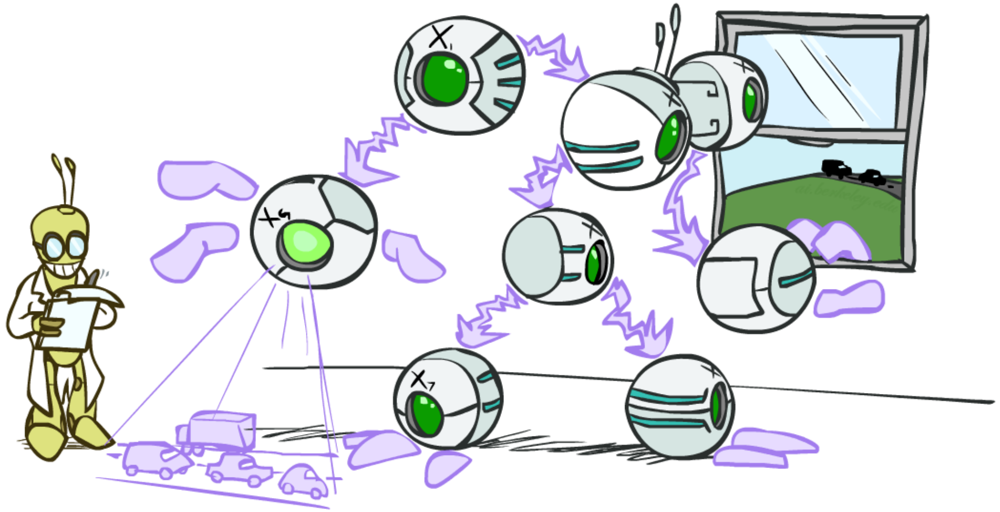
Artificial Intelligence - INFOF311

Bayes nets, exact inference

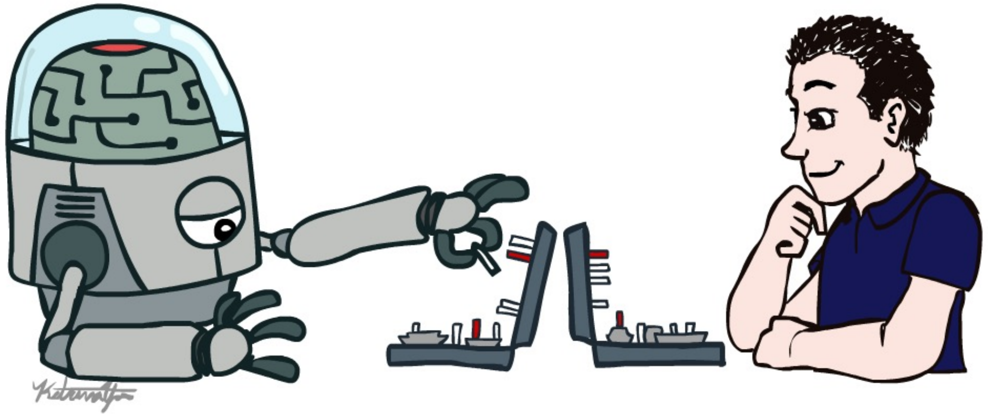


Instructor : Tom Lenaerts

Acknowledgement

We thank Stuart Russell for his generosity in allowing us to use the slide set of the UC Berkeley Course CS188, Introduction to Artificial Intelligence. These slides were created by Dan Klein, Pieter Abbeel and Anca Dragan for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu.

CHAI Center for Human-Compatible Artificial Intelligence



The slides for INFOF311 are slightly modified versions of the slides of the spring and summer CS188 sessions in 2021 and 2022

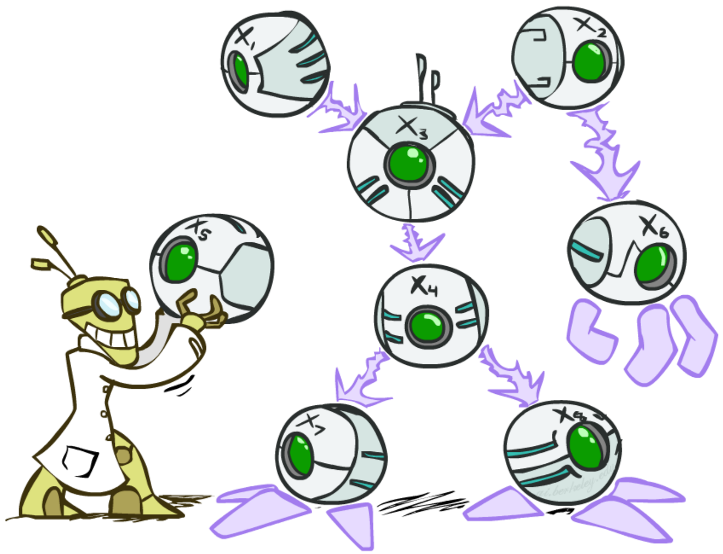
Bayes Net Semantics

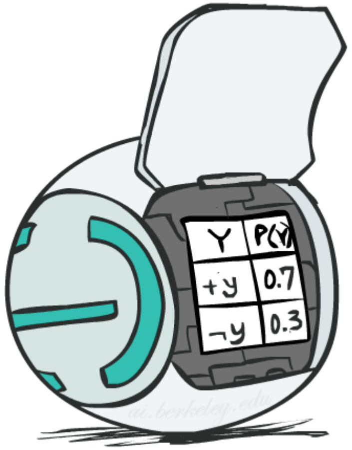
- A Bayes net is an efficient encoding of a probabilistic model of a domain
- A directed, acyclic graph, one node per random variable
- A conditional probability table (CPT) for each node
  - A collection of distributions over X, one for each combination of parents' values

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

- 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:

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





Size of a Bayes Net

- How big is a joint distribution over N Boolean variables?

  2^N

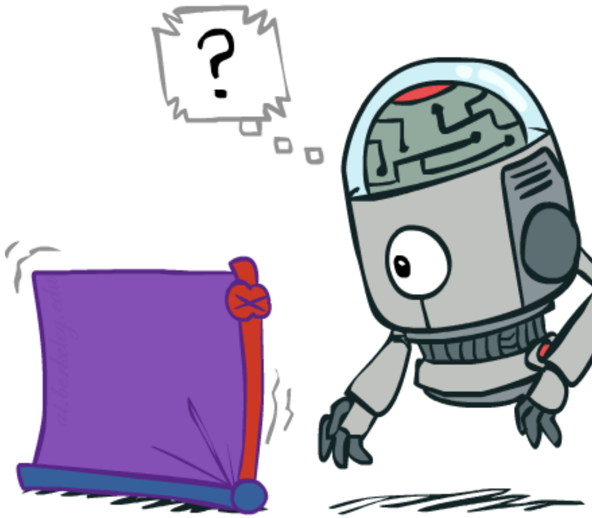
- How big is an N-node net if nodes have up to k parents?

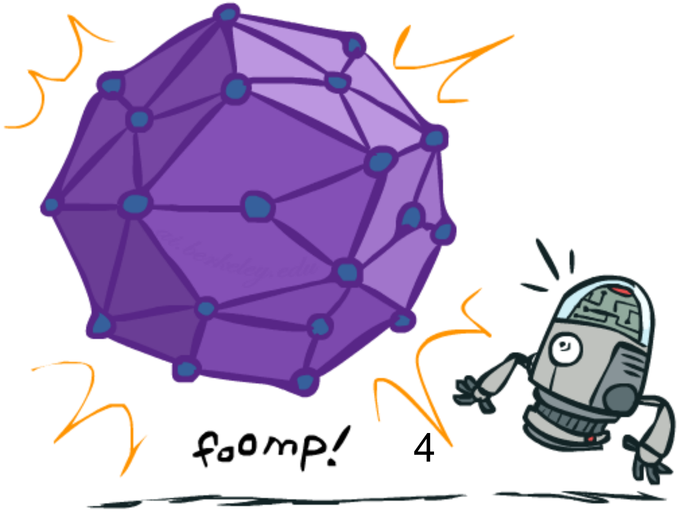
  $O(N * 2^{k+1})$

- Both give you the power to calculate

$$P(X_1, X_2, \ldots X_n)$$

- BNs: Huge space savings!
- Also easier to elicit local CPTs
- Also faster to answer queries (this lecture!)

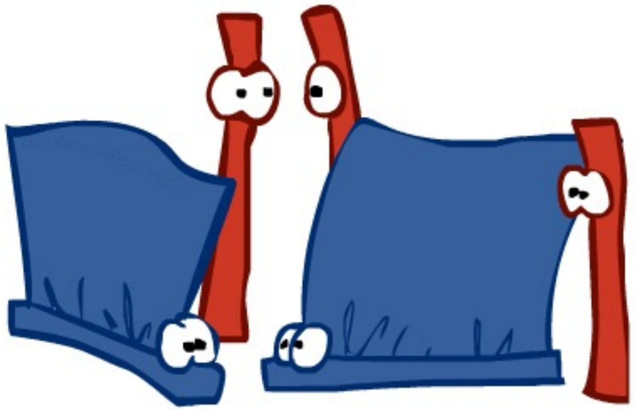


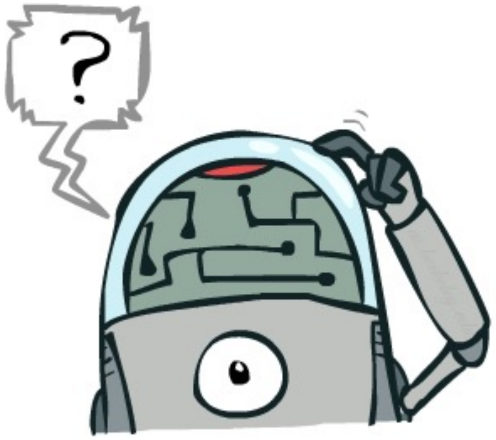


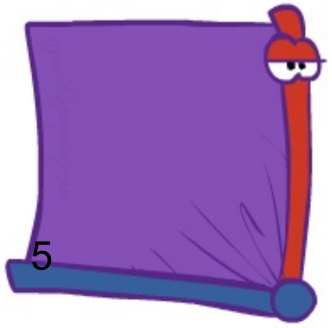
Inference

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

- Examples:
  - Posterior marginal probability
    - $P(Q|e_1,..,e_k)$
    - E.g., what disease might I have?
  - Most likely explanation:
    - $\text{argmax}_{q,r,s}\ P(Q=q,R=r,S=s|e_1,..,e_k)$
    - E.g., what did they say?







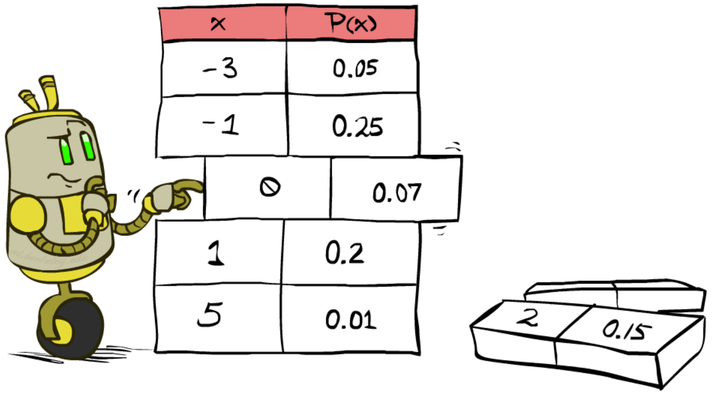
Inference by Enumeration

- Probability model $P(X_1, \ldots, X_n)$ is given
- Partition the variables $X_1, \ldots, X_n$ into sets as follows:
  - Evidence variables: $\boldsymbol{E} = \boldsymbol{e}$
  - Query variables: $\boldsymbol{Q}$
  - Hidden variables: $\boldsymbol{H}$

- We want:

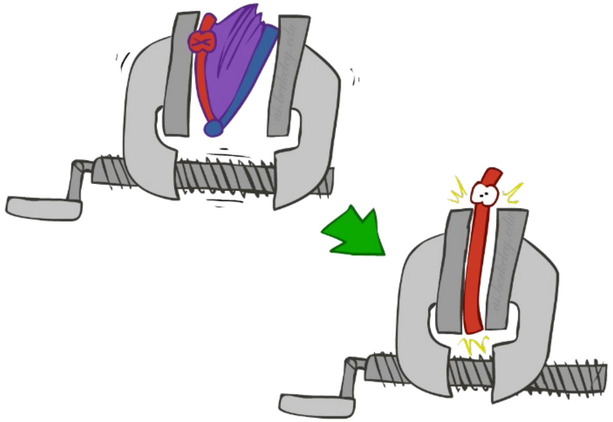
$$P(\boldsymbol{Q} \mid \boldsymbol{e})$$

- Step 1: Select the entries consistent with the evidence



- Step 2: Sum out $\boldsymbol{H}$ from model to get joint of query and evidence

$$P(\boldsymbol{Q}, \boldsymbol{e}) = \sum_h \underbrace{P(\boldsymbol{Q}, \boldsymbol{h}, \boldsymbol{e})}_{X_1, \ldots, X_n}$$

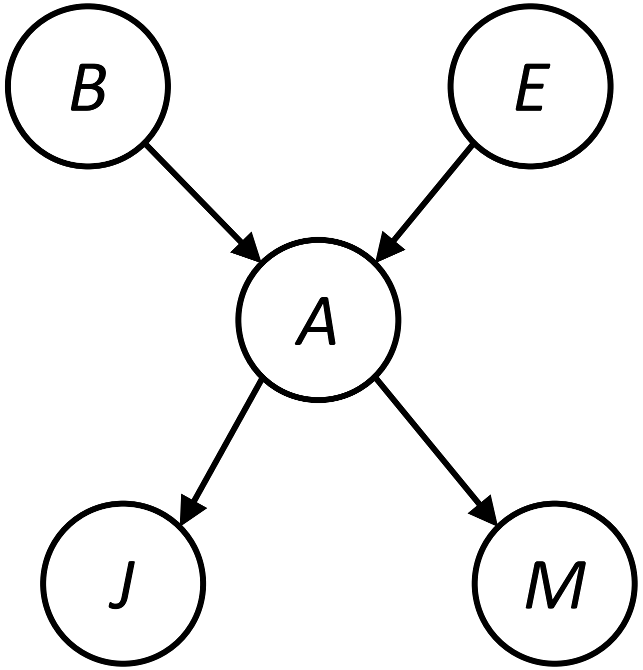


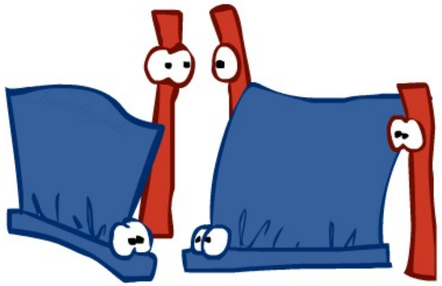
- Step 3: Normalize

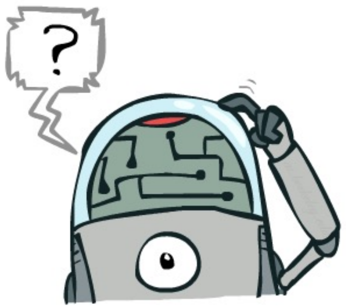
$$P(\boldsymbol{Q} \mid \boldsymbol{e}) = \alpha\, P(\boldsymbol{Q}, \boldsymbol{e})$$

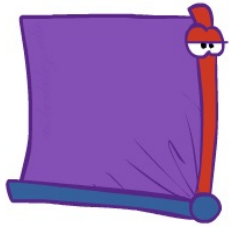
Inference by Enumeration in Bayes Net

- Reminder of inference by enumeration:
  - Any probability of interest can be computed by summing entries from the joint distribution: $P(\boldsymbol{Q} \mid \boldsymbol{e}) = \alpha \sum_h P(\boldsymbol{Q}, \boldsymbol{h}, \boldsymbol{e})$
  - Entries from the joint distribution can be obtained from a BN by multiplying the corresponding conditional probabilities
- $P(B \mid j, m) = \alpha \sum_{e,a} P(B, e, a, j, m)$
- $= \alpha \sum_{e,a} P(B)\, P(e)\, P(a \mid B,e)\, P(j \mid a)\, P(m \mid a)$
- So, inference in Bayes nets means computing sums of products of numbers. Sounds easy!!
- Problem: sums of ***exponentially many*** products!









Can we do better?

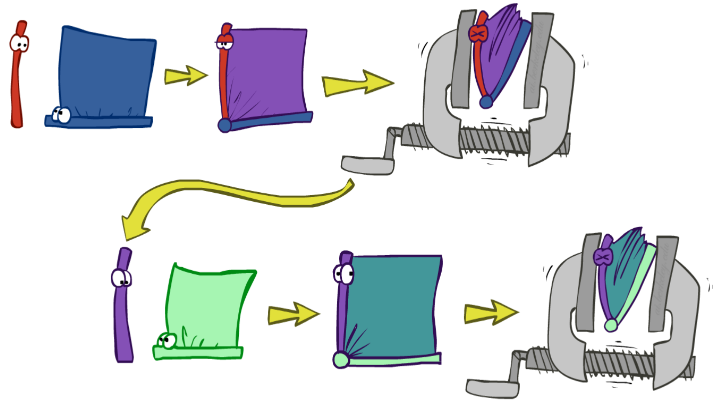
- Consider **uwy + uwz + uxy + uxz + vwy + vwz + vxy +vxz**
  - 16 multiplies, 7 adds
  - Lots of repeated sub-expressions!
- Rewrite as **(u+v)(w+x)(y+z)**
  - 2 multiplies, 3 adds
- $\sum_{e,a} P(B)\, \mathrm{P(e)}\, \mathrm{P}(\mathrm{a} \mid B,e)\, P(j \mid a)\, P(m \mid a)$
- $= P(B)P(e)P(\mathrm{a} \mid B,e)P(j \mid a)P(m \mid a) + P(B)P(\neg e)P(\mathrm{a} \mid B,\neg e)P(j \mid a)P(m \mid a)$

  $+ P(B)P(e)P(\neg \mathrm{a} \mid B,e)P(j \mid \neg a)P(m \mid \neg a) + P(B)P(\neg e)P(\neg \mathrm{a} \mid B,\neg e)P(j \mid \neg a)P(m \mid \neg a)$

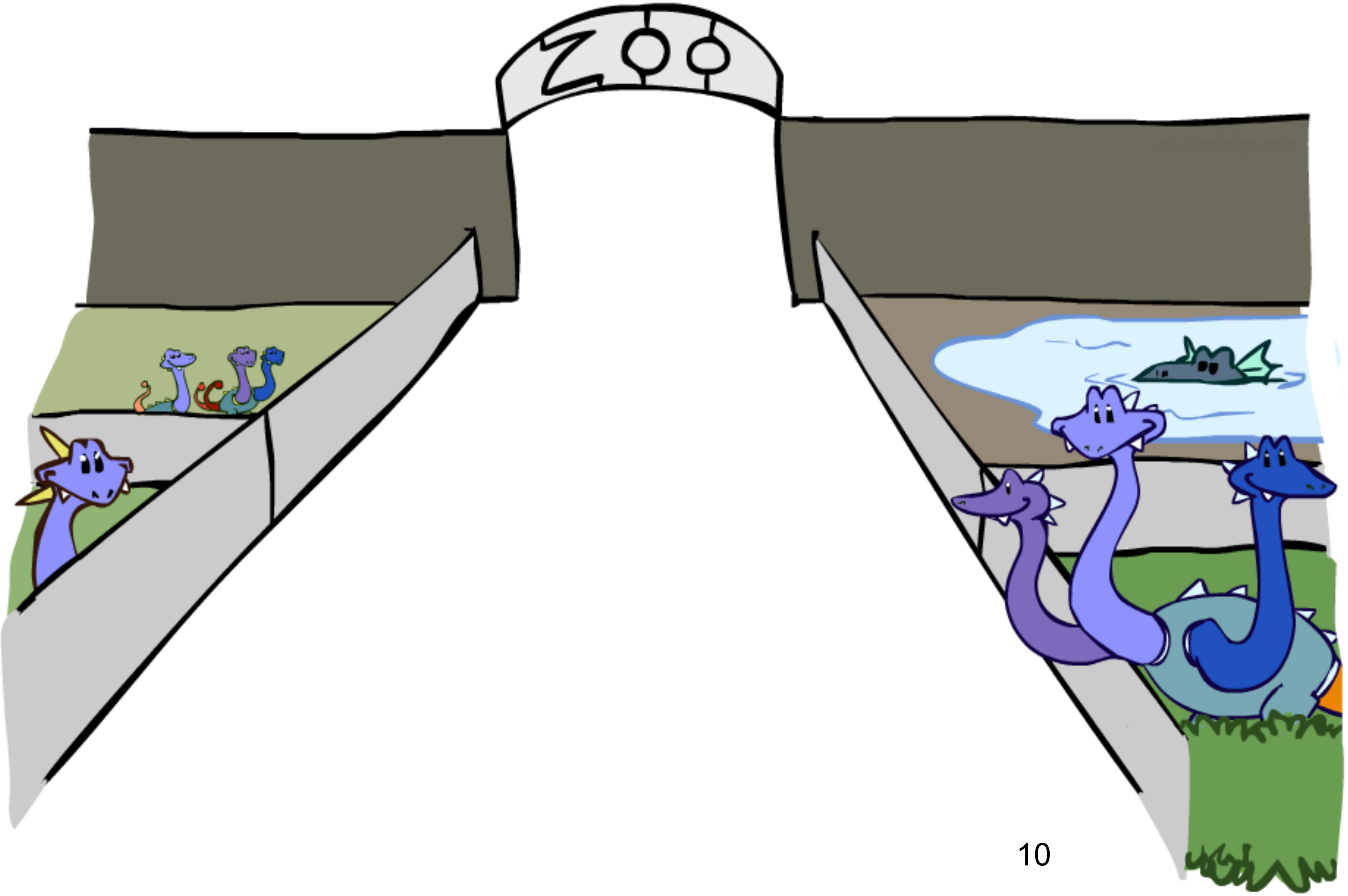
  Lots of repeated sub-expressions!

Variable elimination: The basic ideas

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



Factor Zoo



Factor Zoo I

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

$P(T, W)$

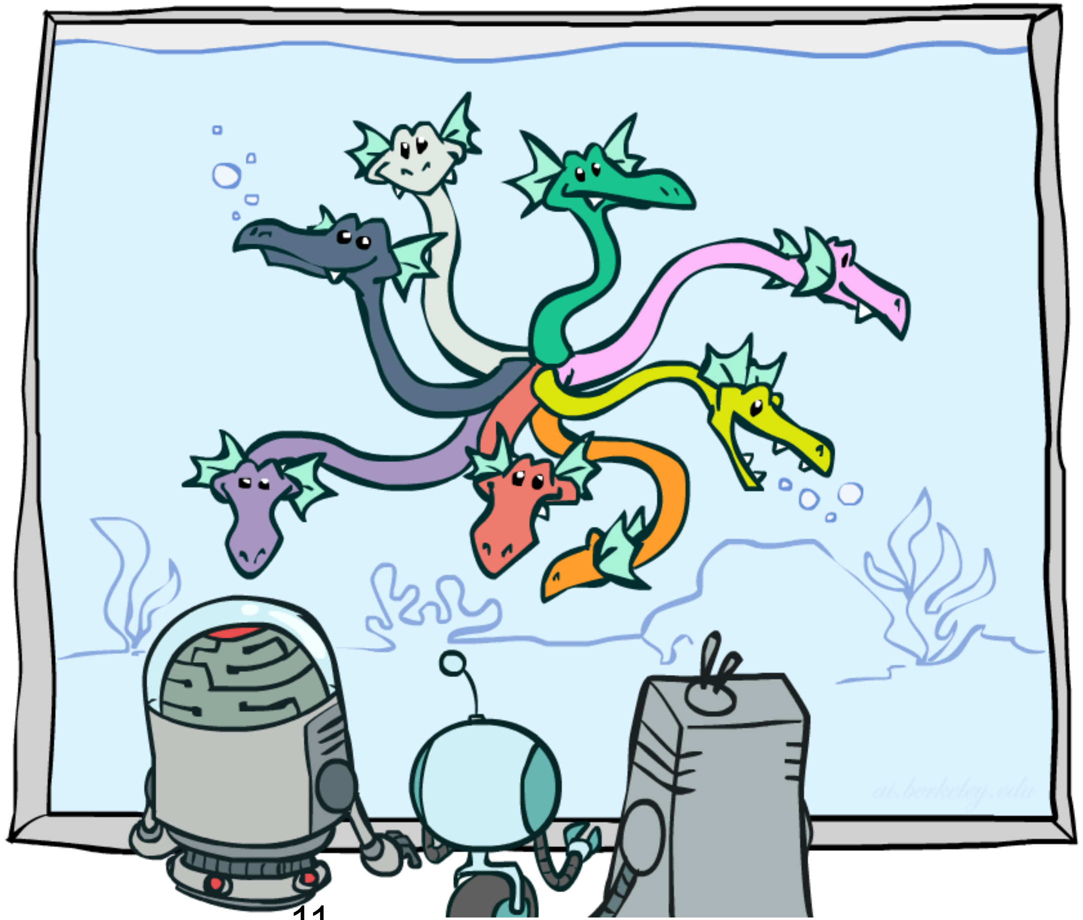
T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

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

$P(cold, W)$

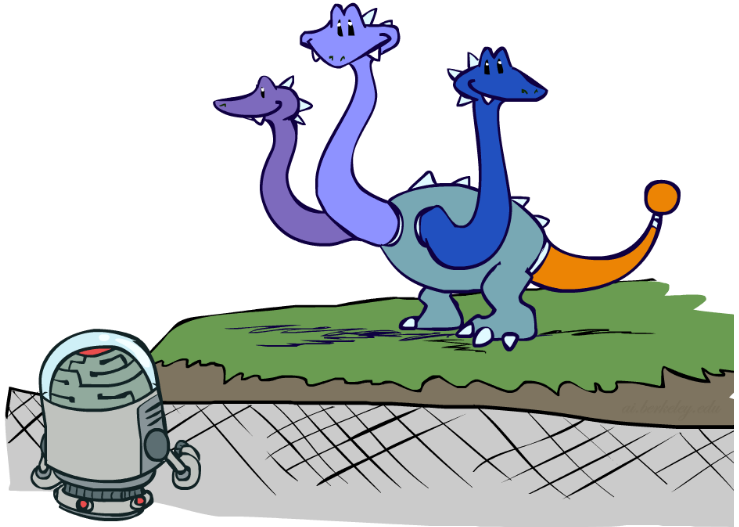
T	W	P
cold	sun	0.2
cold	rain	0.3

- Number of capitals = dimensionality of the table



Factor Zoo II

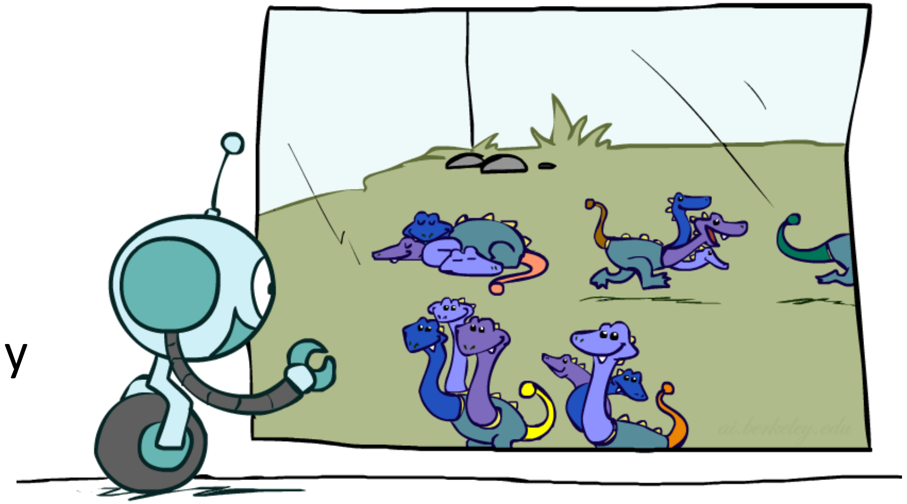
- Single conditional: P(Y | x)
  - Entries P(y | x) for fixed x, all
  - Sums to 1



$P(W|cold)$

T	W	P
cold	sun	0.4
cold	rain	0.6

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



$P(W|T)$

T	W	P
hot	sun	0.8
hot	rain	0.2
cold	sun	0.4
cold	rain	0.6

$\left.\begin{array}{l}\text{hot, sun, 0.8}\\ \text{hot, rain, 0.2}\end{array}\right\} P(W|hot)$

$\left.\begin{array}{l}\text{cold, sun, 0.4}\\ \text{cold, rain, 0.6}\end{array}\right\} P(W|cold)$

Factor Zoo III

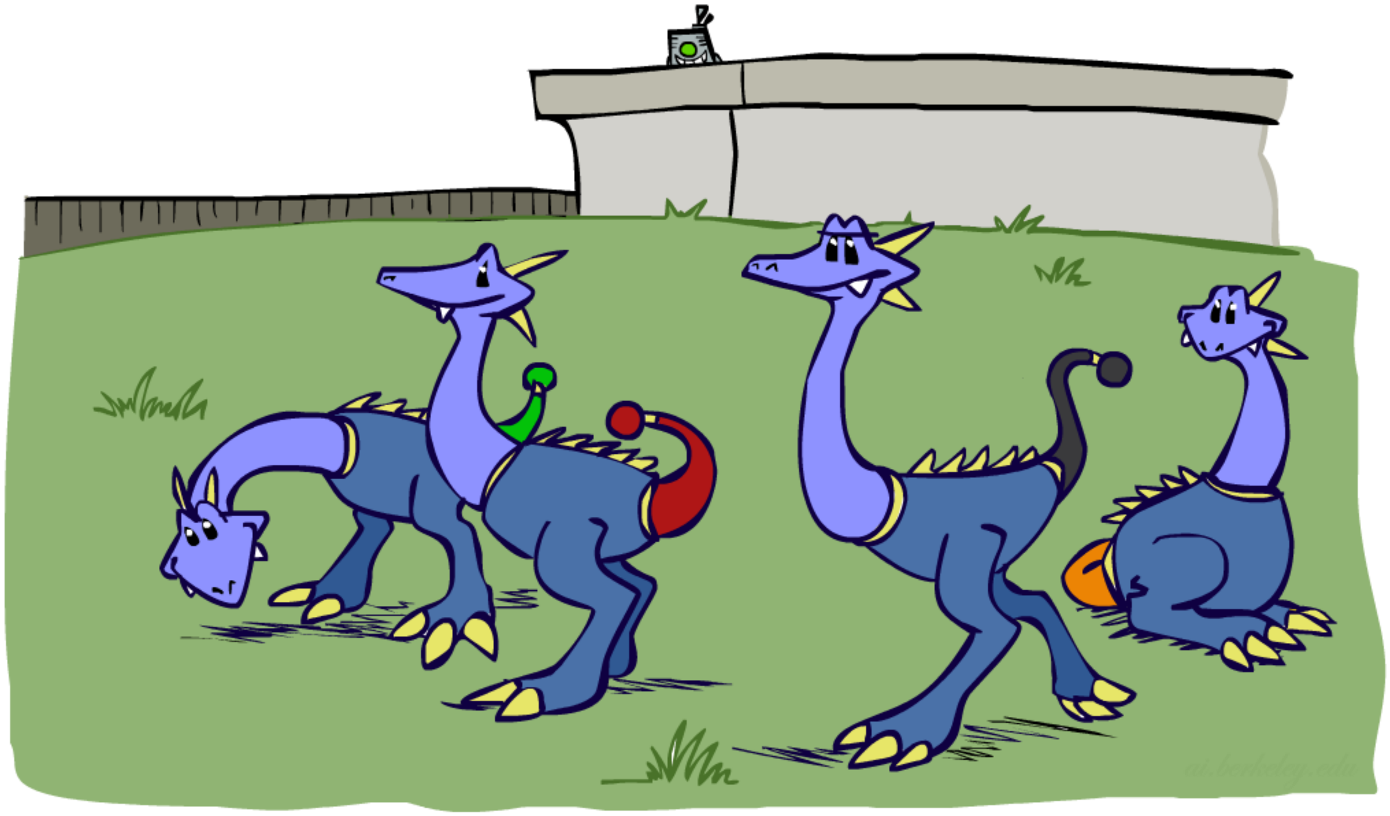
- Specified family: P(y | X)
  - Entries P(y | x) for fixed y, but for all x
  - Sums to … who knows!

$$P(rain|T)$$

T	W	P
hot	rain	0.2
cold	rain	0.6

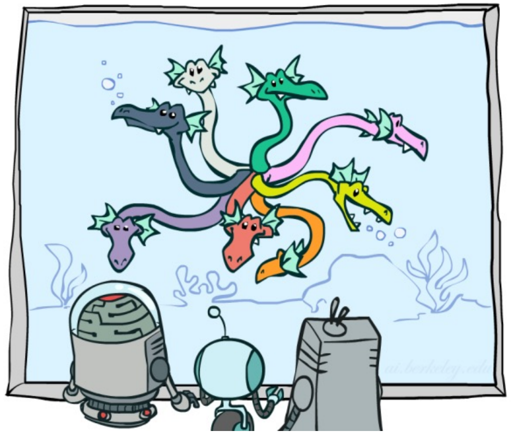
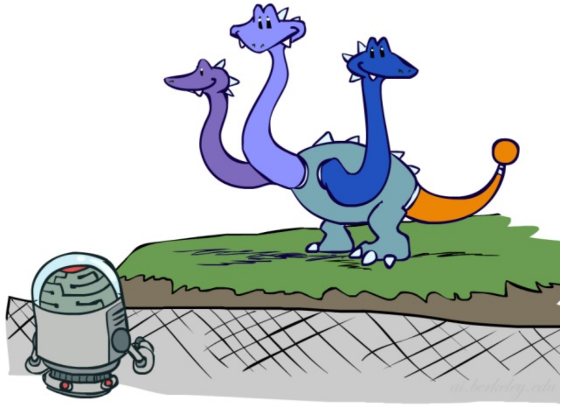
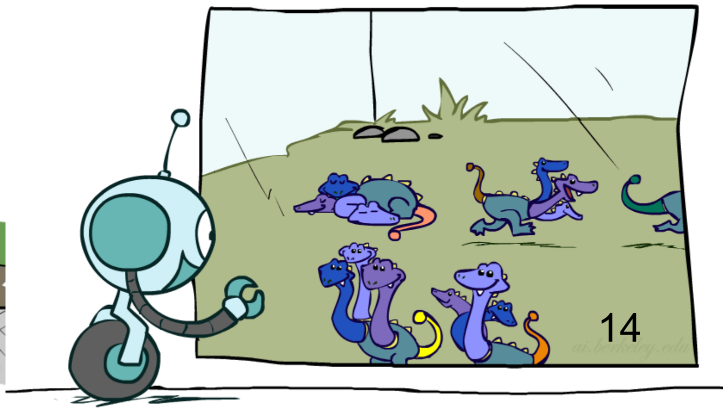
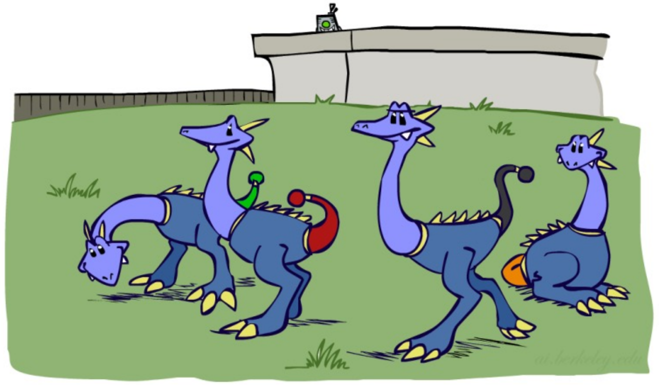
$P(rain|hot)$

$P(rain|cold)$



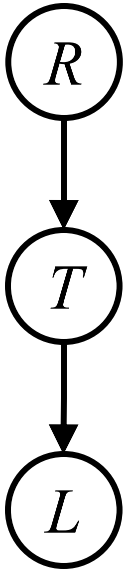
Factor Zoo Summary

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

Example: Traffic Domain

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



$P(R)$

+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)

$P(R)$

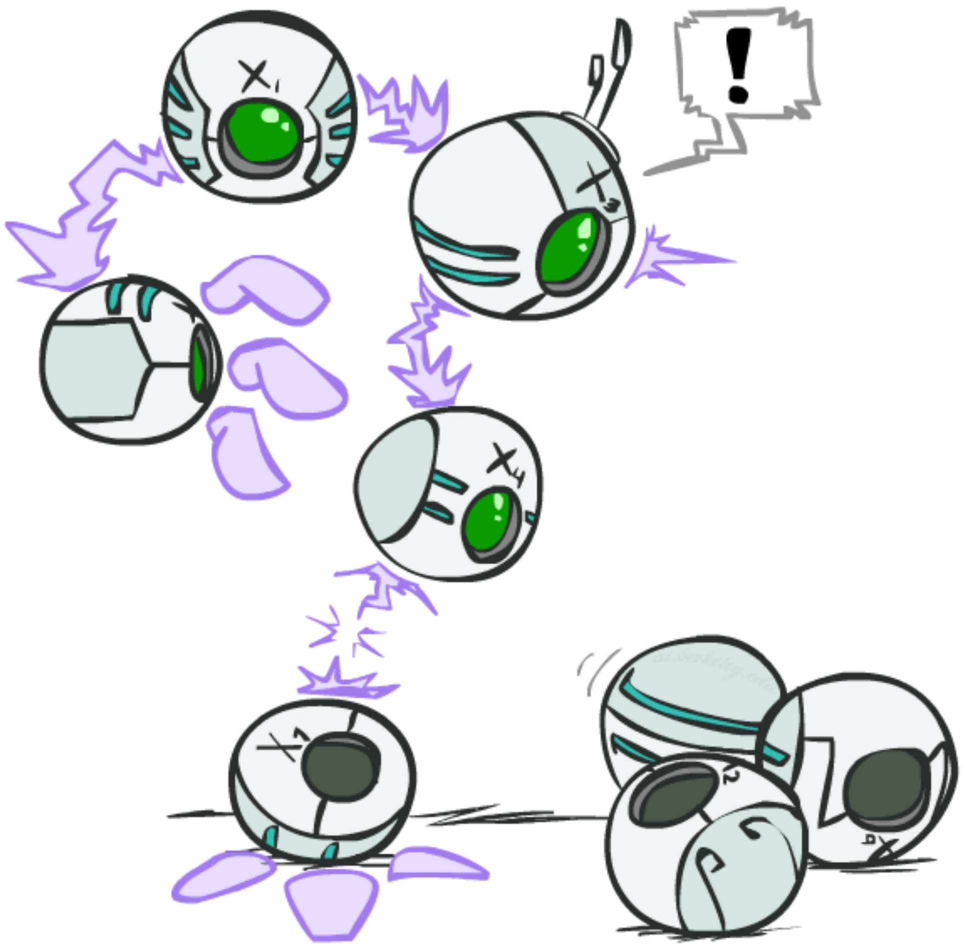
+r	0.1
-r	0.9

$P(T|R)$

+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

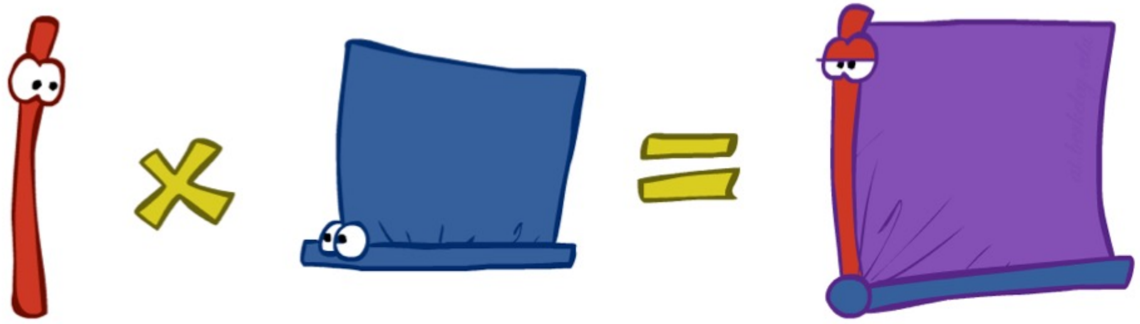
$P(+\ell|T)$

+t	+l	0.3
-t	+l	0.1

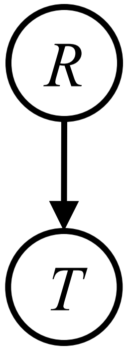
- Procedure: Join all factors, eliminate all hidden variables, normalize

Operation 1: Join Factors

- First basic operation: joining factors
  - Just like a database join
  - Get all factors over the joining variable
  - Build a new factor over the union of the variables involved
  - Each entry is the product of the corresponding entries



- Example: Join on R

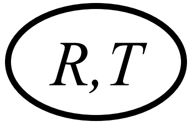


$$P(R) \quad \times \quad P(T|R) \quad \Rightarrow \quad P(R,T)$$

+r	0.1
-r	0.9

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

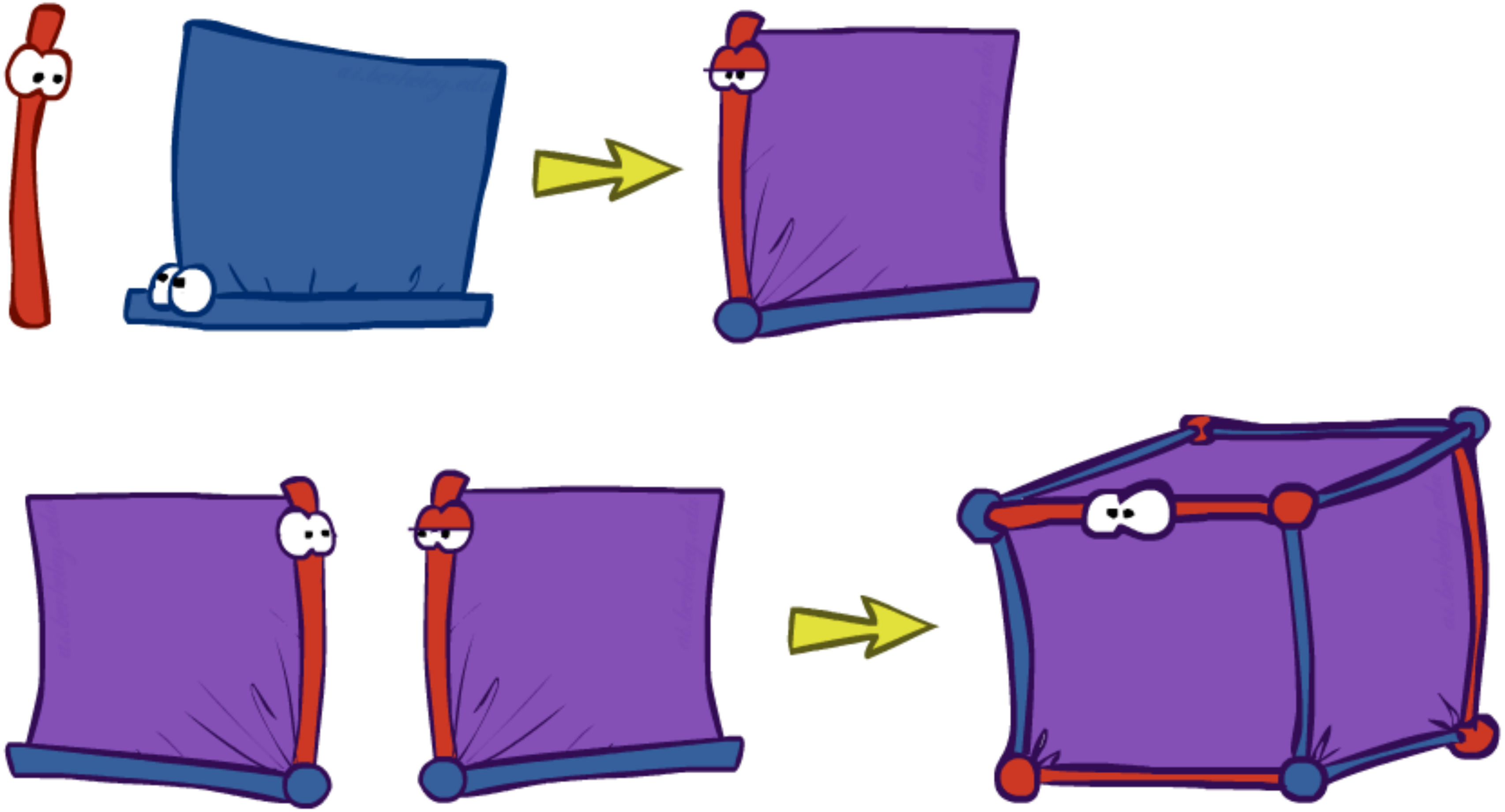
+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81



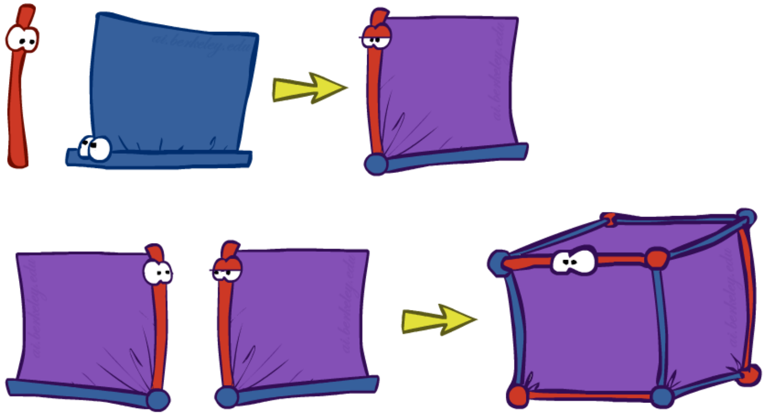
$$\forall r,t: \qquad P(r,t) = P(r) \cdot P(t|r)$$

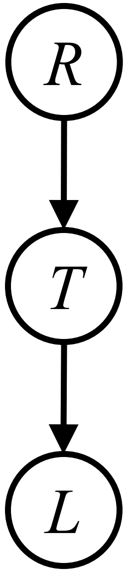
- Computation for each entry: pointwise products

Example: Multiple Joins



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

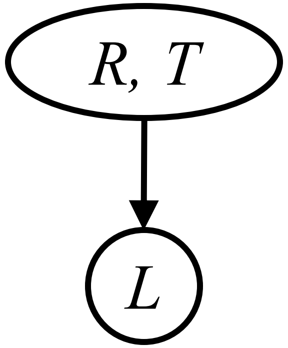
Join R

$P(R,T)$

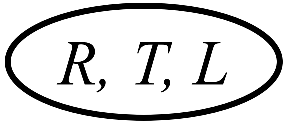
+r	+t	0.08
+r	-t	0.02
-r	+t	0.09
-r	-t	0.81

$P(L|T)$

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



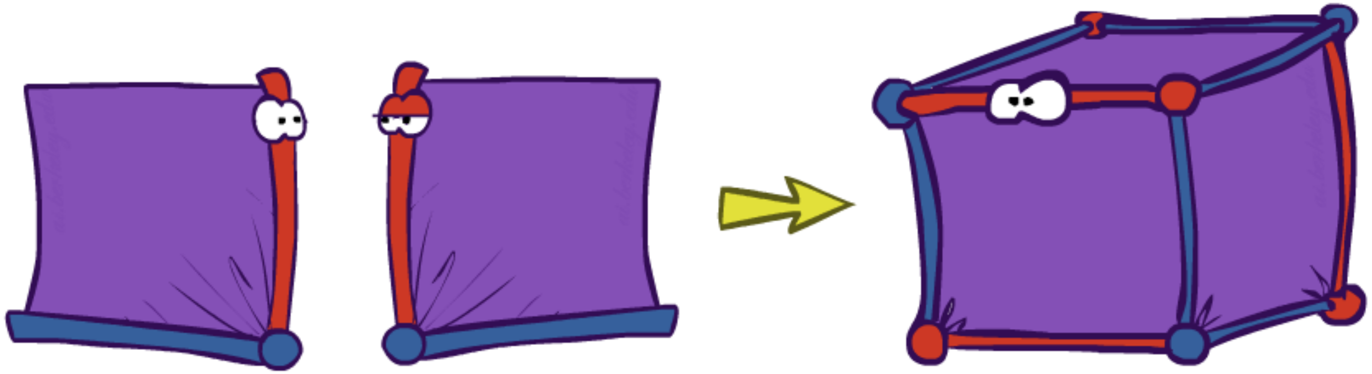
Join T



$P(R,T,L)$

+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: Making larger factors



- Example: $P(U,V) \times P(V,W) \times P(W,X) = P(U,V,W,X)$
- 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 VE 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:

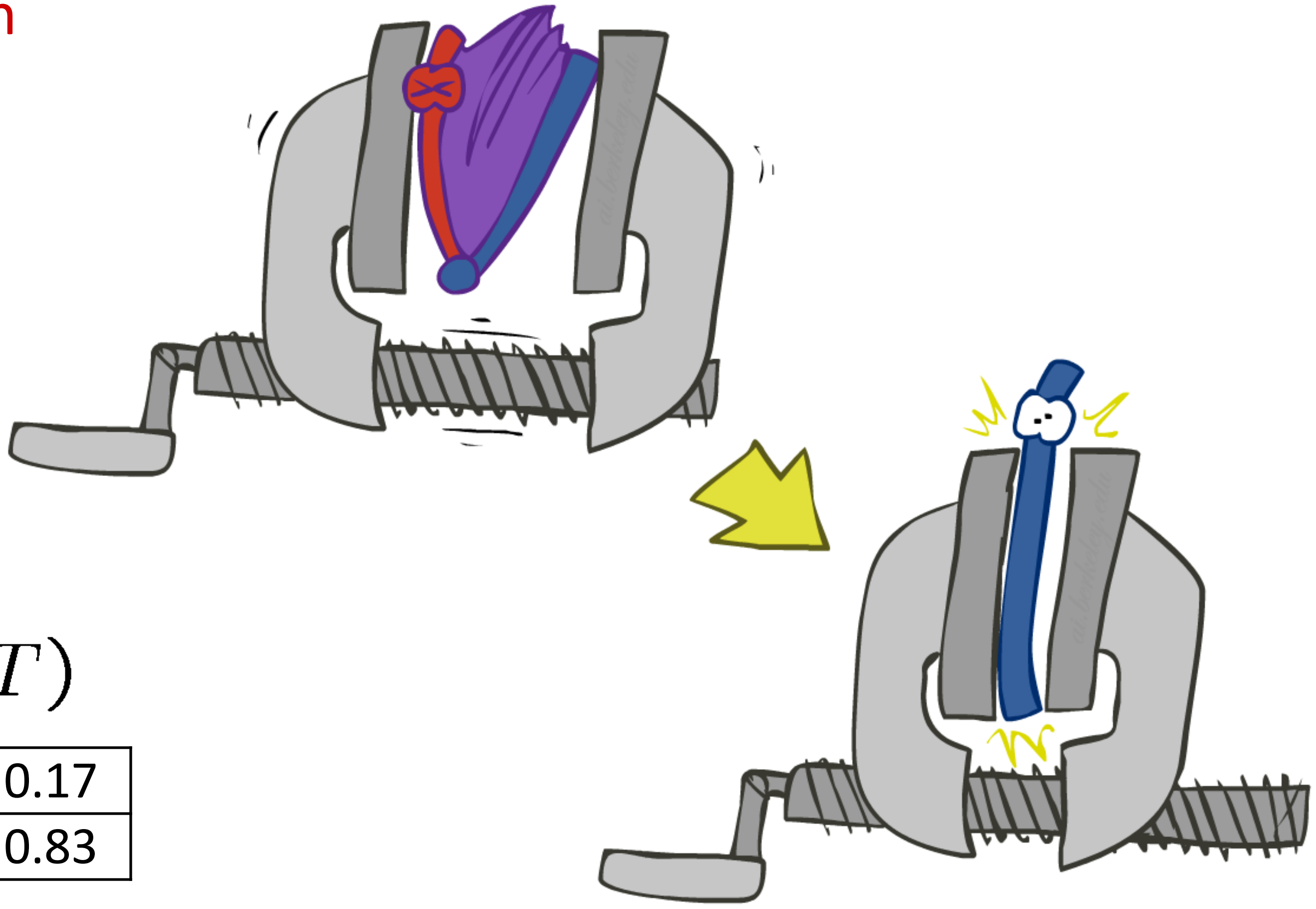
$P(R,T)$

+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 | T, L | 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

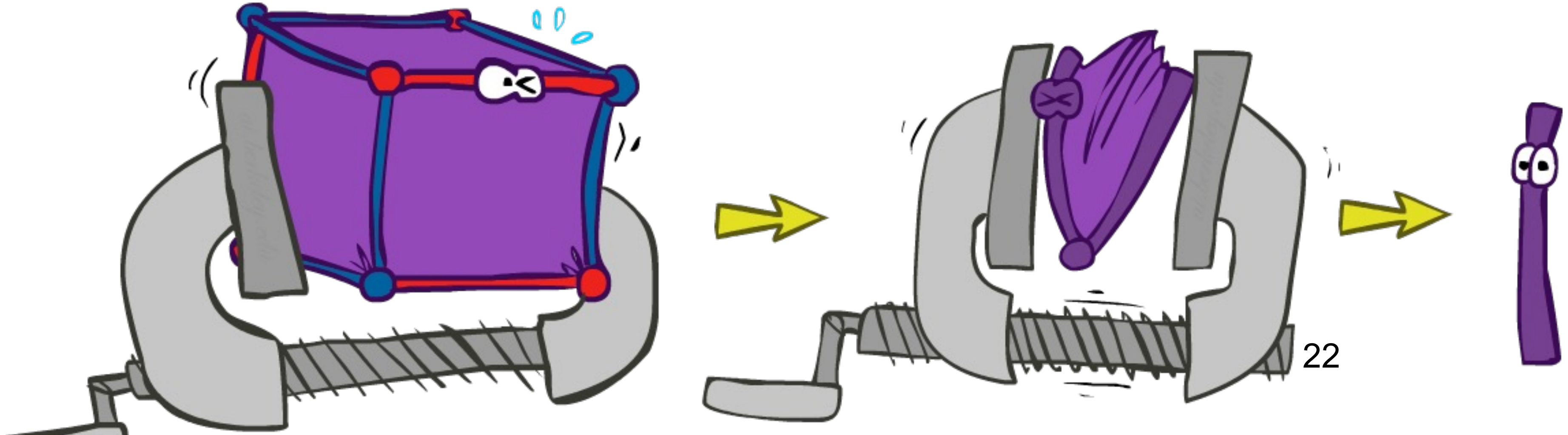
$P(T,L)$

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

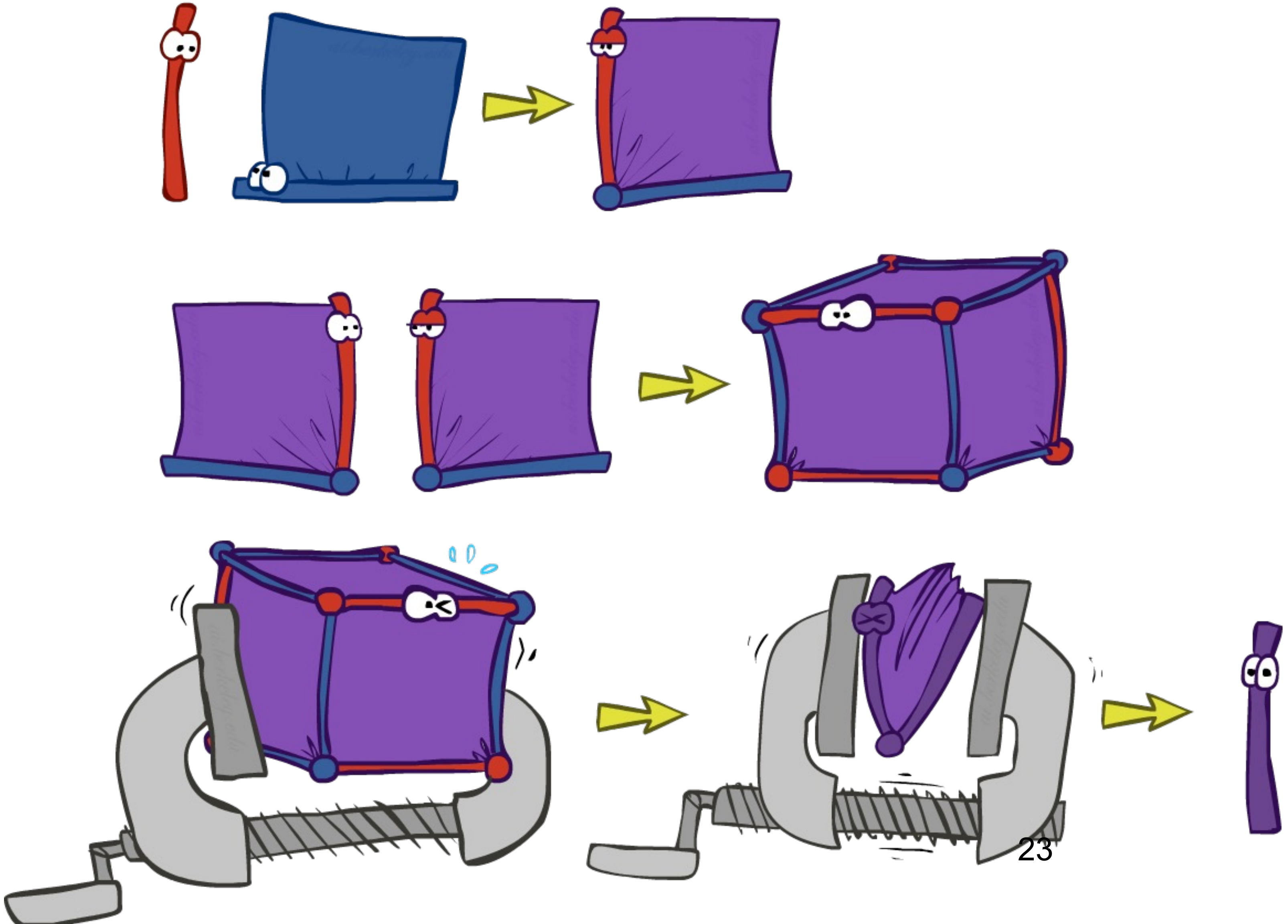
Sum out T

$P(L)$

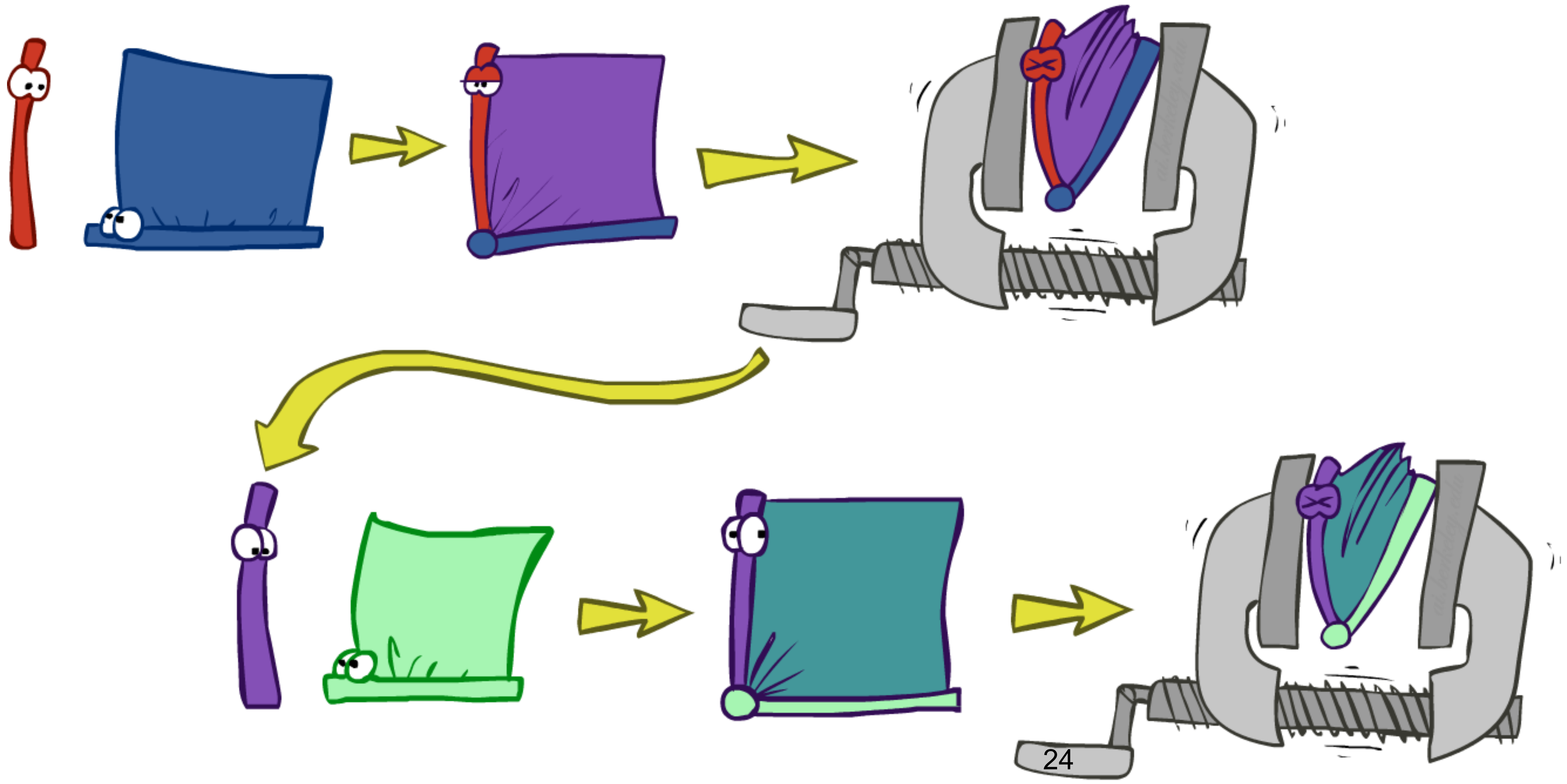
+l	0.134
-l	0.886



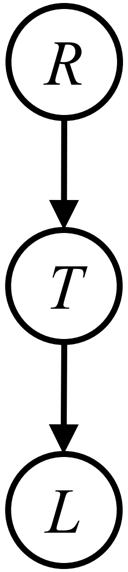
Thus Far: Multiple Join, Multiple Eliminate (= Inference by Enumeration)



Variable Elimination



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

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

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:

$P(+r)$

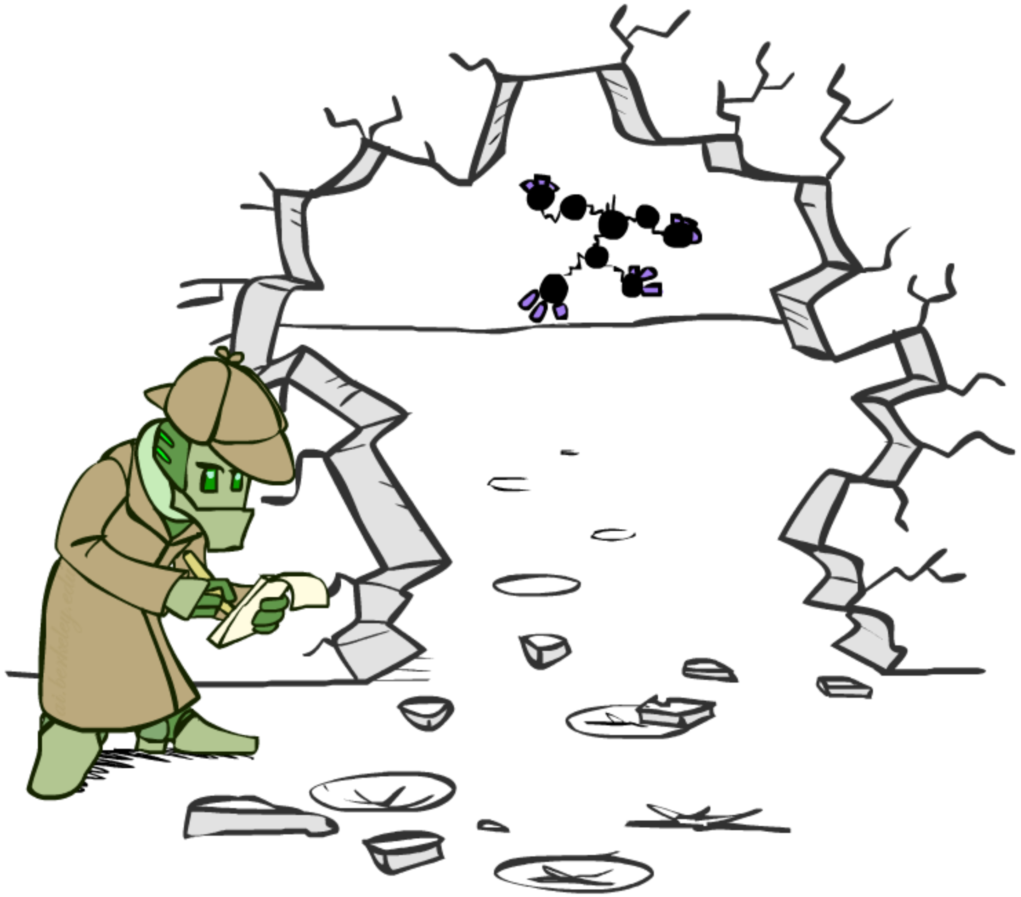
+r	0.1

$P(T| +r)$

+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:

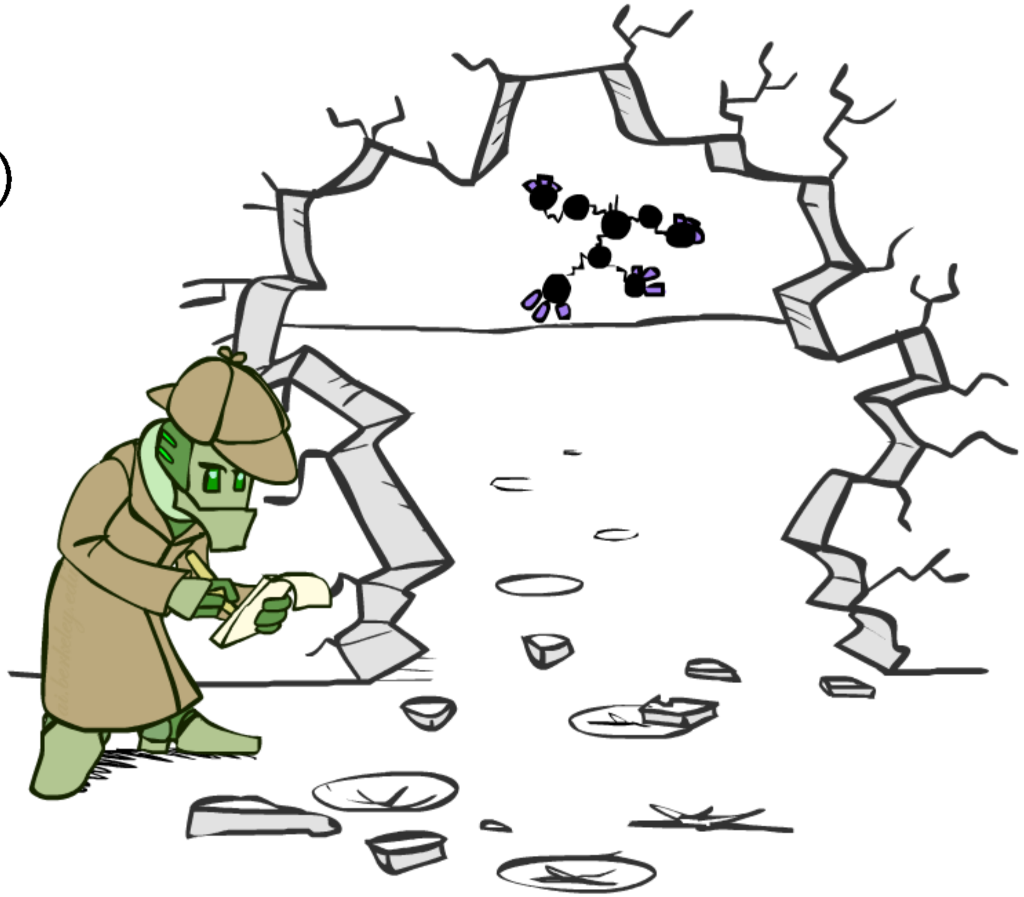
$P(+r, L)$

+r	+l	0.026
+r	-l	0.074

Normalize

$P(L | +r)$

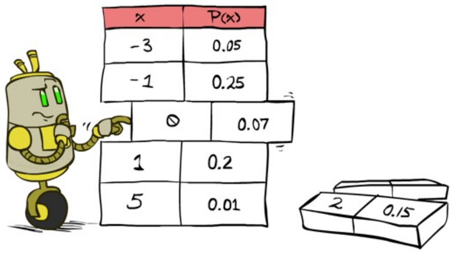
+l	0.26
-l	0.74



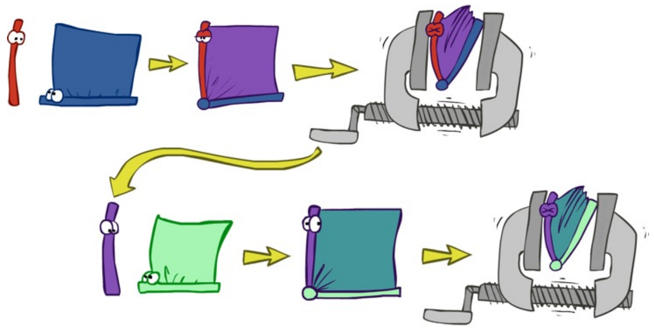
- To get our answer, just normalize this!
- That's it!

General Variable Elimination

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

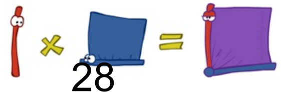


- Start with initial factors:
  - Local CPTs (but instantiated by evidence)



- While there are still hidden variables (not Q or evidence):
  - Pick a hidden variable H
  - Join all factors mentioning H
  - Eliminate (sum out) H

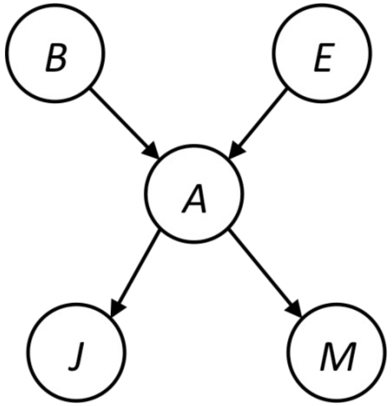
- Join all remaining factors and normalize



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

Example

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



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

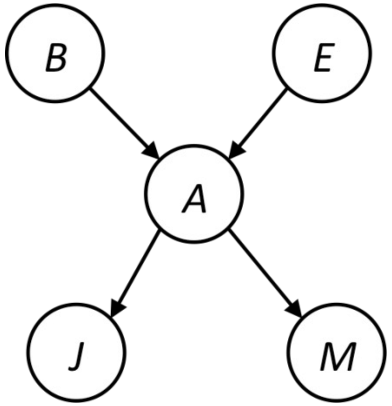
Choose A

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

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

Example

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



Choose E

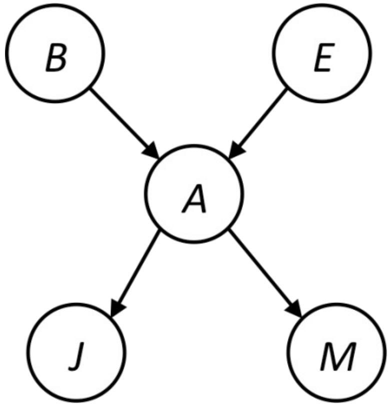
$$P(E) \quad P(j,m|B,E) \xrightarrow{\times} P(j,m,E|B) \xrightarrow{\sum} P(j,m|B)$$

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

Finish with B

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

Same Example in Equations



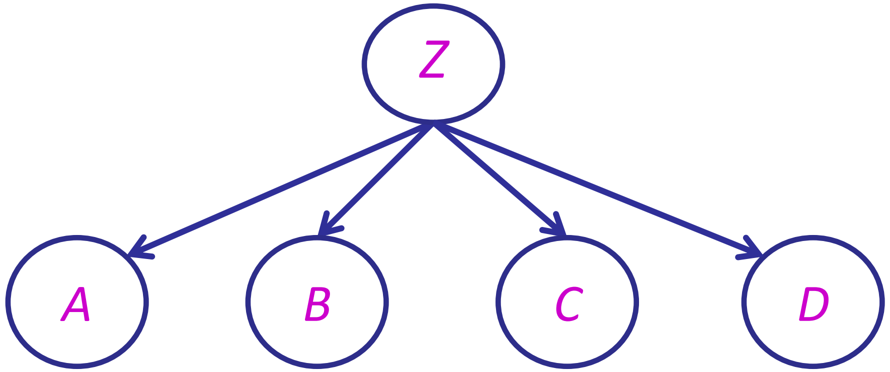
$$P(B|j,m) \propto P(B,j,m)$$

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

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

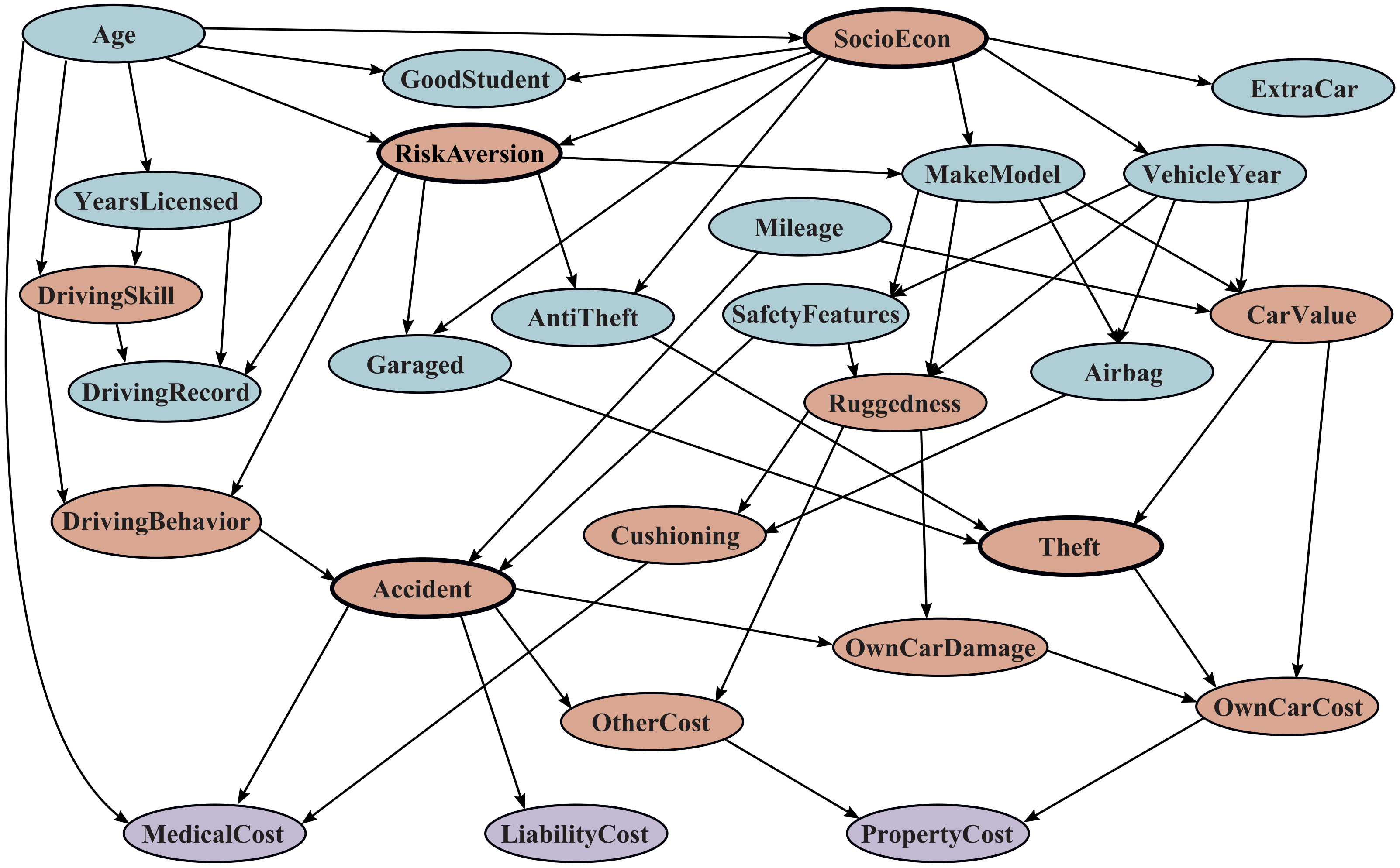
All we are doing is exploiting uwy + uwz + uxy + uxz + vwy + vwz + vxy +vxz = (u+v)(w+x)(y+z) to improve computational efficiency!

Order matters



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

Example Bayes' Net: Car Insurance



Enumeration: **227M** operations

Elimination: **221K** operations

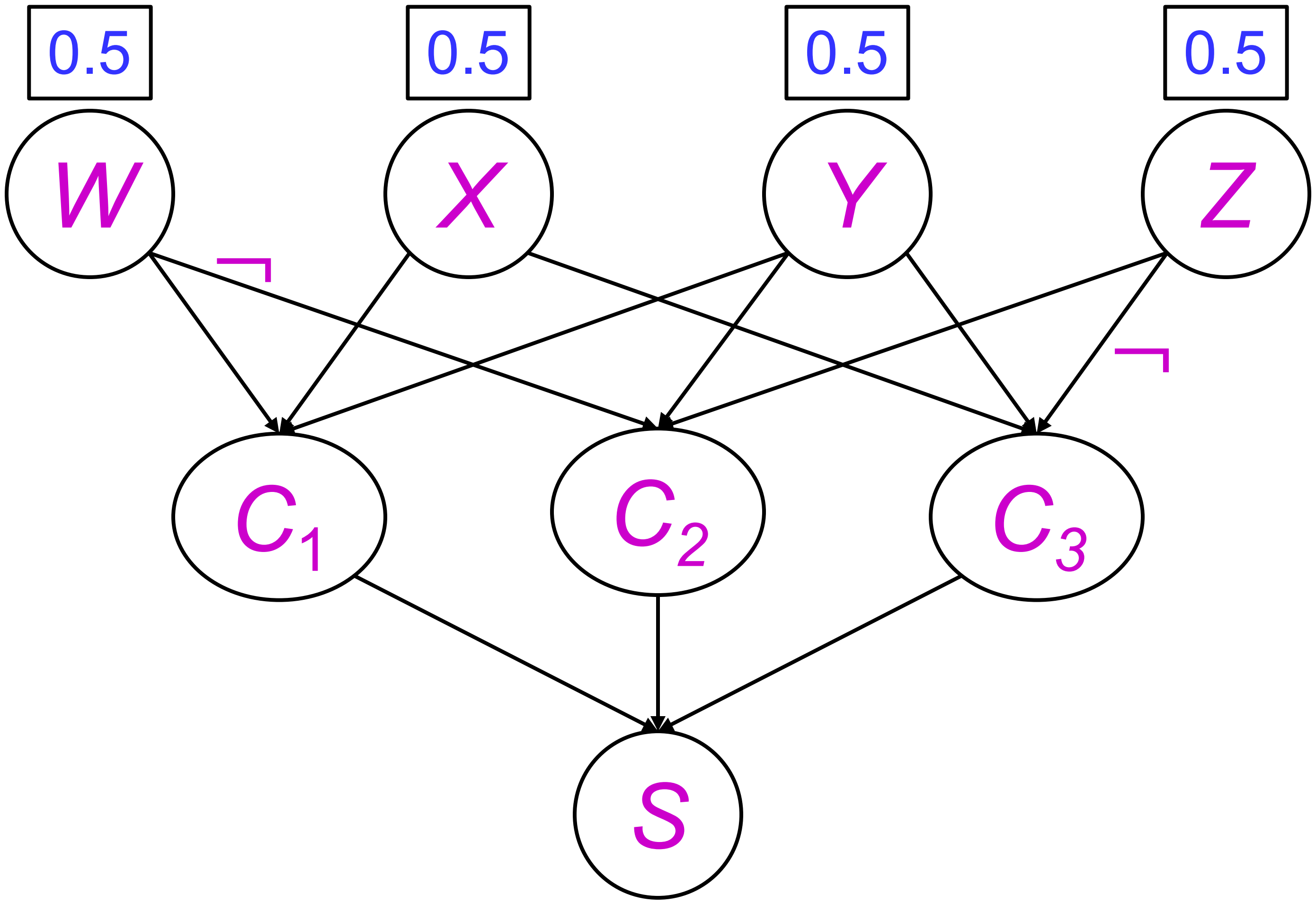
VE: Computational and Space Complexity

- 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!

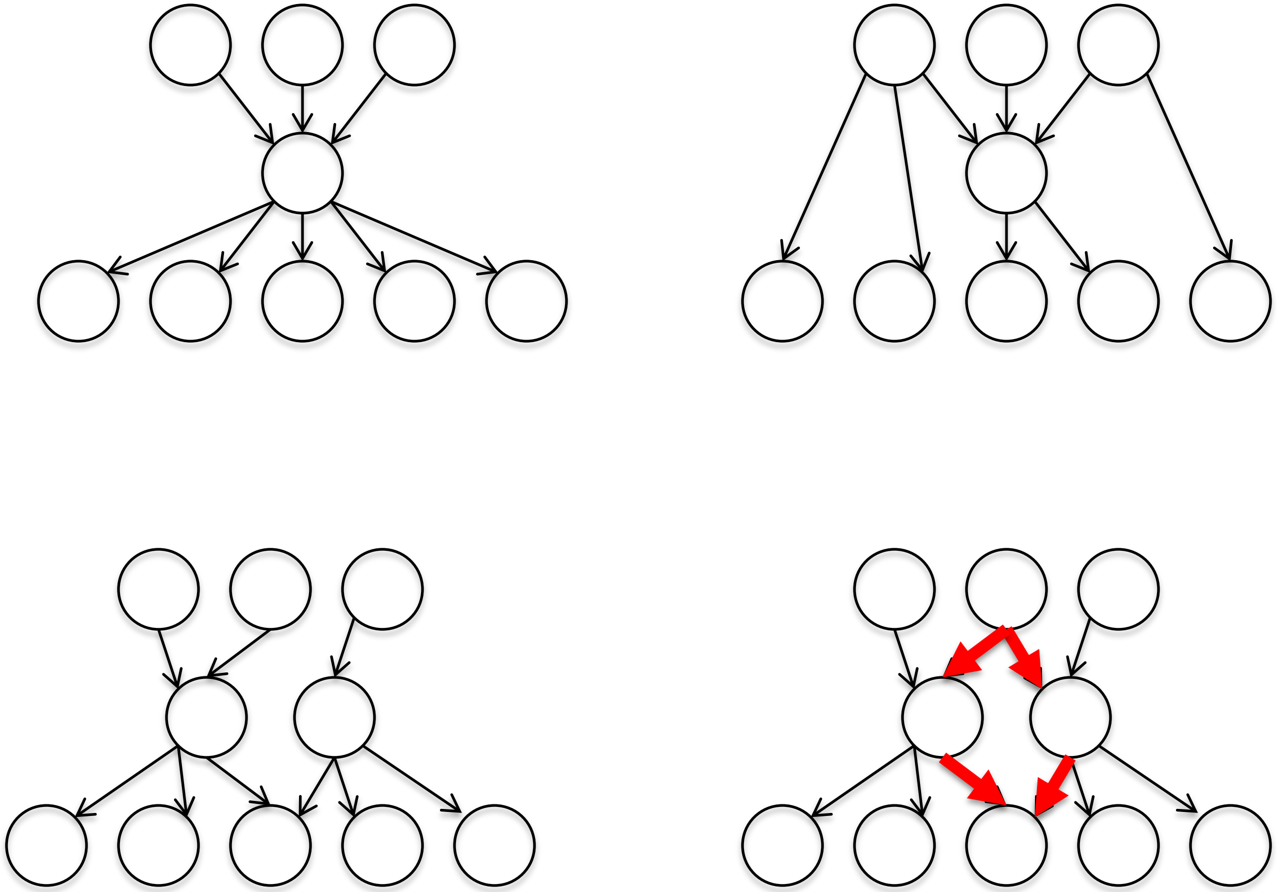
Worst Case Complexity? Reduction from SAT



- Variables: *W, X, Y, Z*
- CNF clauses:
  1. $C_1 = W \vee X \vee Y$
  2. $C_2 = Y \vee Z \vee \neg W$
  3. $C_3 = X \vee Y \vee \neg Z$
- 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 leave towards the roots



Summary

- Exact inference = sums of products of conditional probabilities from the network
- Enumeration is always exponential
- Variable elimination reduces this by avoiding the recomputation of repeated subexpressions
  - Massive speedups in practice
  - Linear time for polytrees
- Exact inference is #P-hard
- Next: approximate inference