [0, 1)
    - E.g. random() in python
  - Step 2: Convert this sample u into an outcome for the given distribution by having each target outcome associated with a sub-interval of [0,1) with sub-interval size equal to probability of the outcome

- Example

C	P(C)
red	0.6
green	0.1
blue	0.3

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

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



Sampling in Bayes’ Nets

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

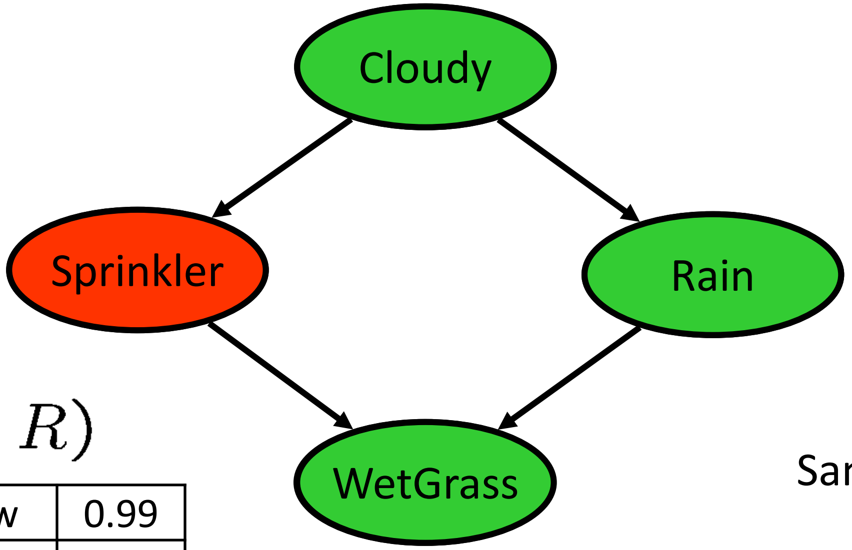
Prior Sampling: Example

$P(C)$

+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:

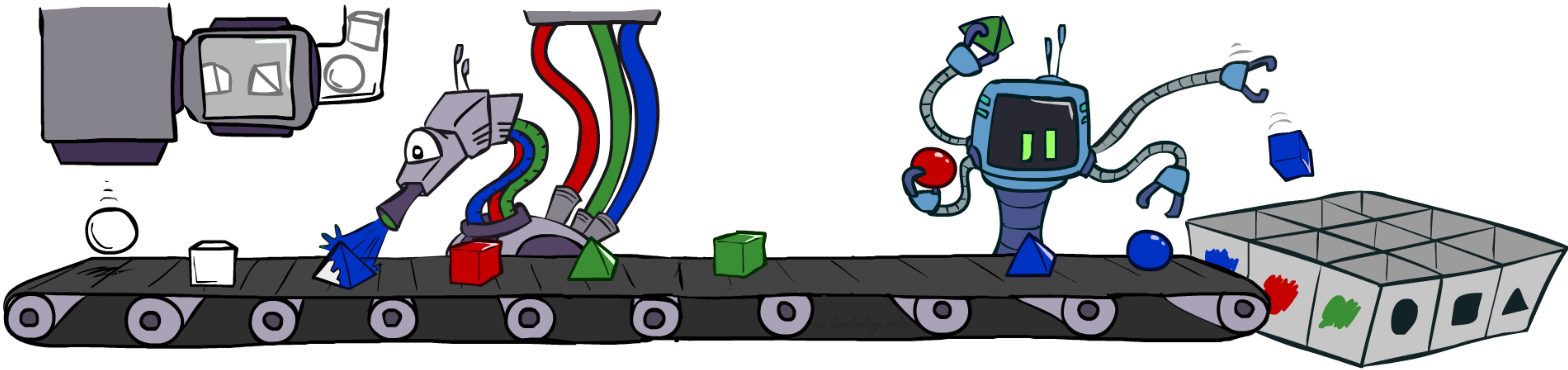
+c, -s, +r, +w

-c, +s, -r, +w

…

Prior Sampling: Algorithm

- For $i = 1, 2, \ldots, n$ 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:

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

- …i.e. the BN's joint probability

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

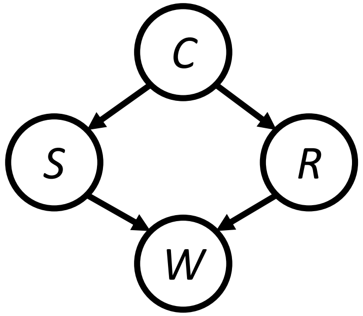
- Then

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

- i.e., the sampling procedure is consistent

Example

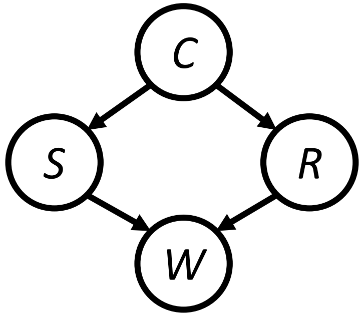
- We’ll get a bunch of samples from the BN:
  - +c, -s, +r, +w
  - +c, +s, +r, +w
  - -c, +s, +r, -w
  - +c, -s, +r, +w
  - -c, -s, -r, +w



- 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
    - P(C | +w)?   P(C | +r, +w)?
    - Can also use this to estimate expected value of f(X) - Monte Carlo Estimation
  - What about P(C | -r, -w)?

Rejection Sampling

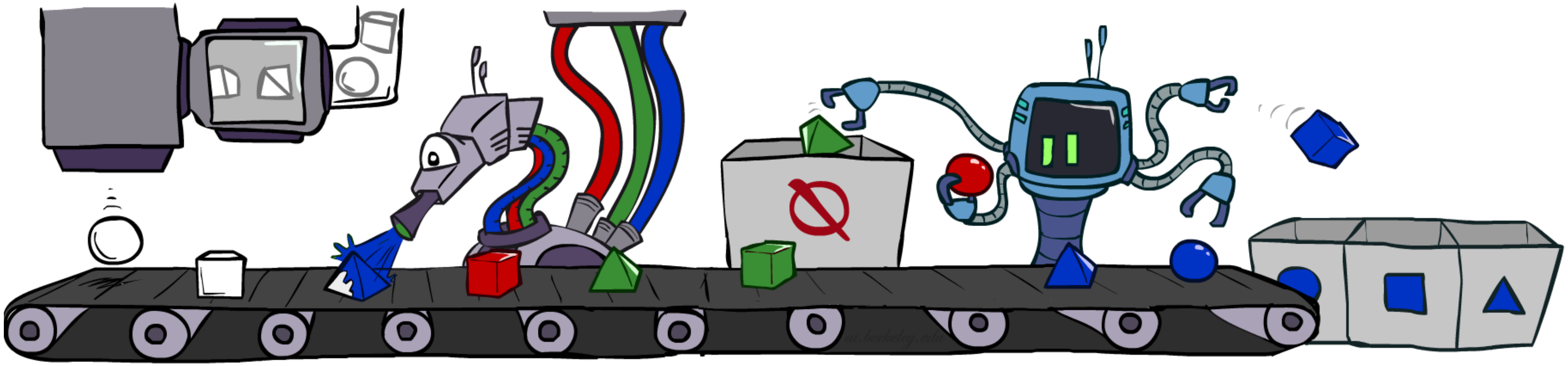
- Let's say we want P(C)
  - Just tally counts of C as we go



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

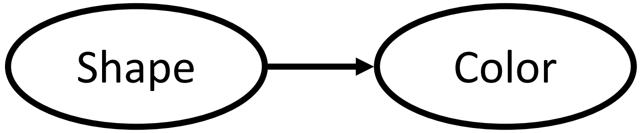
Rejection Sampling: Algorithm

- Input: evidence instantiation
- For i = 1, 2, …, n in topological order
  - Sample x_i from $P(X_i \mid Parents(X_i))$
  - If x_i not consistent with evidence
    - Reject: return – no sample is generated in this cycle
- Return $(x_1, x_2, \ldots, x_n)$

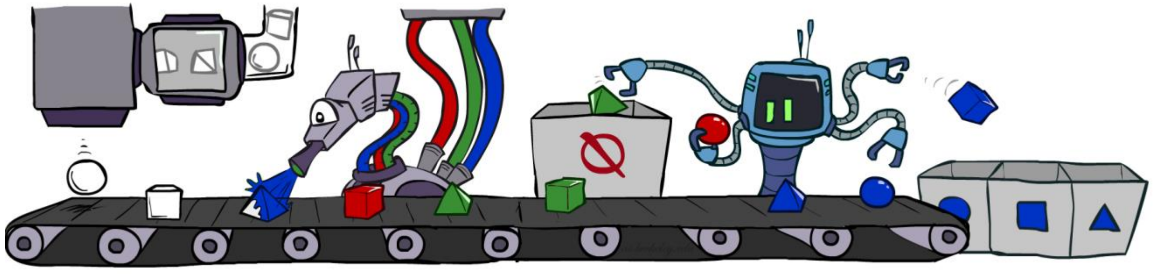


Likelihood Weighting

- Problem with rejection sampling:
  - If evidence is unlikely, rejects lots of samples
  - Consider P(Shape | blue)

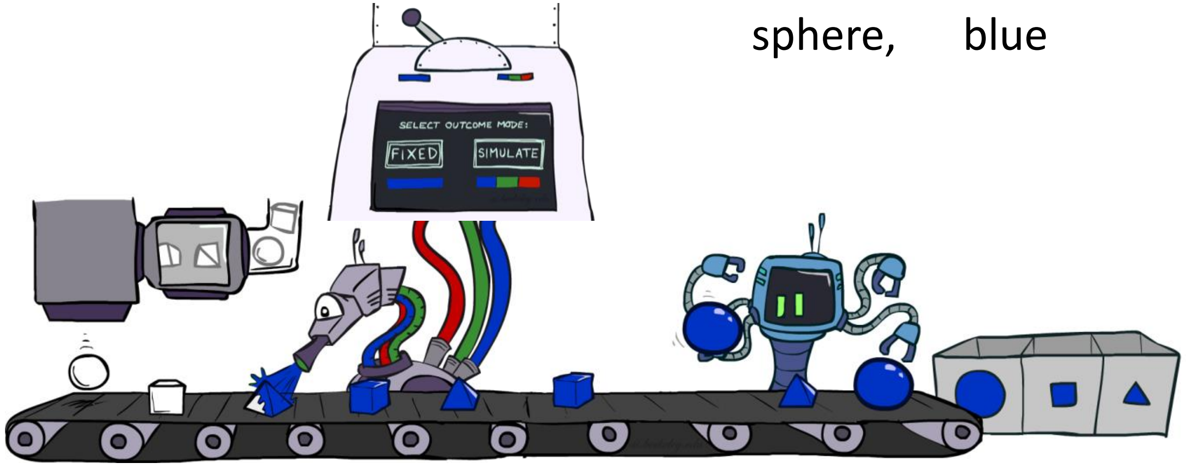


~~pyramid, green~~
~~pyramid, red~~
sphere, blue
~~cube, red~~
~~sphere, green~~



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

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



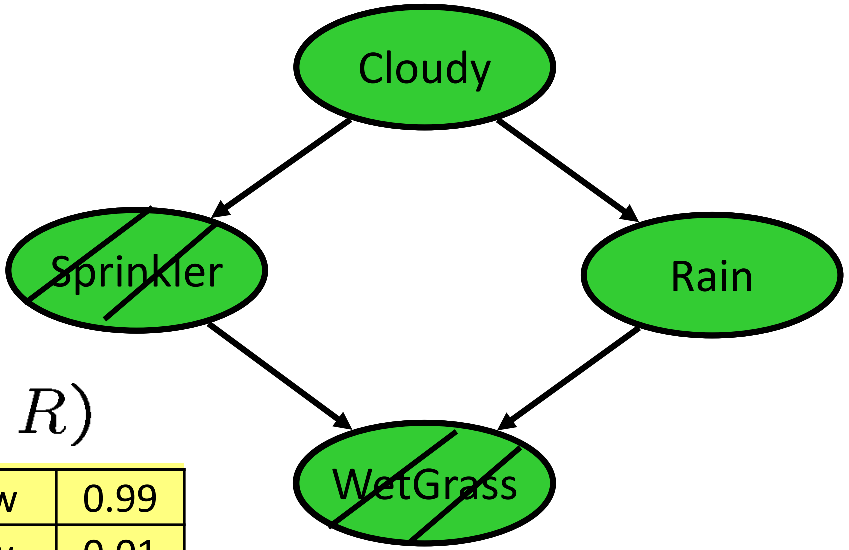
Likelihood Weighting: Example

$P(C)$

+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:

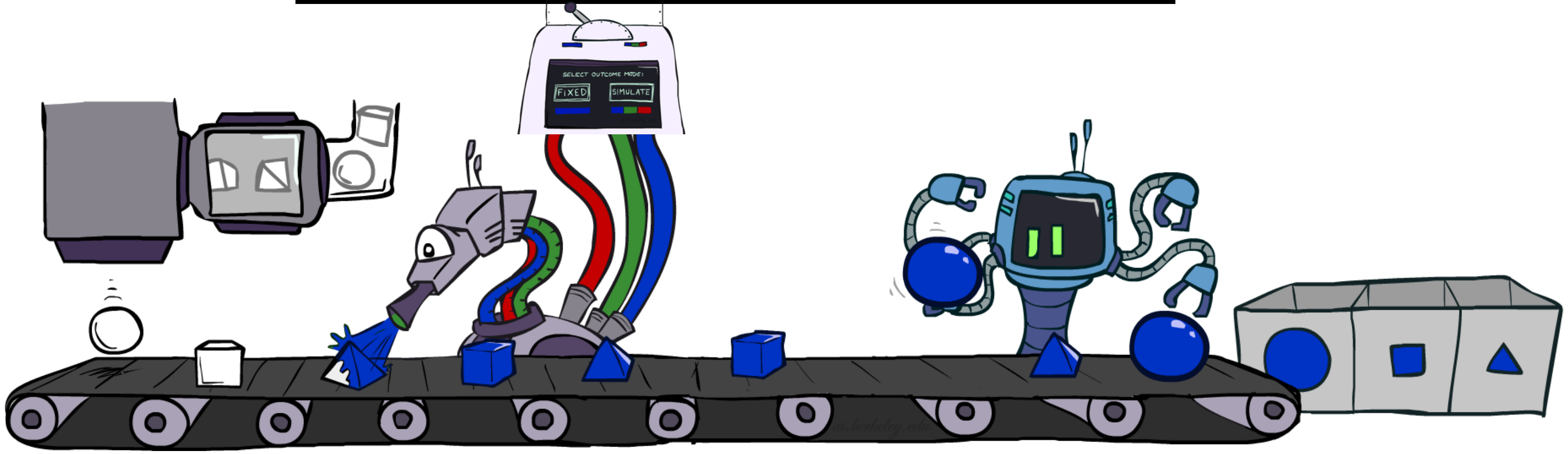
+c, +s, +r, +w   w = 1.0 x

-c, +s, -r, +w   w = 1.0 x 0.5 x 0.90

…

Likelihood Weighting: Algorithm

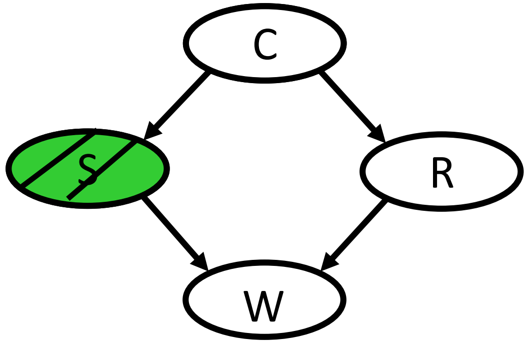
- Input: evidence instantiation
- $w = 1.0$
- for $i = 1, 2, \ldots, n$ in topological order
  - if X_i is an evidence variable
    - X_i = observation x_i for X_i
    - Set $w = w * P(x_i \mid \text{Parents}(X_i))$
  - else
    - Sample x_i from $P(X_i \mid \text{Parents}(X_i))$
- return $(x_1, x_2, \ldots, x_n)$, w



Likelihood Weighting

- Sampling distribution if z sampled and e fixed evidence

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



- Now, samples have weights

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

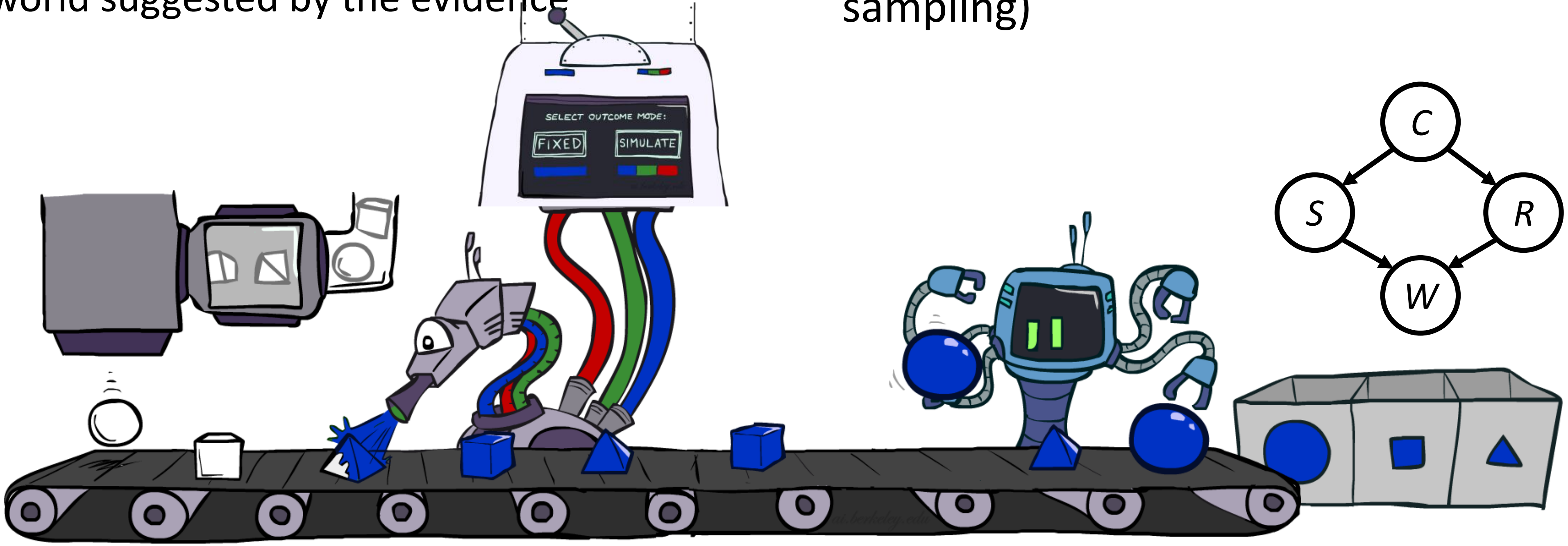
- Together, weighted sampling distribution is consistent

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

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

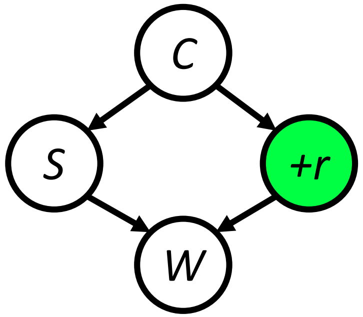
Likelihood Weighting

- Likelihood weighting is helpful
  - We have taken evidence into account as we generate the sample
  - E.g. here, W's value will get picked based on the evidence values of S, R
  - More of our samples will reflect the state of the world suggested by the evidence
- Likelihood weighting doesn't solve all our problems
  - Evidence influences the choice of downstream variables, but not upstream ones (C isn't more likely to get a value matching the evidence)
- We would like to consider evidence when we sample every variable (leads to Gibbs sampling)

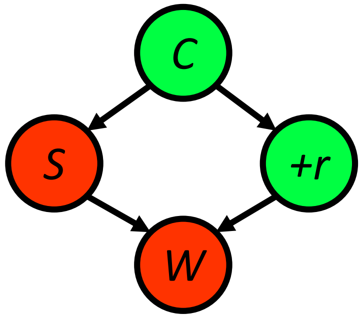


Gibbs Sampling: Example P(S | +r)

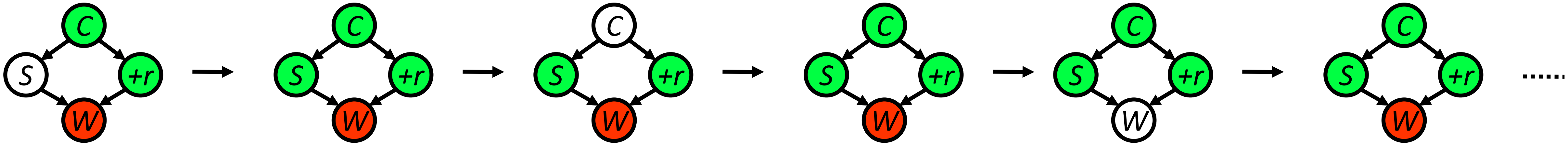
- Step 1: Fix evidence
  - R = +r



- Step 2: Initialize other variables
  - Randomly



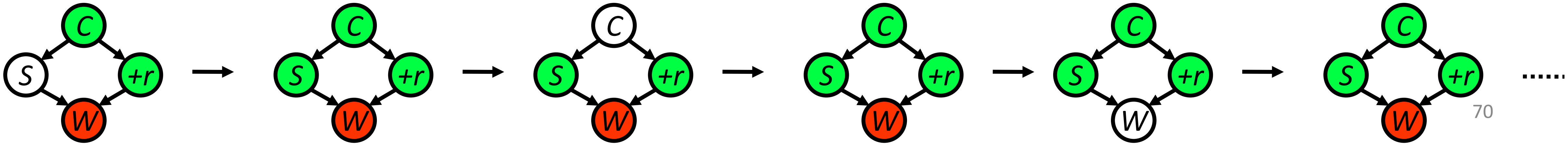
- Steps 3: Repeat
  - Choose a non-evidence variable X
  - Resample X from P(X | all other variables)*



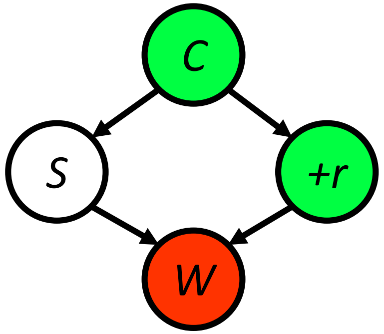
Sample from $P(S|+c,-w,+r)$ Sample from $P(C|+s,-w,+r)$ Sample from $P(W|+s,+c,+r)$

Gibbs Sampling

- Procedure
  - Keep track of a full instantiation $x_1, \dots, x_n$
  - Start with an arbitrary instantiation consistent with the evidence
  - Sample one variable at a time, conditioned on all the rest, but keep evidence fixed
  - Keep repeating this for a long time
- Property
  - In the limit of repeating this infinitely many times the resulting samples come from the correct distribution (i.e. conditioned on evidence)
- Rationale
  - Both upstream and downstream variables condition on evidence
- In contrast:
  - Likelihood weighting only conditions on upstream evidence, and hence weights obtained in likelihood weighting can sometimes be very small
  - Sum of weights over all samples is indicative of how many “effective” samples were obtained, so we want high weight



Resampling of One Variable



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

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

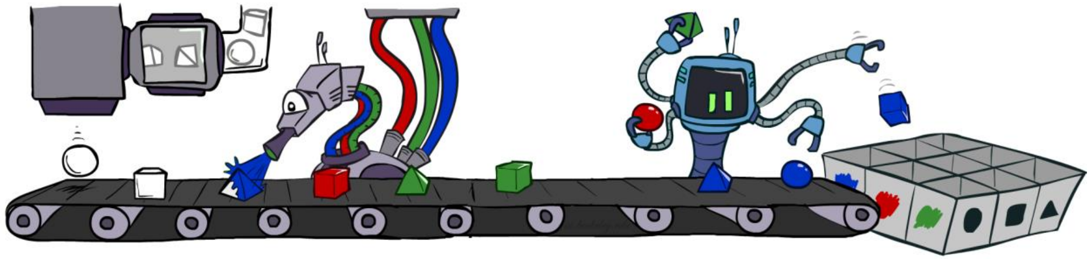
- Many things cancel out – only CPTs with S remain!
- More generally: only CPTs that have resampled variable need to be considered, and joined together

More Details on Gibbs Sampling*

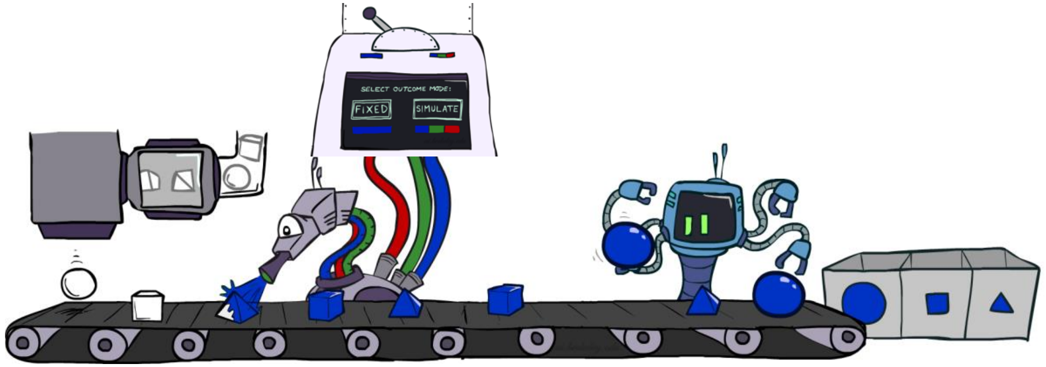
- Gibbs sampling belongs to a family of sampling methods called Markov chain Monte Carlo (MCMC)
  - Specifically, it is a special case of a subset of MCMC methods called Metropolis-Hastings
- You can read more about this here:
  - https://ermongroup.github.io/cs228-notes/inference/sampling/

Bayes' Net Sampling Summary

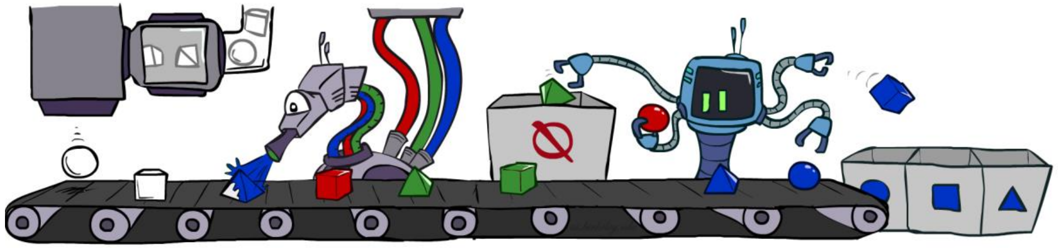
- Prior Sampling P(Q)



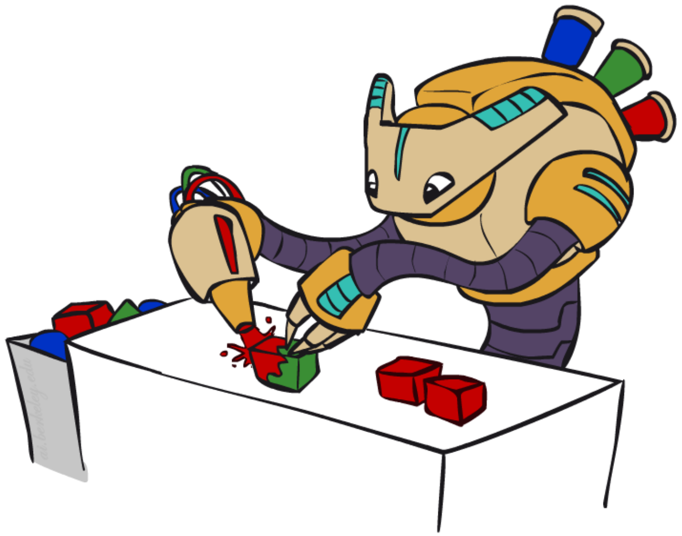
- Likelihood Weighting P(Q|e)



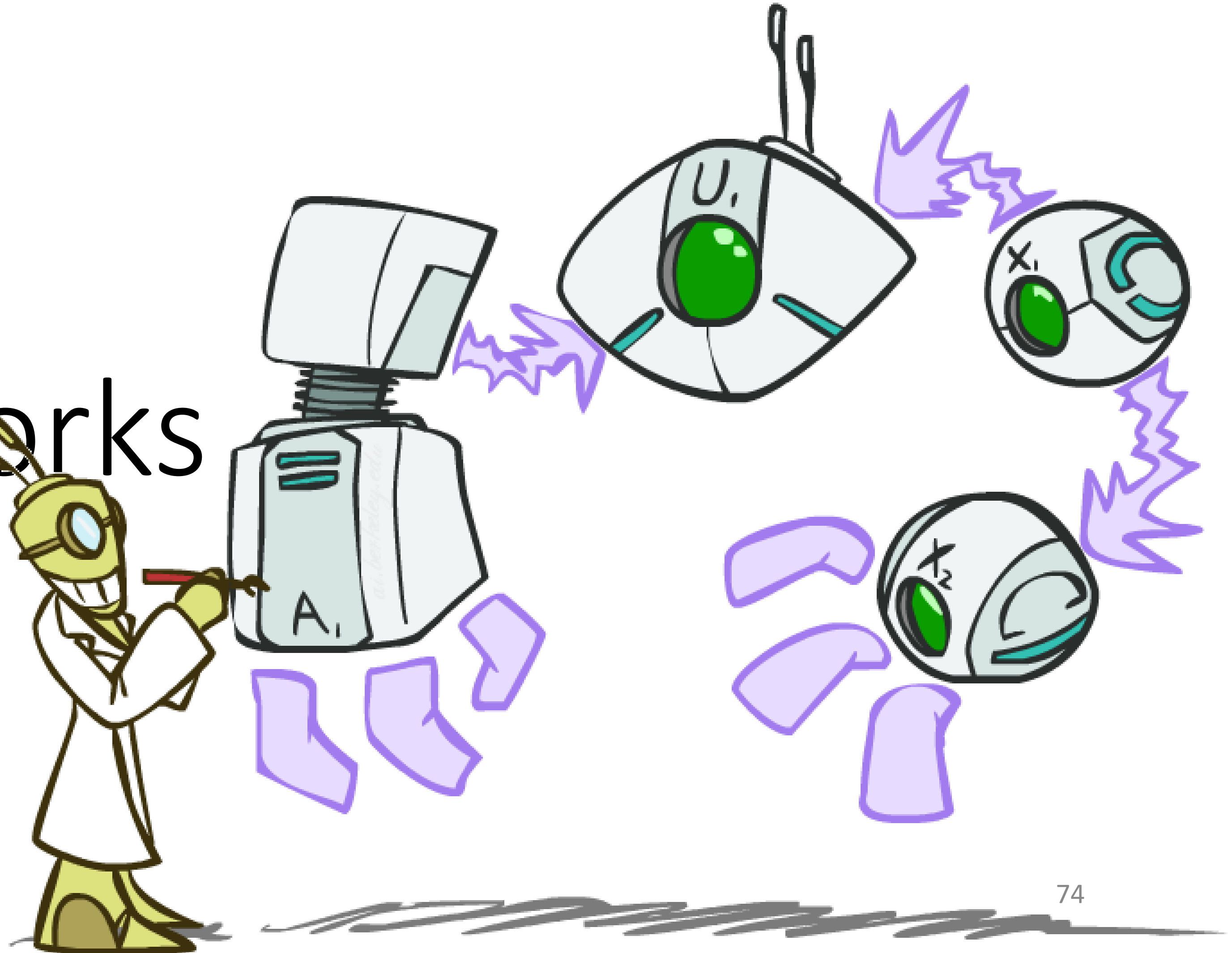
- Rejection Sampling P(Q|e)



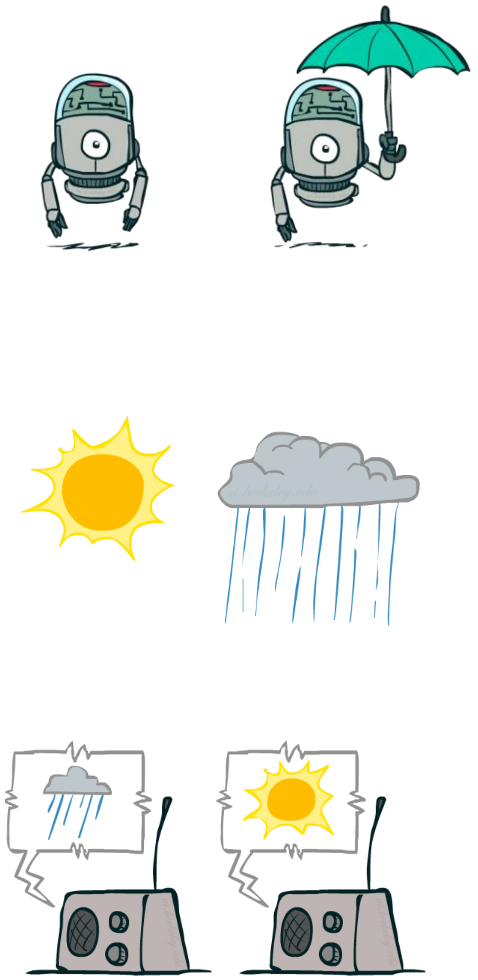
- Gibbs Sampling P(Q|e)

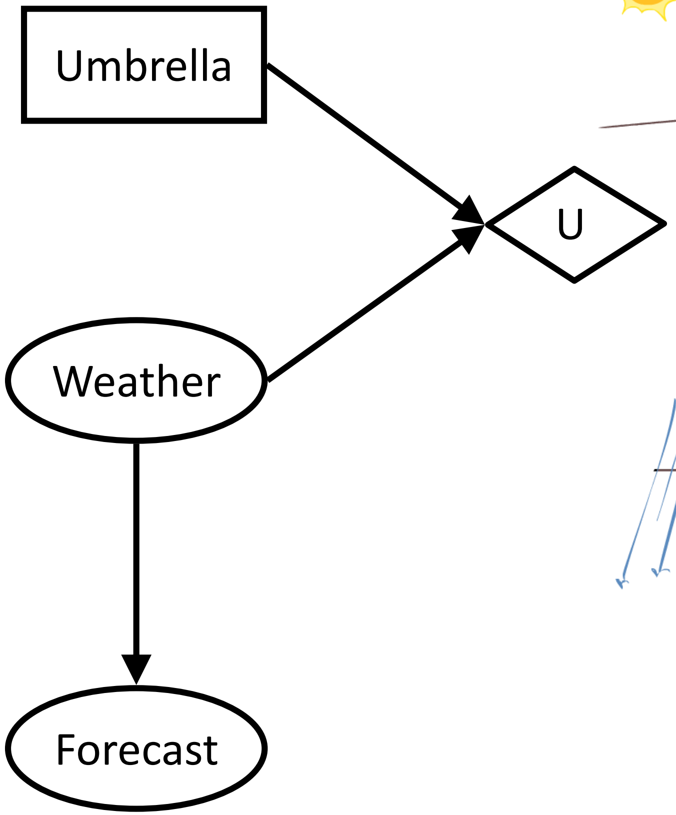


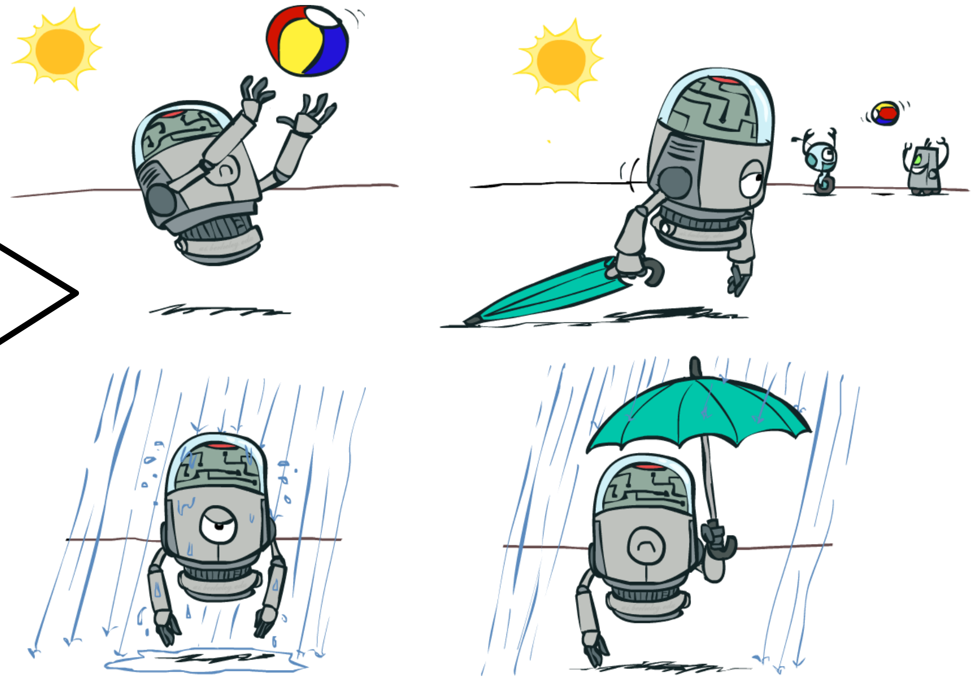
Decision Networks



Decision Networks





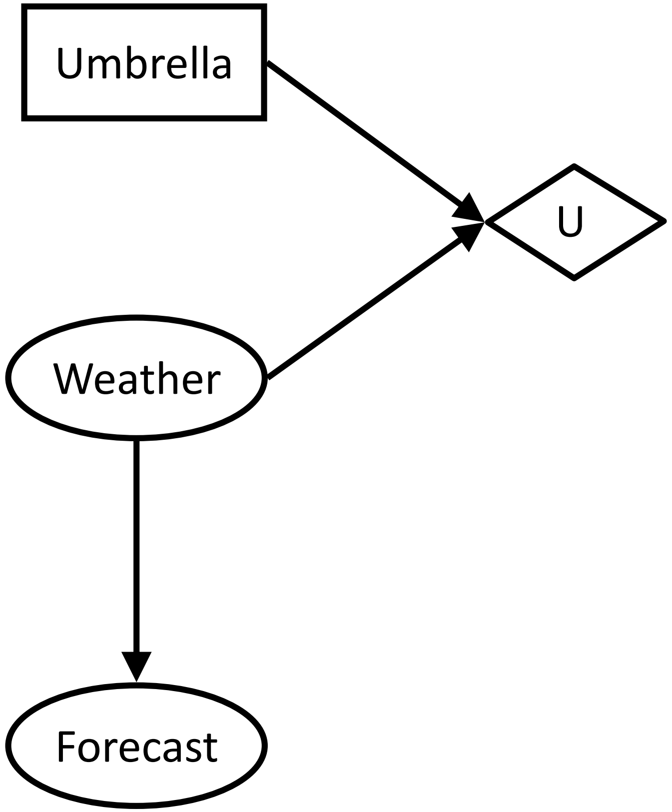


Decision Networks 2

- **MEU: choose the action which maximizes the expected utility given the evidence**

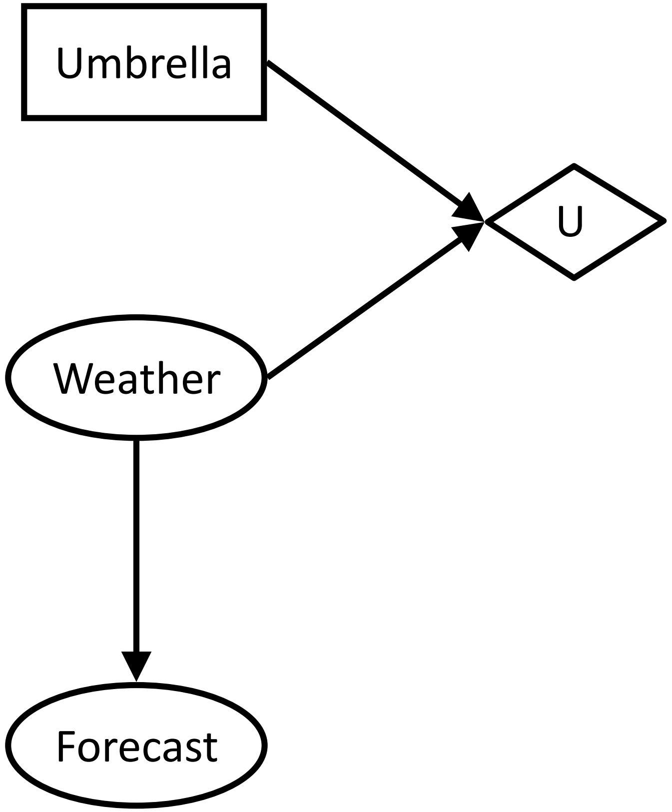
- Can directly operationalize this with decision networks
  - Bayes nets with nodes for utility and actions
  - Lets us calculate the expected utility for each action

- New node types:
  - Chance nodes (just like BNs)
  - Actions (rectangles, cannot have parents, act as observed evidence)
  - Utility node (diamond, depends on action and chance nodes)



Decision Networks 3

- Action selection
  - Instantiate all evidence
  - Set action node(s) each possible way
  - Calculate posterior for all parents of utility node, given the evidence
  - Calculate expected utility for each action
  - Choose maximizing action



Maximum Expected Utility

Umbrella = leave

$$\mathrm{EU}(\mathrm{leave}) = \sum_{w} P(w)U(\mathrm{leave}, w)$$

$$= 0.7 \cdot 100 + 0.3 \cdot 0 = 70$$

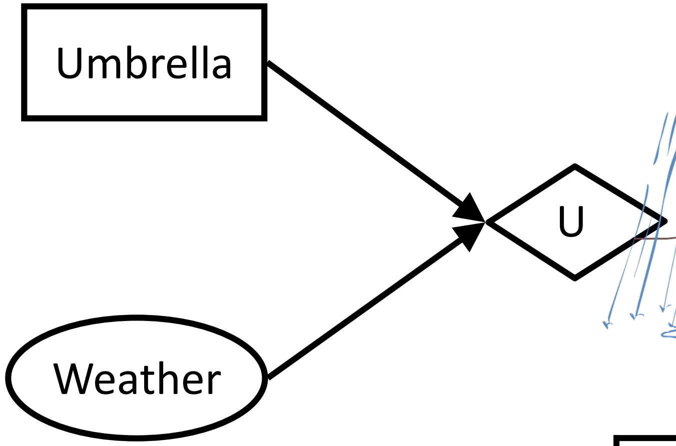
Umbrella = take

$$\mathrm{EU}(\mathrm{take}) = \sum_{w} P(w)U(\mathrm{take}, w)$$

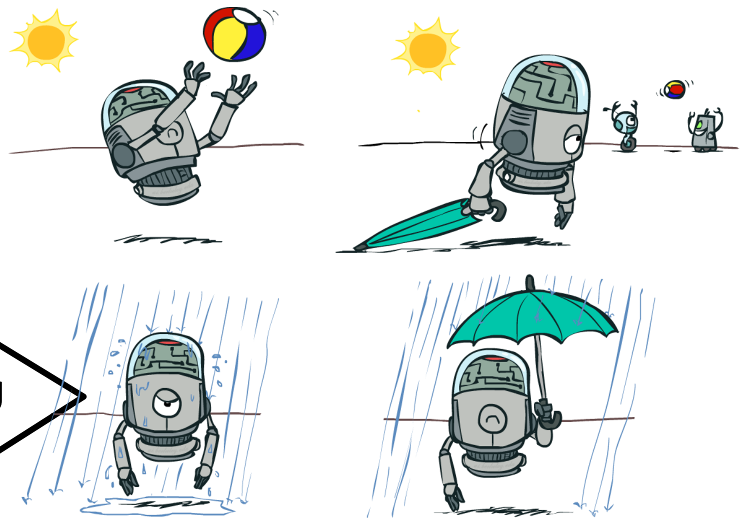
$$= 0.7 \cdot 20 + 0.3 \cdot 70 = 35$$

Optimal decision = leave

$$\mathrm{MEU}(\emptyset) = \max_{a} \mathrm{EU}(a) = 70$$

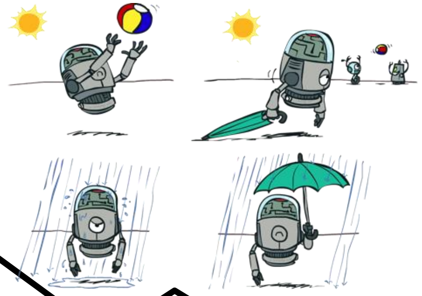


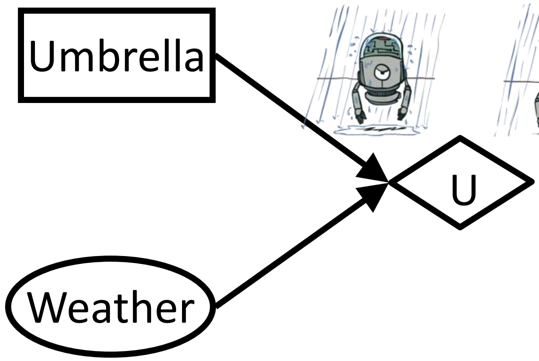
W	P(W)
sun	0.7
rain	0.3

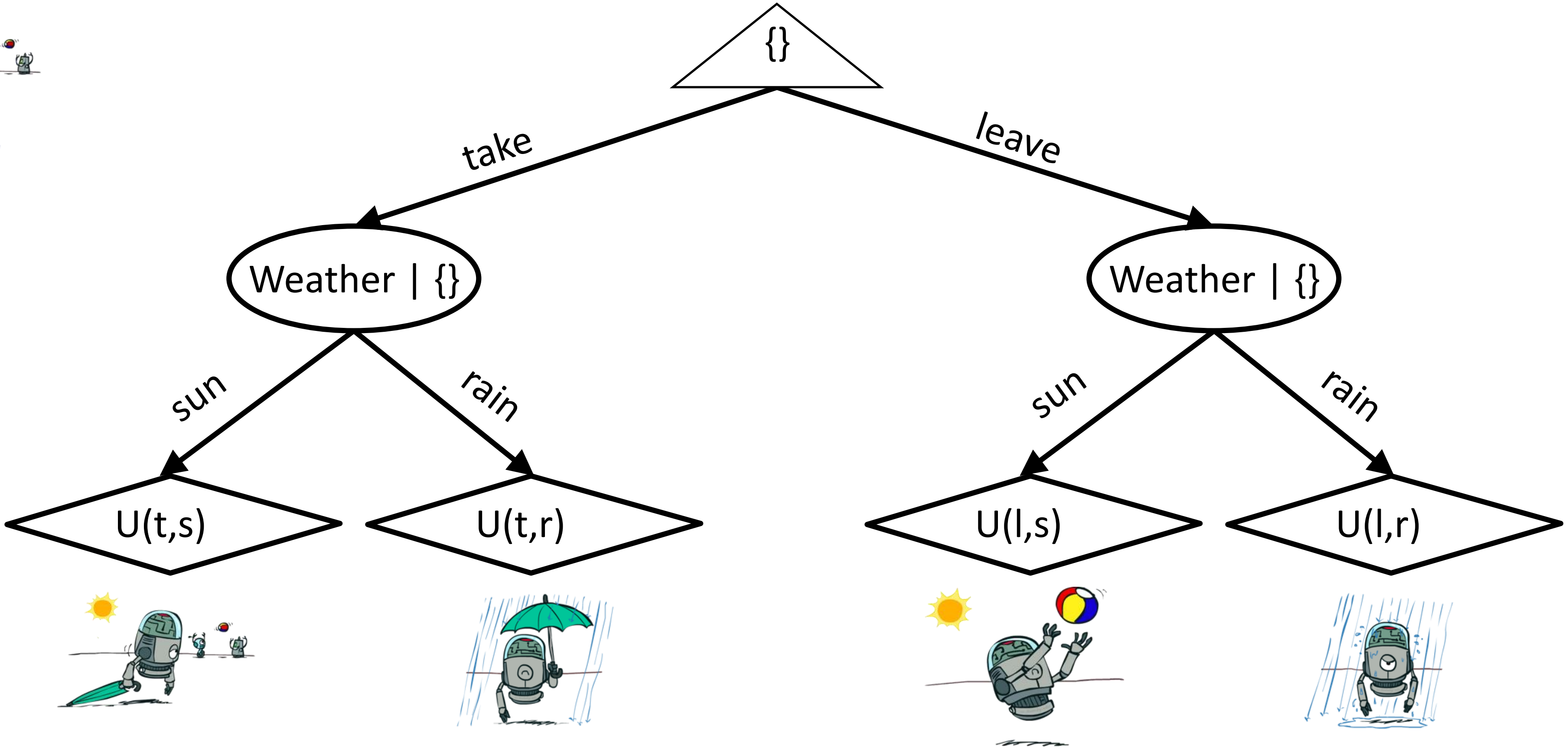


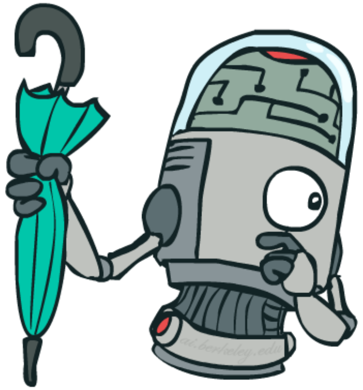
A	W	U(A,W)
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Decisions as Outcome Trees









- Almost exactly like expectimax / MDPs
- What’s changed?

Maximum Expected Utility

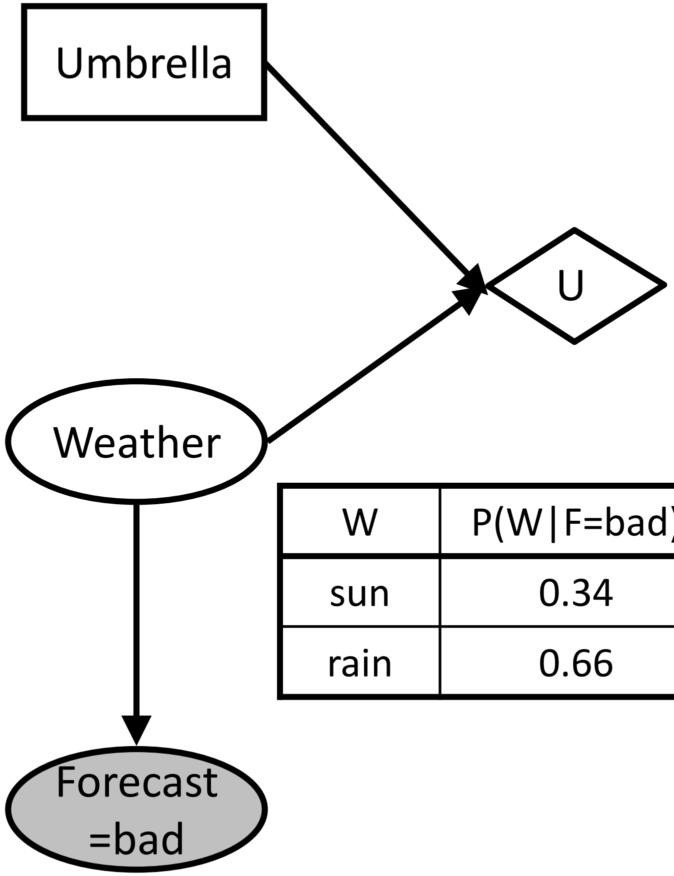
Umbrella = leave

$$\mathrm{EU}(\mathrm{leave}|\mathrm{bad}) = \sum_w P(w|\mathrm{bad})U(\mathrm{leave}, w)$$

$$P(W) \qquad P(F|W)$$

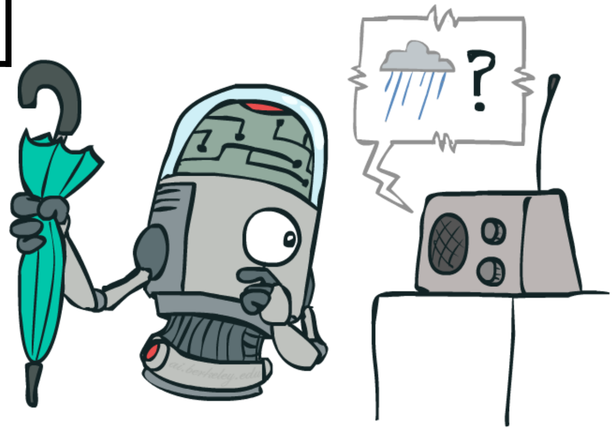
$$P(W|F) = \frac{P(W,F)}{\sum_w P(w,F)}$$

$$= \frac{P(F|W)P(W)}{\sum_w P(F|w)P(w)}$$



A	W	U(A,W)
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

W	P(W|F=bad)
sun	0.34
rain	0.66



Maximum Expected Utility 2

Umbrella = leave

$$\mathrm{EU}(\mathrm{leave}|\mathrm{bad}) = \sum_w P(w|\mathrm{bad})U(\mathrm{leave}, w)$$

$$= 0.34 \cdot 100 + 0.66 \cdot 0 = 34$$

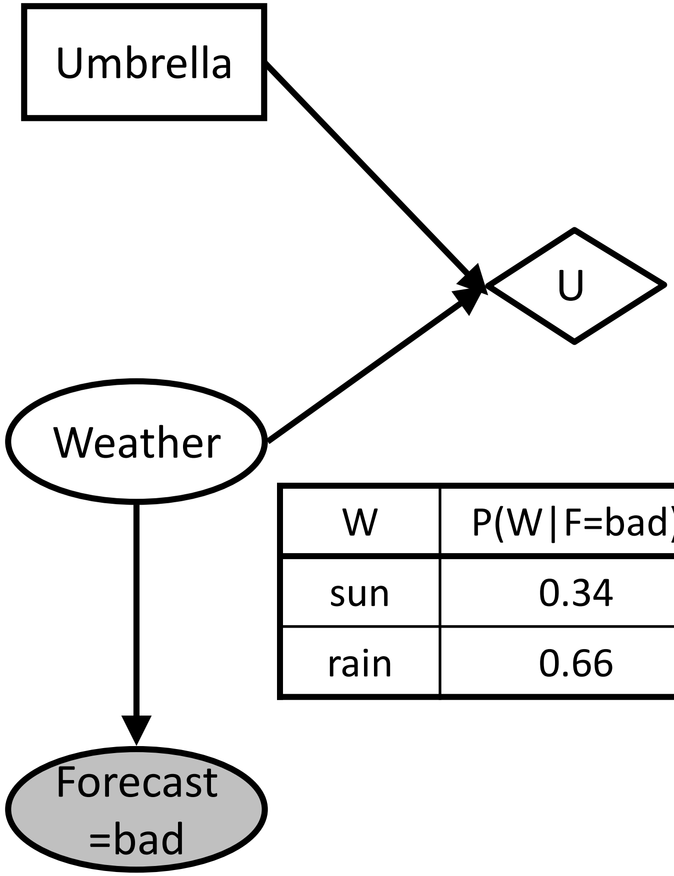
Umbrella = take

$$\mathrm{EU}(\mathrm{take}|\mathrm{bad}) = \sum_w P(w|\mathrm{bad})U(\mathrm{take}, w)$$

$$= 0.34 \cdot 20 + 0.66 \cdot 70 = 53$$

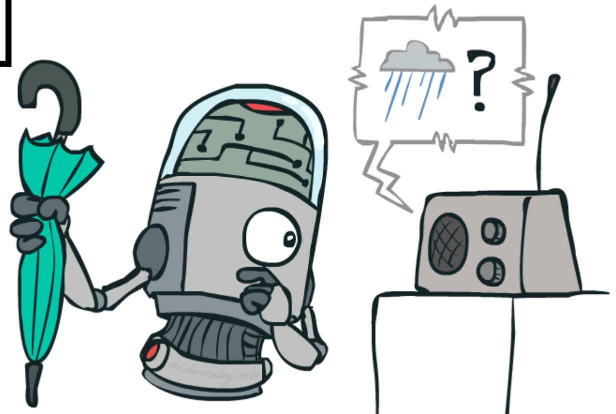
Optimal decision = take

$$\mathrm{MEU}(F = \mathrm{bad}) = \max_a \mathrm{EU}(a|\mathrm{bad}) = 53$$

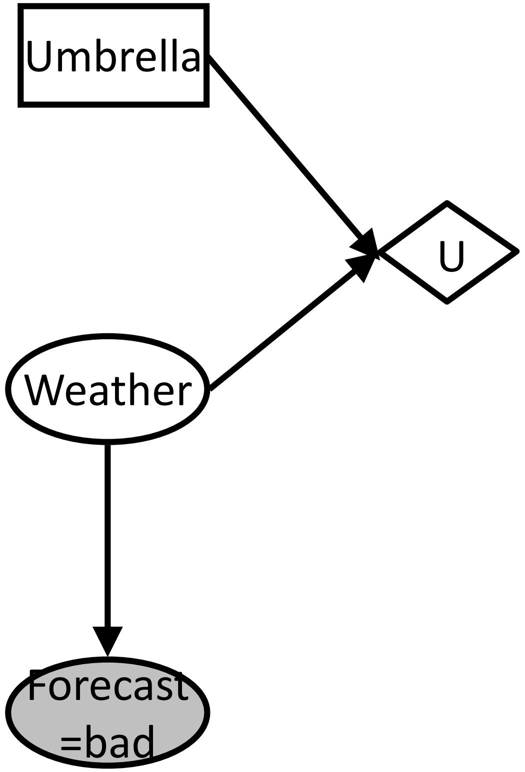


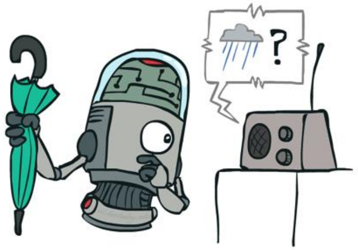
A	W	U(A,W)
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

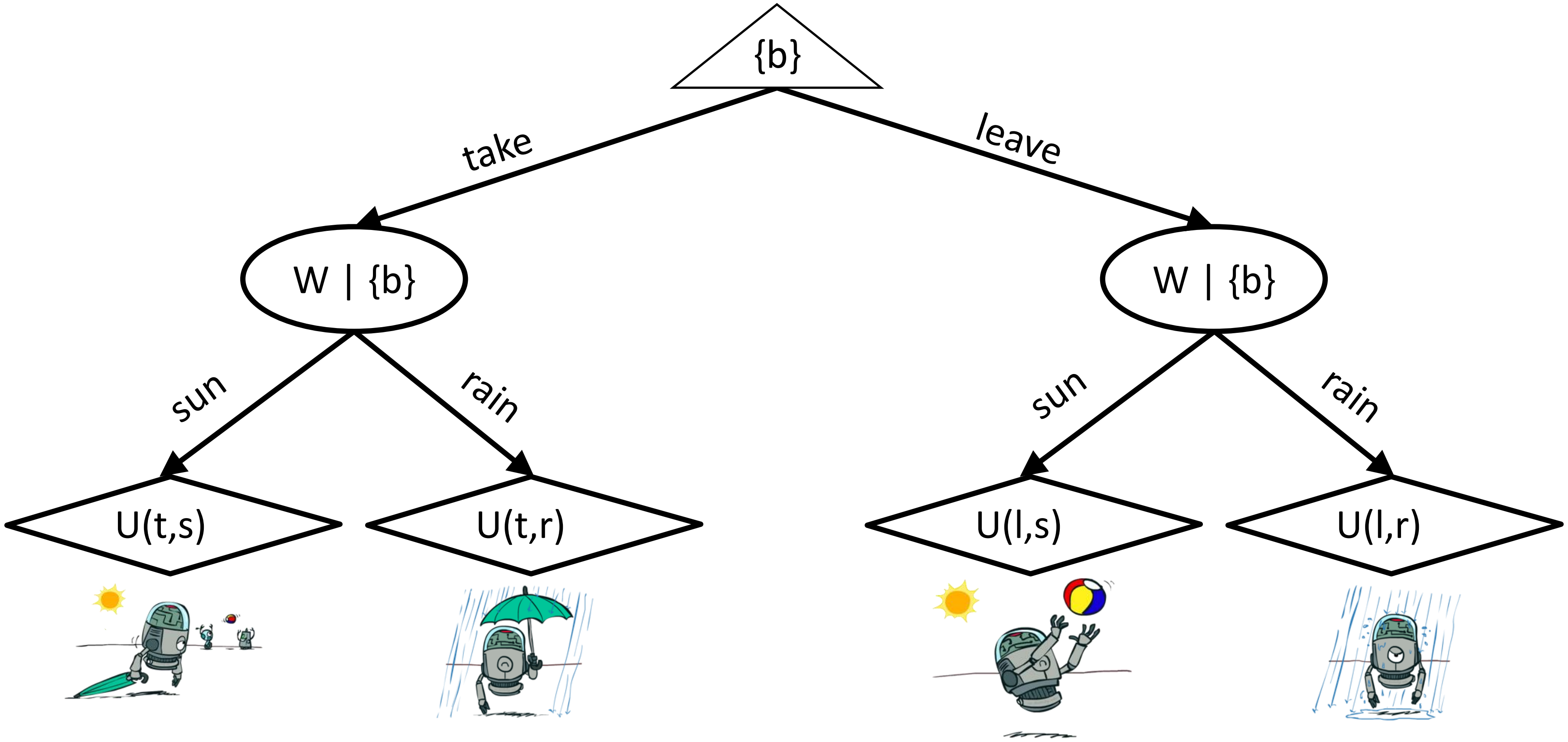
W	P(W|F=bad)
sun	0.34
rain	0.66



Decisions as Outcome Trees



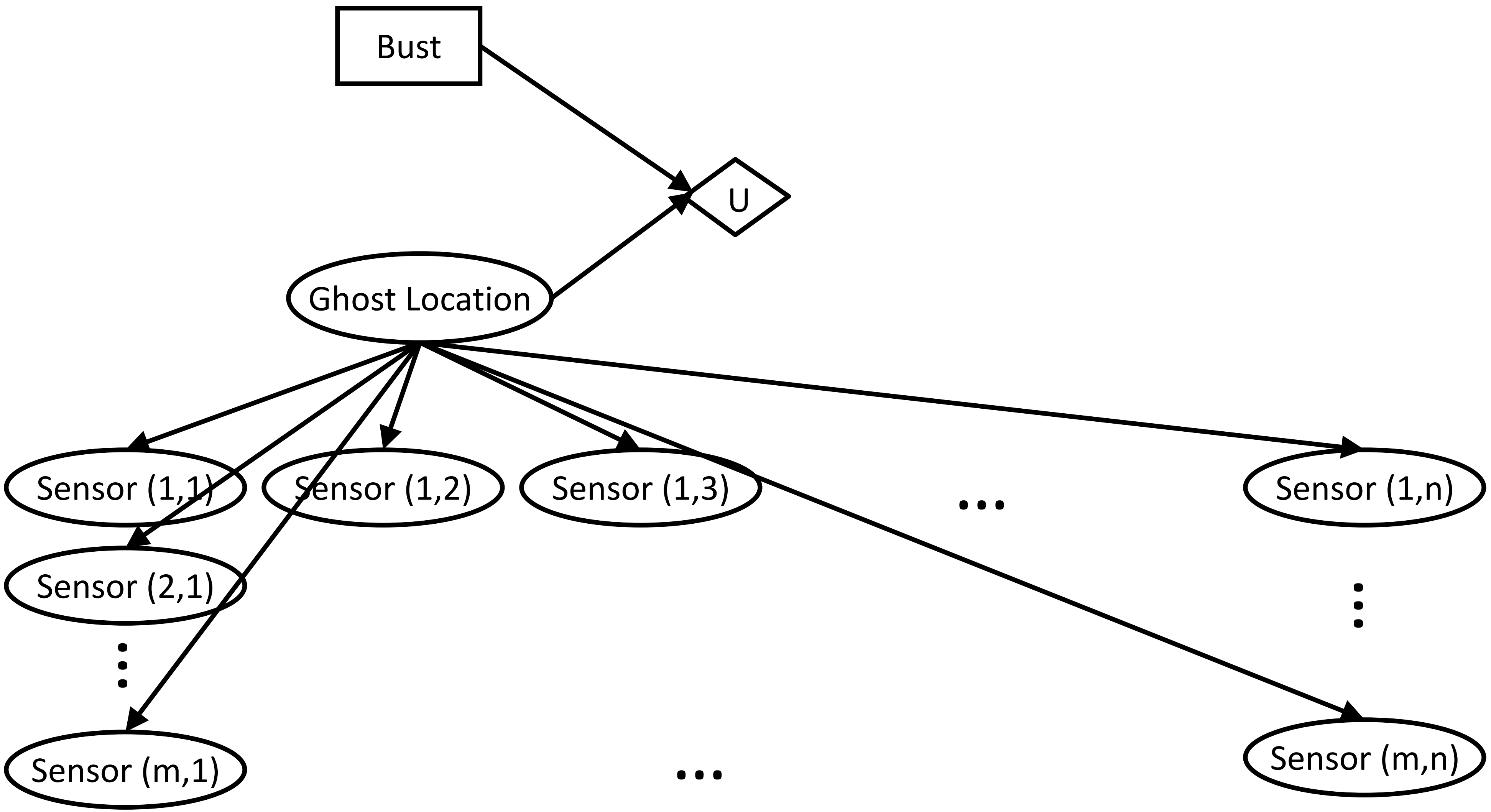




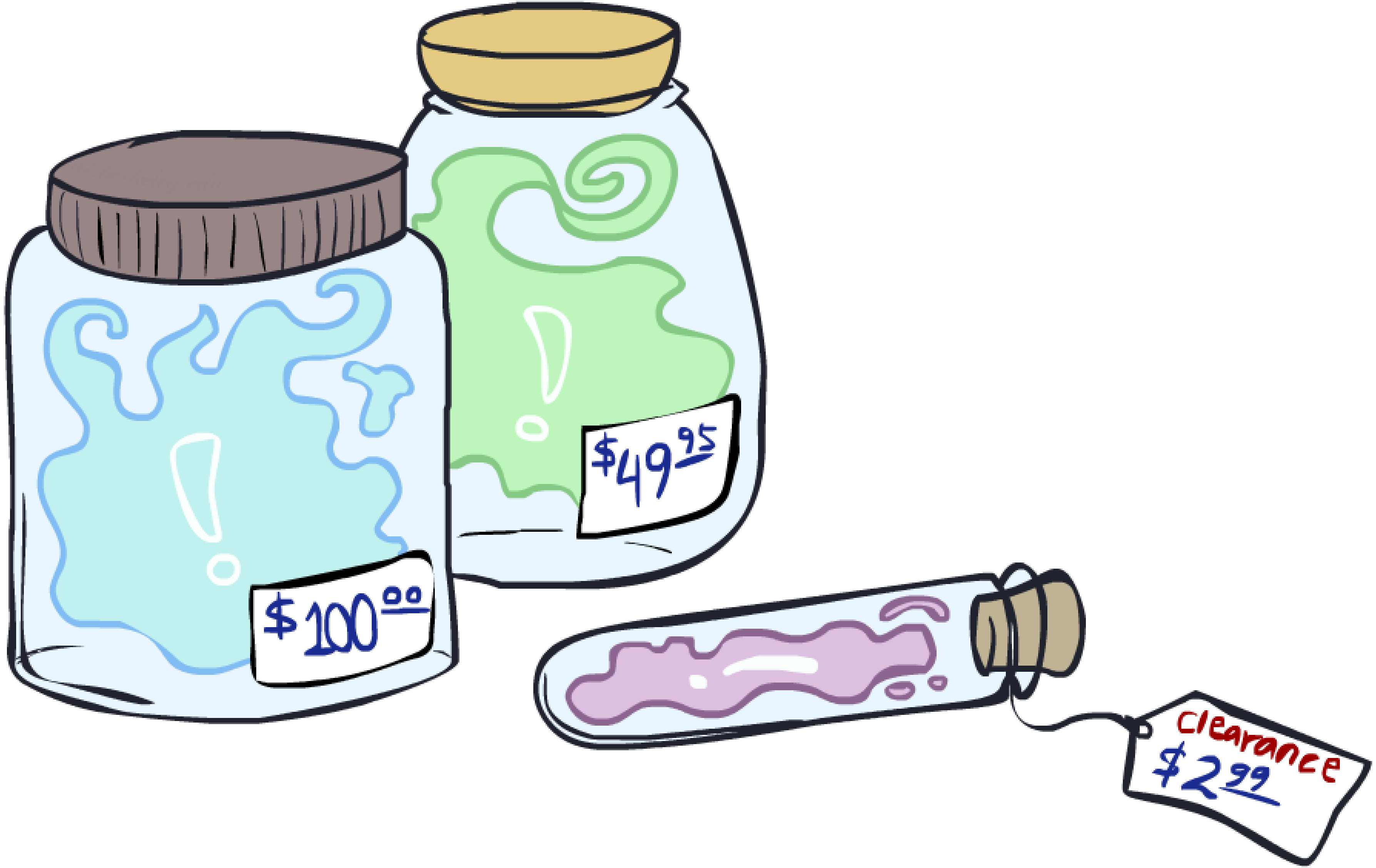
Video of Demo Ghostbusters with Probability

Ghostbusters Decision Network

Demo: Ghostbusters with probability



Value of Information



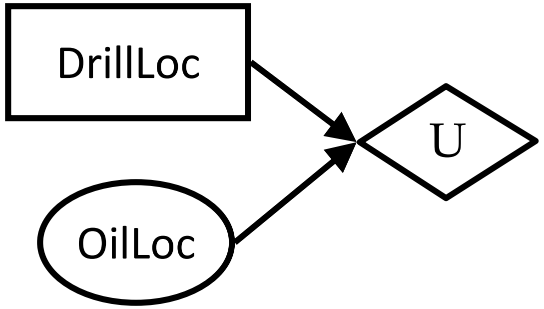
Value of Information

- Idea: compute value of acquiring evidence
  - Can be done directly from decision network

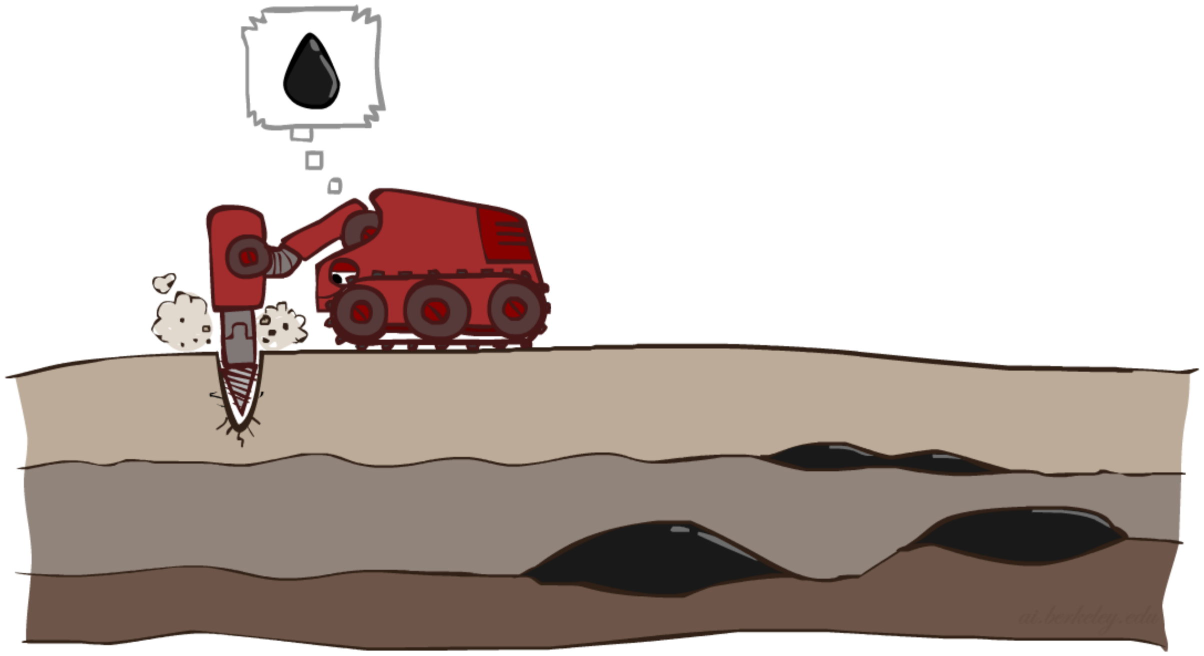
- Example: buying oil drilling rights
  - Two blocks A and B, exactly one has oil, worth k
  - You can drill in one location
  - Prior probabilities 0.5 each, & mutually exclusive
  - Drilling in either A or B has EU = k/2, MEU = k/2

- Question: what's the value of information of O?
  - Value of knowing which of A or B has oil
  - Value is expected gain in MEU from new info
  - Survey may say "oil in a" or "oil in b," prob 0.5 each
  - If we know OilLoc, MEU is k (either way)
  - Gain in MEU from knowing OilLoc?
  - VPI(OilLoc) = k/2
  - Fair price of information: k/2

O	P
a	1/2
b	1/2



D	O	U
a	a	k
a	b	0
b	a	0
b	b	k



Value of Perfect Information

MEU with no evidence

$$\mathrm{MEU}(\emptyset) = \max_a \mathrm{EU}(a) = 70$$

MEU if forecast is bad

$$\mathrm{MEU}(F = \mathrm{bad}) = \max_a \mathrm{EU}(a|\mathrm{bad}) = 53$$

MEU if forecast is good

$$\mathrm{MEU}(F = \mathrm{good}) = \max_a \mathrm{EU}(a|\mathrm{good}) = 95$$

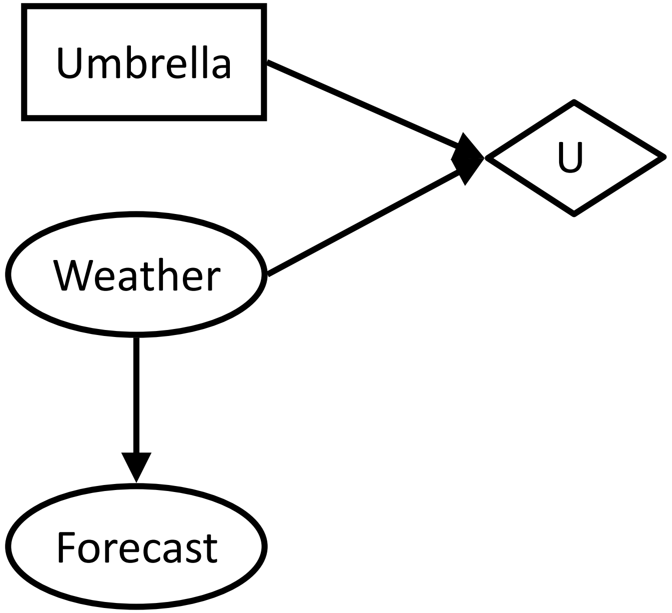
Forecast distribution

F	P(F)
good	0.59
bad	0.41

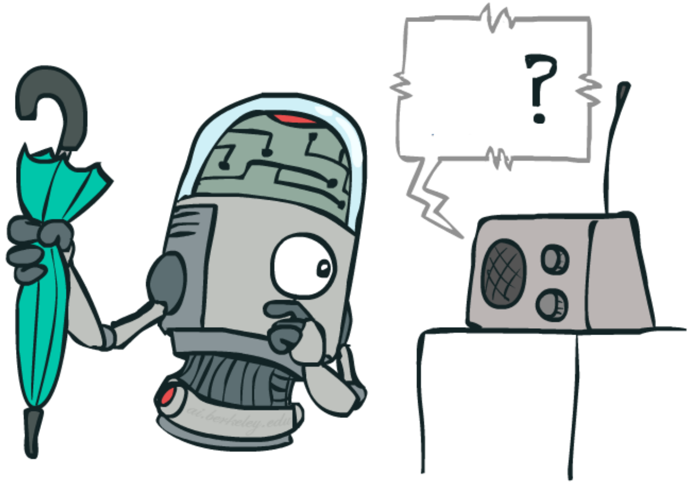
$$0.59 \cdot (95) + 0.41 \cdot (53) - 70$$

$$77.8 - 70 = 7.8$$

$$\mathrm{VPI}(E'|e) = \left(\sum_{e'} P(e'|e)\mathrm{MEU}(e, e') \right) - \mathrm{MEU}(e)$$



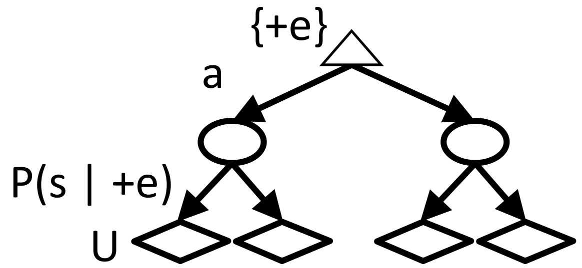
A	W	U
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70



Value of Information

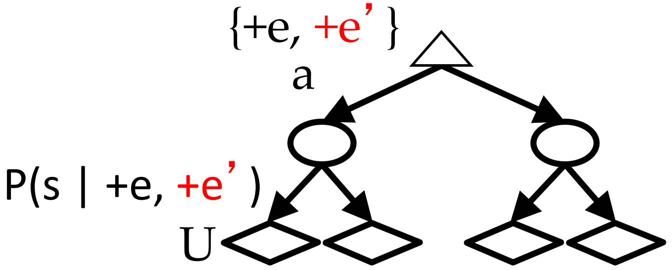
- Assume we have evidence E=e. Value if we act now:

$$\text{MEU}(e) = \max_a \sum_s P(s|e)\, U(s,a)$$



- Assume we see that E’ = e’. Value if we act then:

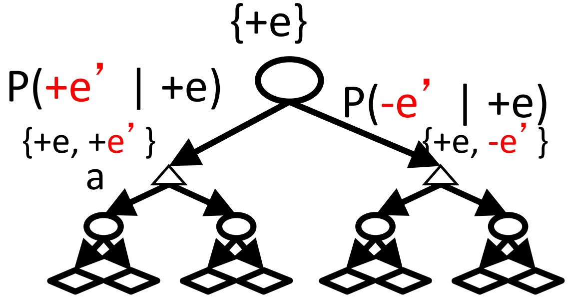
$$\text{MEU}(e,e') = \max_a \sum_s P(s|e,e')\, U(s,a)$$



- BUT E’ is a random variable whose value is unknown, so we don’t know what e’ will be

- Expected value if E’ is revealed and then we act:

$$\text{MEU}(e,E') = \sum_{e'} P(e'|e)\text{MEU}(e,e')$$



- Value of information: how much MEU goes up by revealing E’ first then acting, over acting now:

$$\text{VPI}(E'|e) = \text{MEU}(e,E') - \text{MEU}(e)$$

Value of Information 2

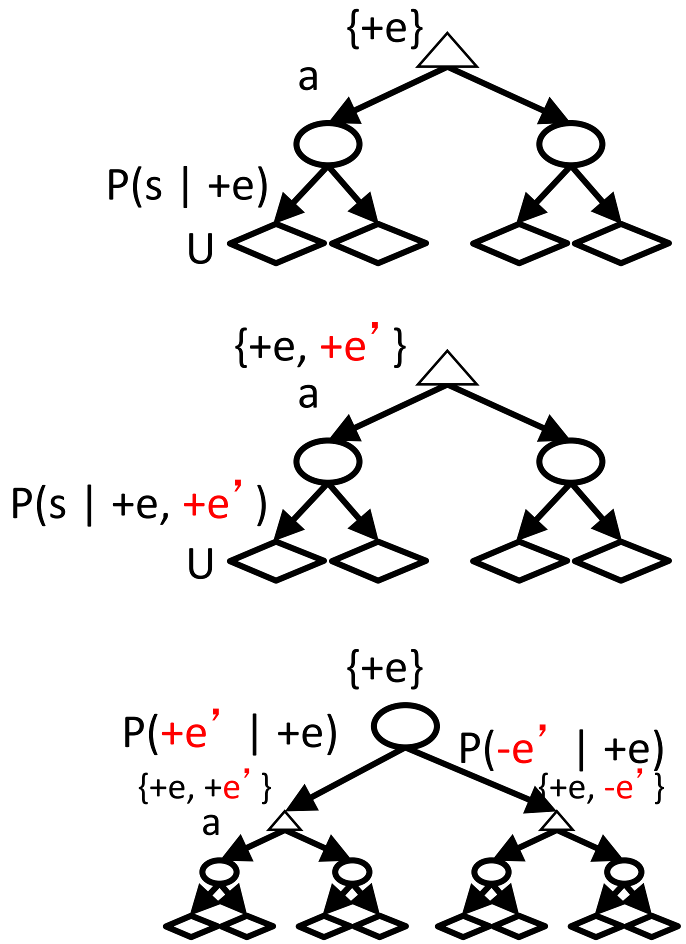
$$\text{MEU}(e, E') = \sum_{e'} P(e'|e)\text{MEU}(e, e')$$

$$= \sum_{e'} P(e'|e) \max_a \sum_s P(s|e, e')U(s, a)$$

$$\text{MEU}(e) = \max_a \sum_s P(s|e)\, U(s, a)$$

$$= \max_a \sum_{e'} \sum_s P(s, e'|e)U(s, a)$$

$$= \max_a \sum_{e'} P(e|e') \sum_s P(s|e, e')U(s, a)$$



VPI Properties

- Nonnegative

$$\forall E', e : \text{VPI}(E'|e) \geq 0$$

- Nonadditive

  (think of observing E_j twice)

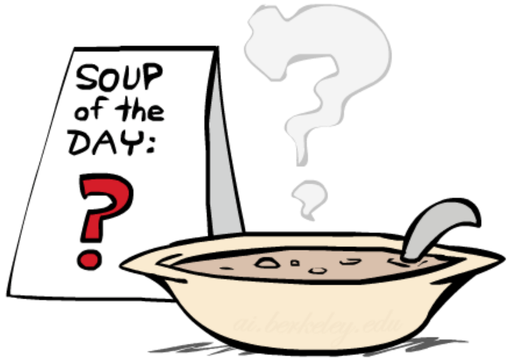
$$\text{VPI}(E_j, E_k|e) \neq \text{VPI}(E_j|e) + \text{VPI}(E_k|e)$$

- Order-independent

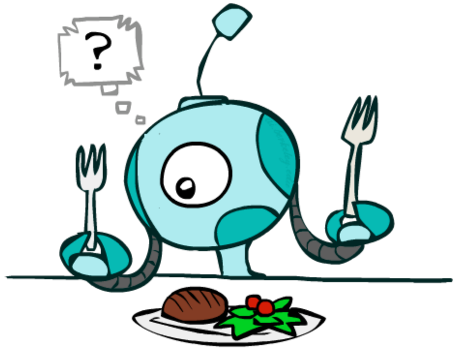
$$\text{VPI}(E_j, E_k|e) = \text{VPI}(E_j|e) + \text{VPI}(E_k|e, E_j)$$
$$= \text{VPI}(E_k|e) + \text{VPI}(E_j|e, E_k)$$

Quick VPI Questions

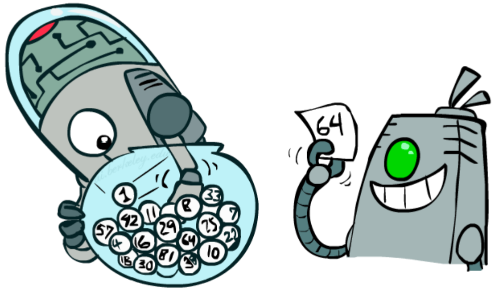
- The soup of the day is either clam chowder or split pea, but you wouldn’t order either one. What’s the value of knowing which it is?



- There are two kinds of plastic forks at a picnic. One kind is slightly sturdier. What’s the value of knowing which?



- You’re playing the lottery. The prize will be $0 or $100. You can play any number between 1 and 100 (chance of winning is 1%). What is the value of knowing the winning number?

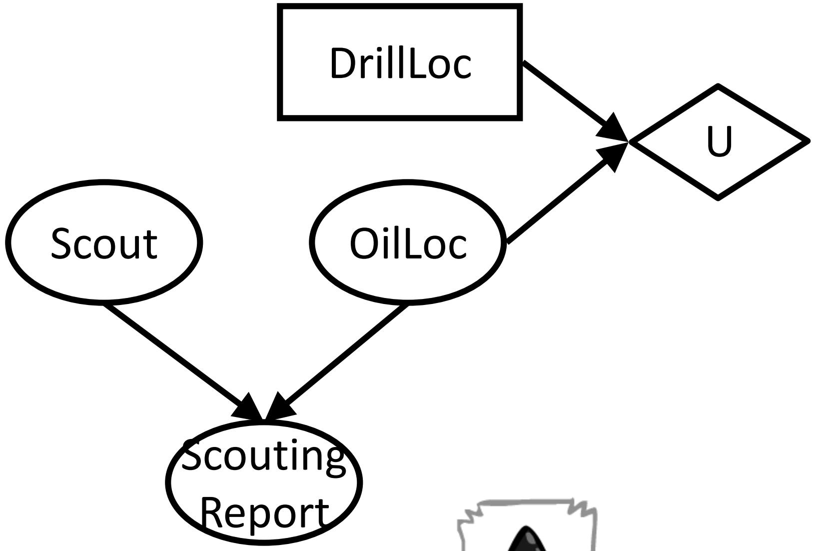


Value of Imperfect Information?

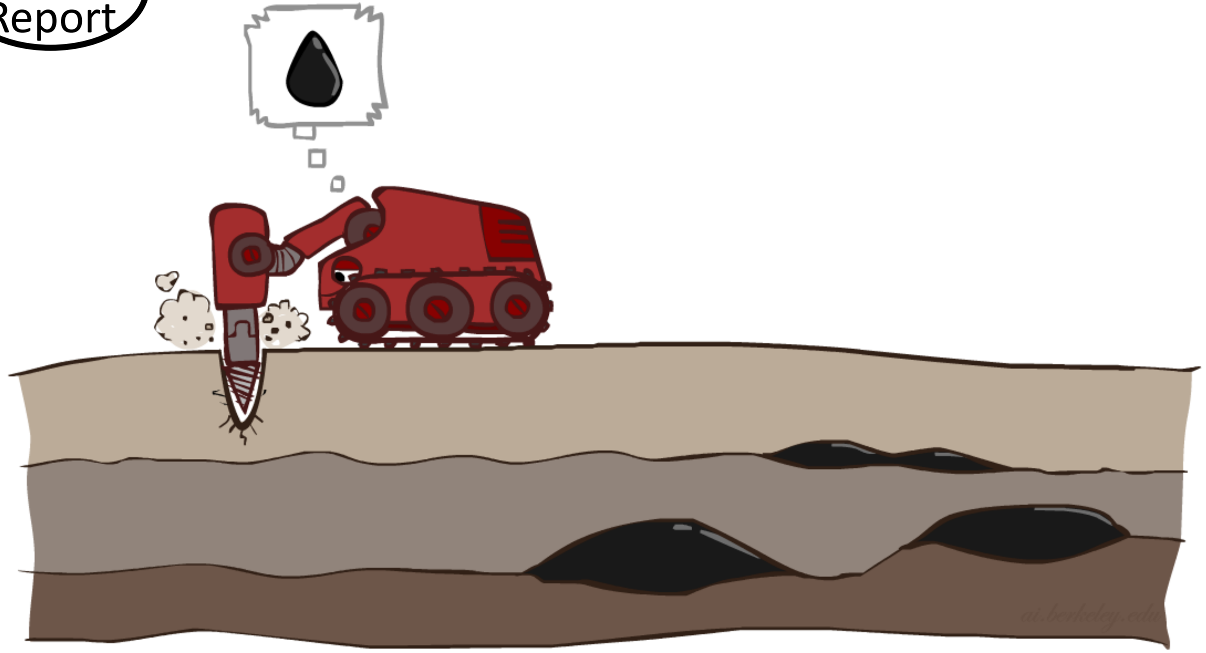
- No such thing
- Information corresponds to the observation of a node in the decision network
- If data is “noisy” that just means we don’t observe the original variable, but another variable which is a noisy version of the original one

VPI Question

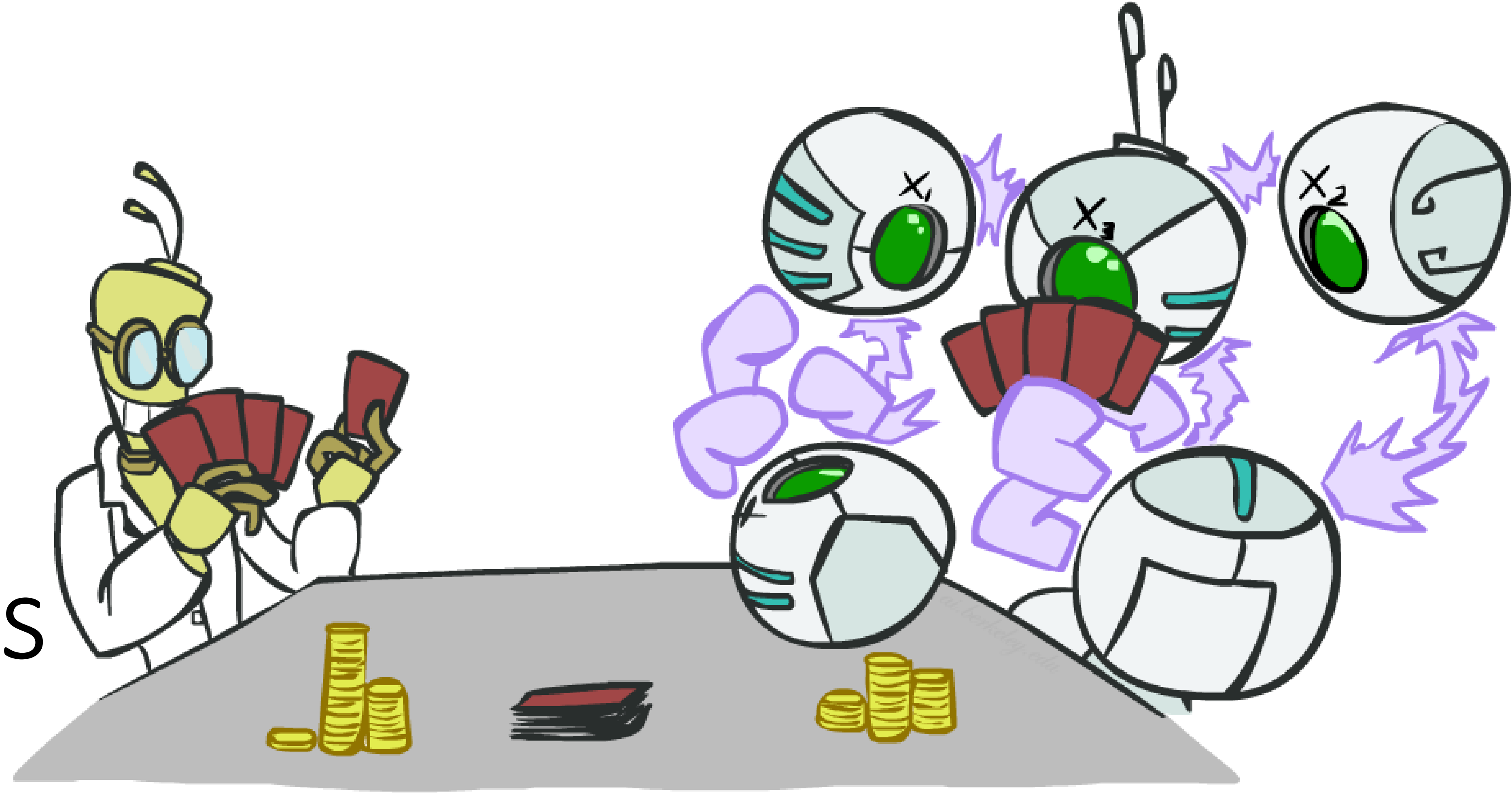
- VPI(OilLoc) ?
- VPI(ScoutingReport) ?
- VPI(Scout) ?
- VPI(Scout | ScoutingReport) ?



- Generally:
  If Parents(U) $\perp\!\!\!\perp$ Z | CurrentEvidence
  Then VPI(Z | CurrentEvidence) = 0



POMDPs

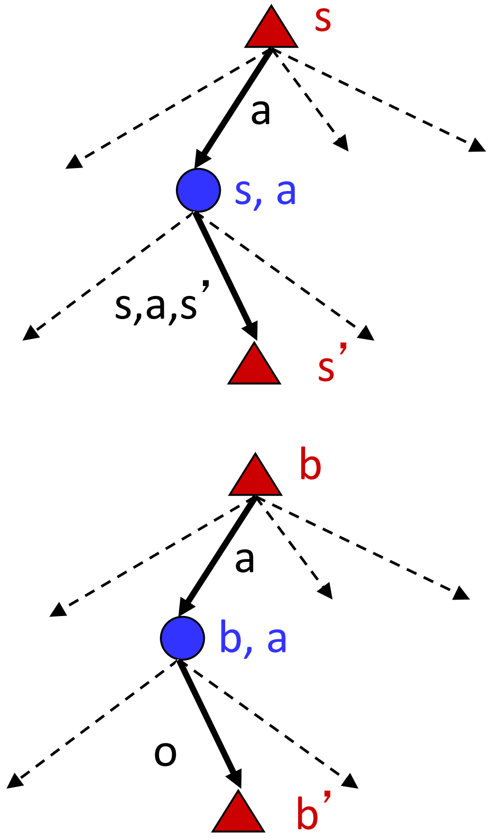


POMDPs

- MDPs have:
  - States S
  - Actions A
  - Transition function P(s' |s,a) (or T(s,a,s'))
  - Rewards R(s,a,s')

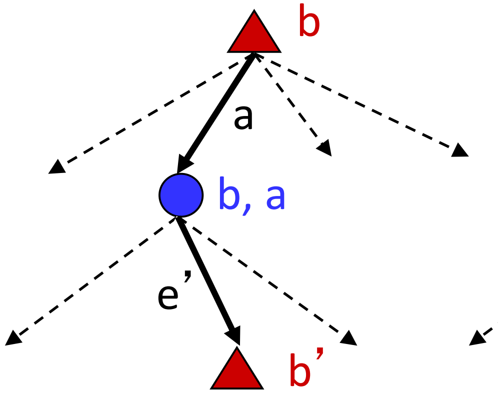
- POMDPs add:
  - Observations O
  - Observation function P(o|s) (or O(s,o))

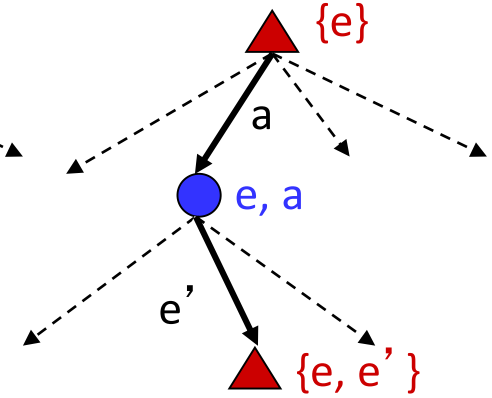
- POMDPs are MDPs over belief states b (distributions over S)



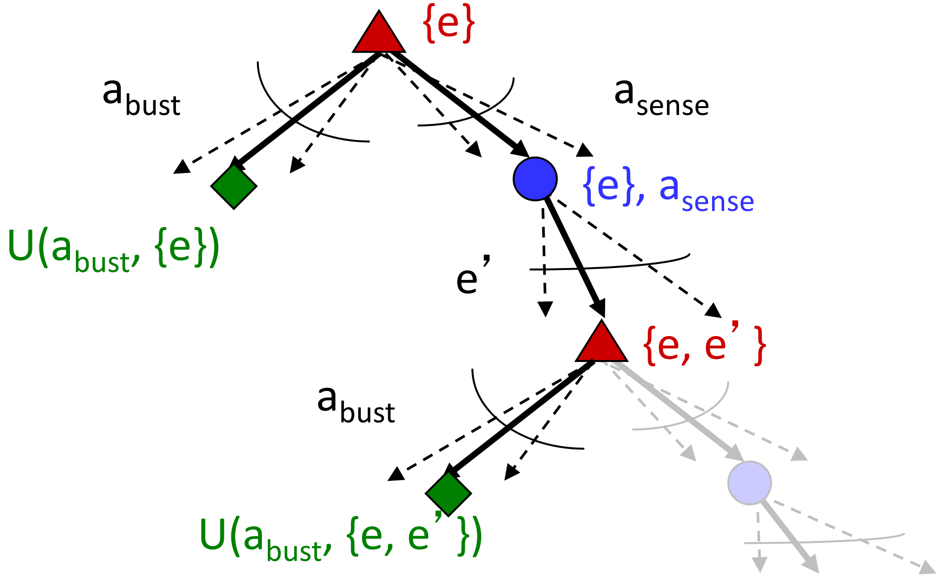
Example: Ghostbusters

- In (static) Ghostbusters:
  - Belief state determined by evidence to date {e}
  - Tree really over evidence sets
  - Probabilistic reasoning needed to predict new evidence given past evidence





- Solving POMDPs
  - One way: use truncated expectimax to compute approximate value of actions
  - What if you only considered busting or one sense followed by a bust?
  - You get a VPI-based agent!



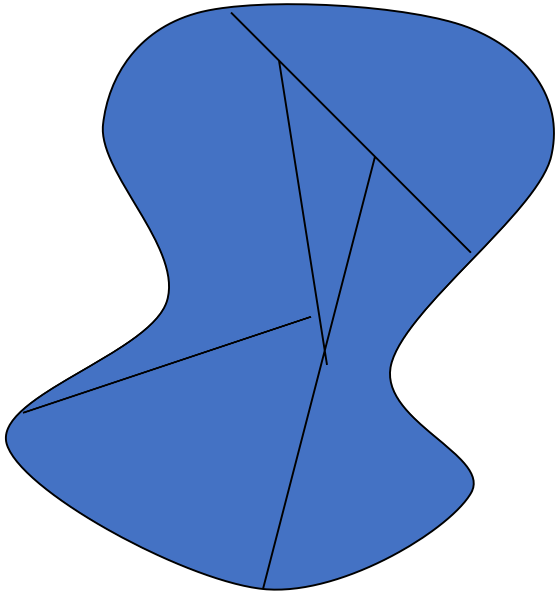
Video of Demo Ghostbusters with VPI

More Generally*

- General solutions map belief functions to actions
  - Can divide regions of belief space (set of belief functions) into policy regions (gets complex quickly)
  - Can build approximate policies using discretization methods
  - Can factor belief functions in various ways

- Overall, POMDPs are very (actually PSACE-) hard

- Most real problems are POMDPs, but we can rarely solve then in general!



Summary

Shuai Li
https://shuaili8.github.io

- Bayes rule
- Inference
- Variable Elimination
- Sampling
- Decision Networks

Questions?