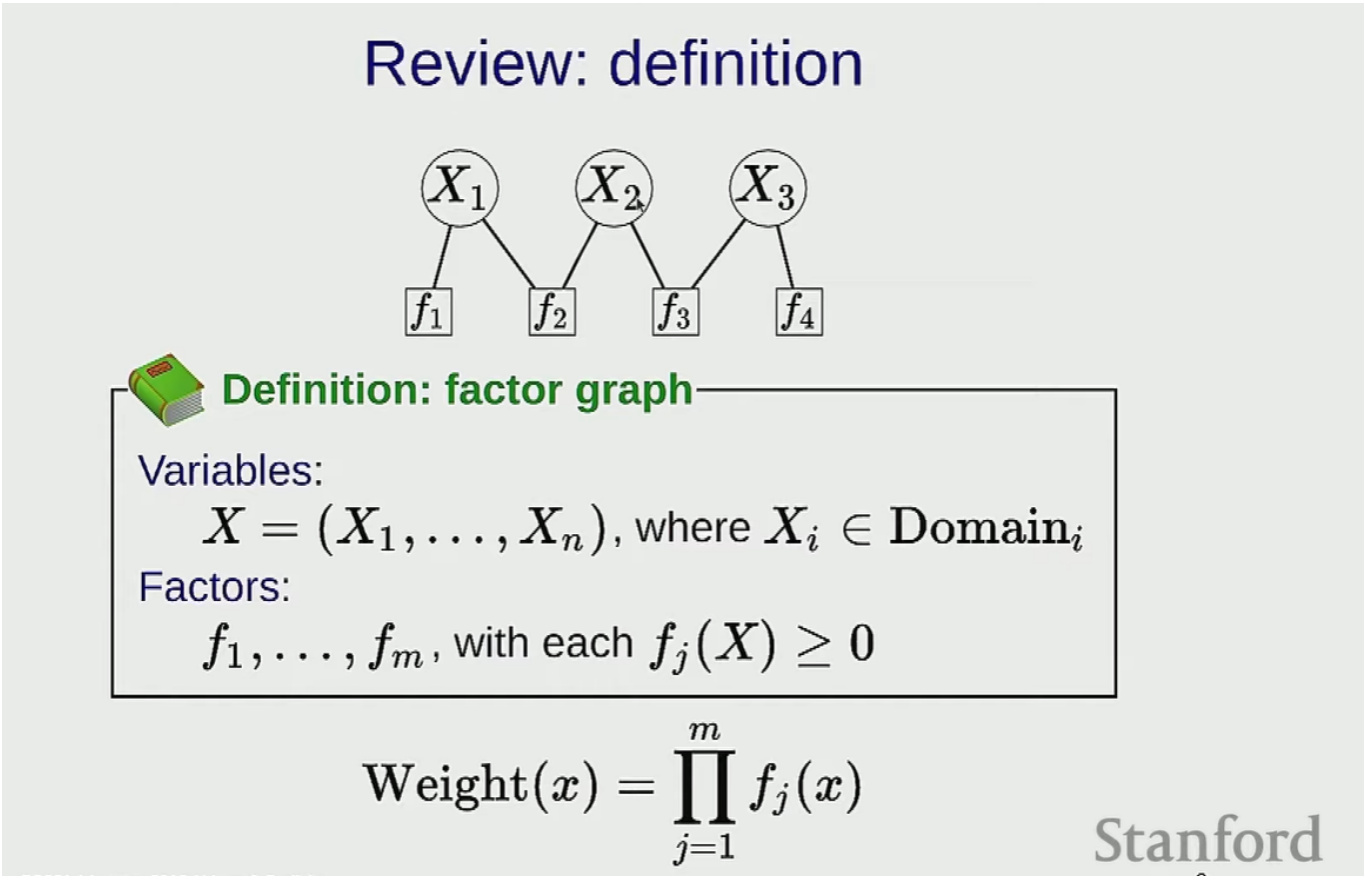
Bayesian Networks 1 - Inference

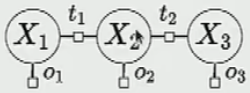


Specify locally (Factors) and optimize globally (Weight) Last time we talked about different algo for finding the maximum weight

Review: object tracking

Problem: object tracking

Sensors report positions: $0, 2, 2$. Objects don't move very fast and sensors are a bit noisy. What path did the object take?



- Variables X_i: location of object at time i
- Transition factors $t_i(x_i, x_{i+1})$: incorporate physics
- Observation factors $o_i(x_i)$: incorporate sensors

[demo: `maxVariableElimination()`]

Stanford

CS221 / Autumn 2019 / Liang & Sadigh 4

The framework is already good, with this you can come up with a lot But what are these factors mean and how do we come up with them? philosophically we might be quite bothered by this

So the goal of this lecture is to give more meaning to the factors, and bayesian network is the way to do that. In a line, bayasian networks are factor graphs plus probability

Previously we already talked about a lot of modeling: search games, MDPs And then we talked about modeling when the order of actions does not matter so much, and it is more nature to think it as variables and assignment And now we will move on to bayasian networks where it will be a higher level of abstraction

Basics

Review: probability

Random variables: sunshine $S \in \{0, 1\}$, rain $R \in \{0, 1\}$

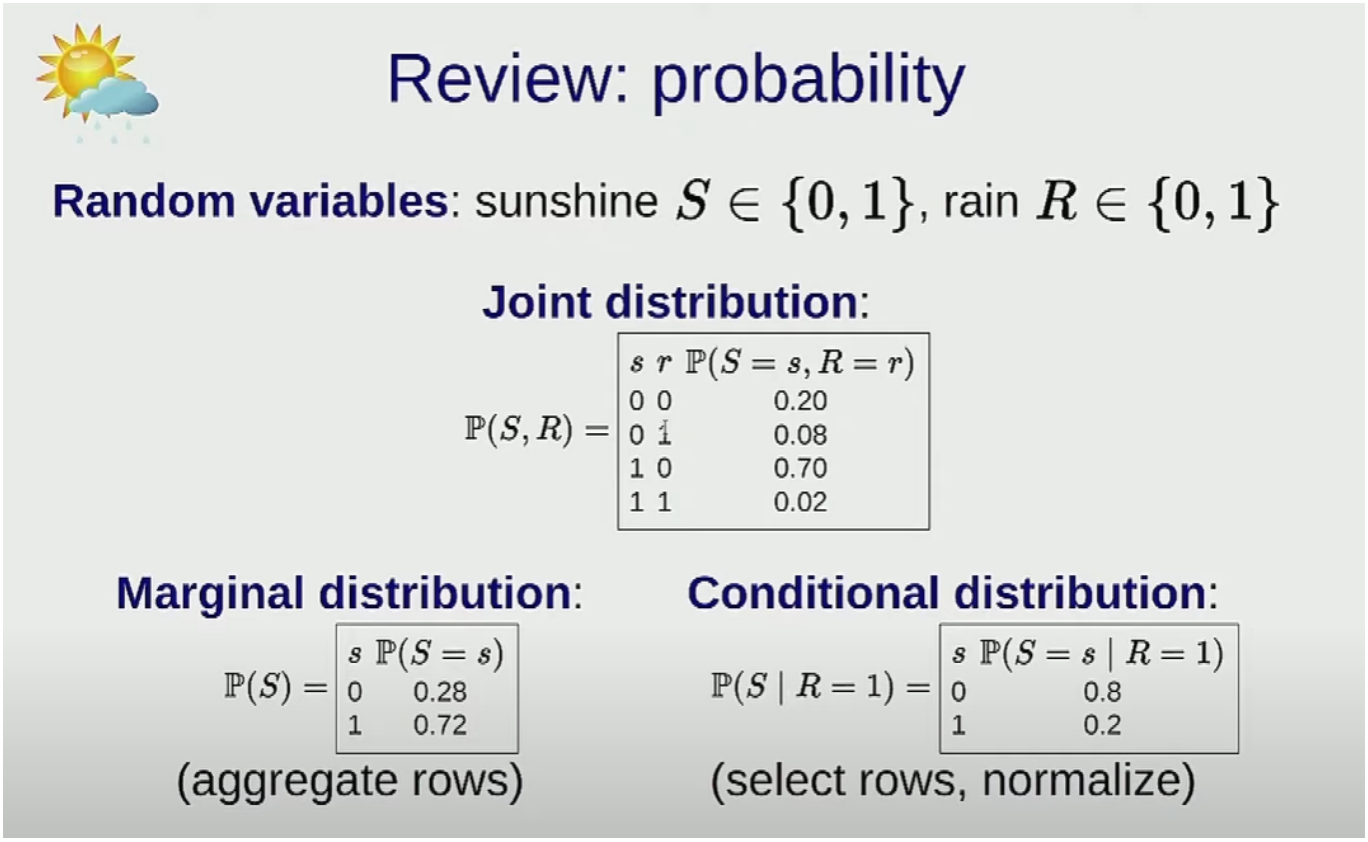
Joint distribution:

$\mathbb{P}(S, R) =$

s	r	$\mathbb{P}(S = s, R = r)$
0	0	0.20
0	1	0.08
1	0	0.70
1	1	0.02

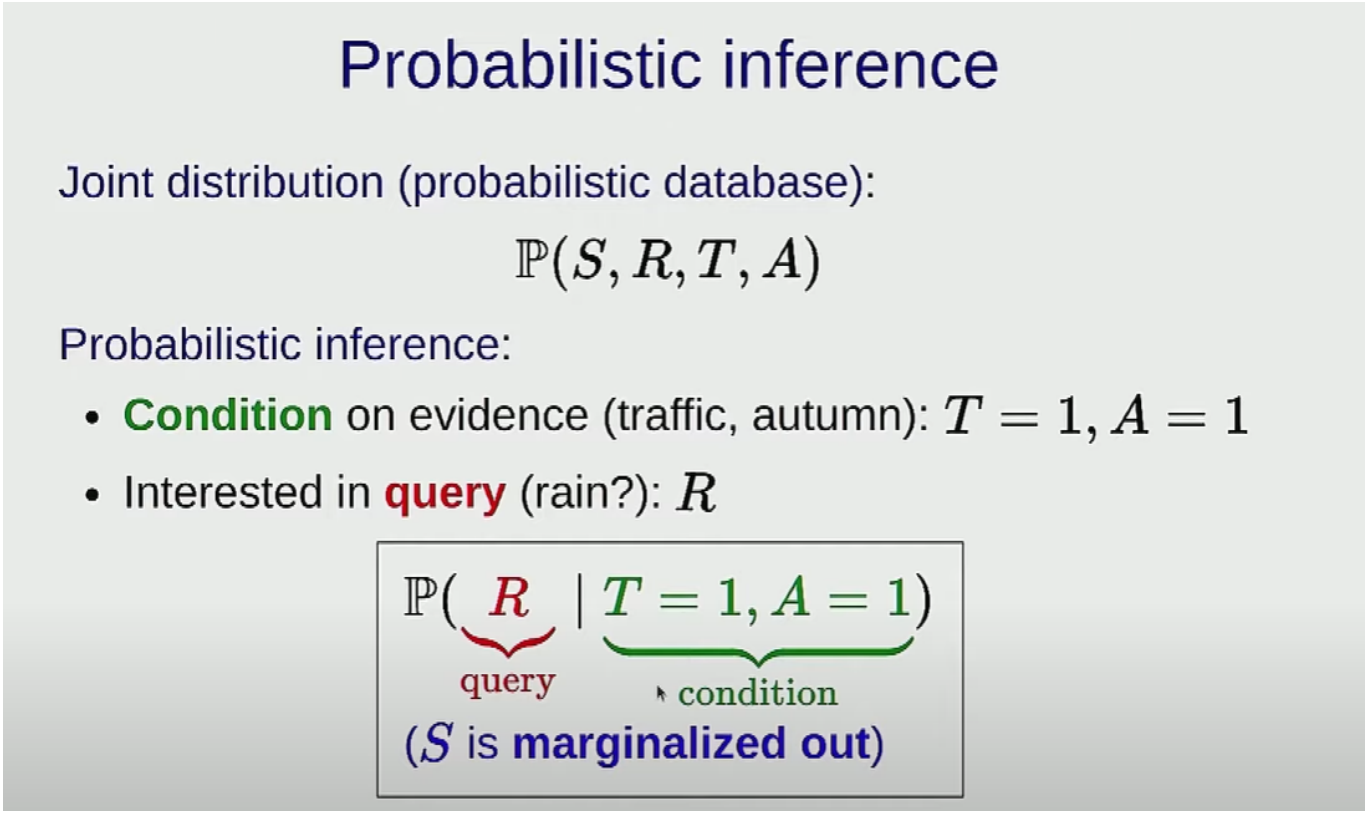
Notations; Captital letters (S, R): random variables samller letters (s,r): values that random variables can take

Notation P with Captital letters P(S, R): the whole probility destributions, eg P(S, R) is the destribution table
Notation P with =s, =r assignments P(S=s,, R=r): represents a signle number, which is a probility, eg: P(S=1, R=0) = 0.7



Marginal distribution: eg I don't care about r, just wanna focus on s > Sum up probs when s = 0, s = 1 and u get the table

Conditional: Select rows based on condition, normalize them and let them sum to 1. normalize (a,b) means a' = a/(a+b), b' = b/(a+b)

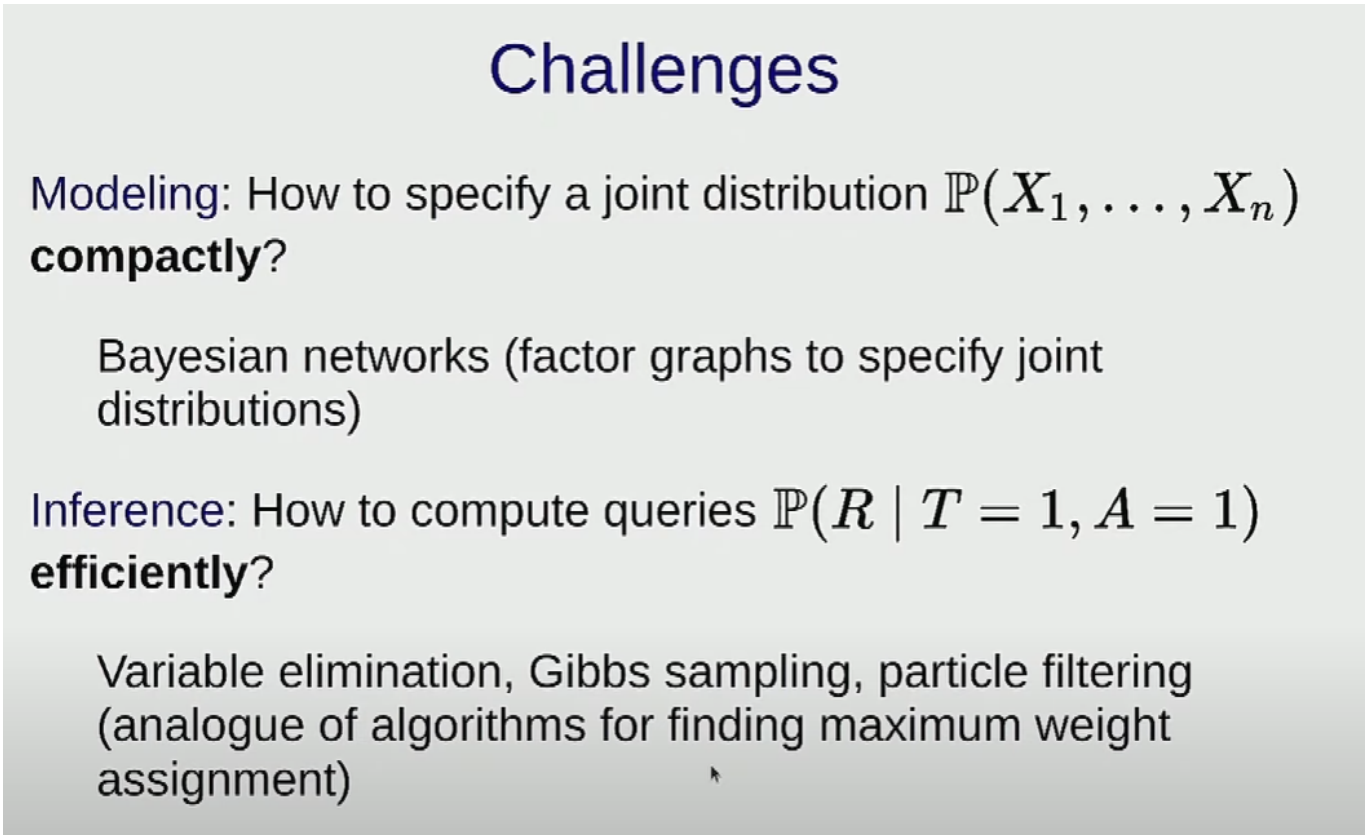


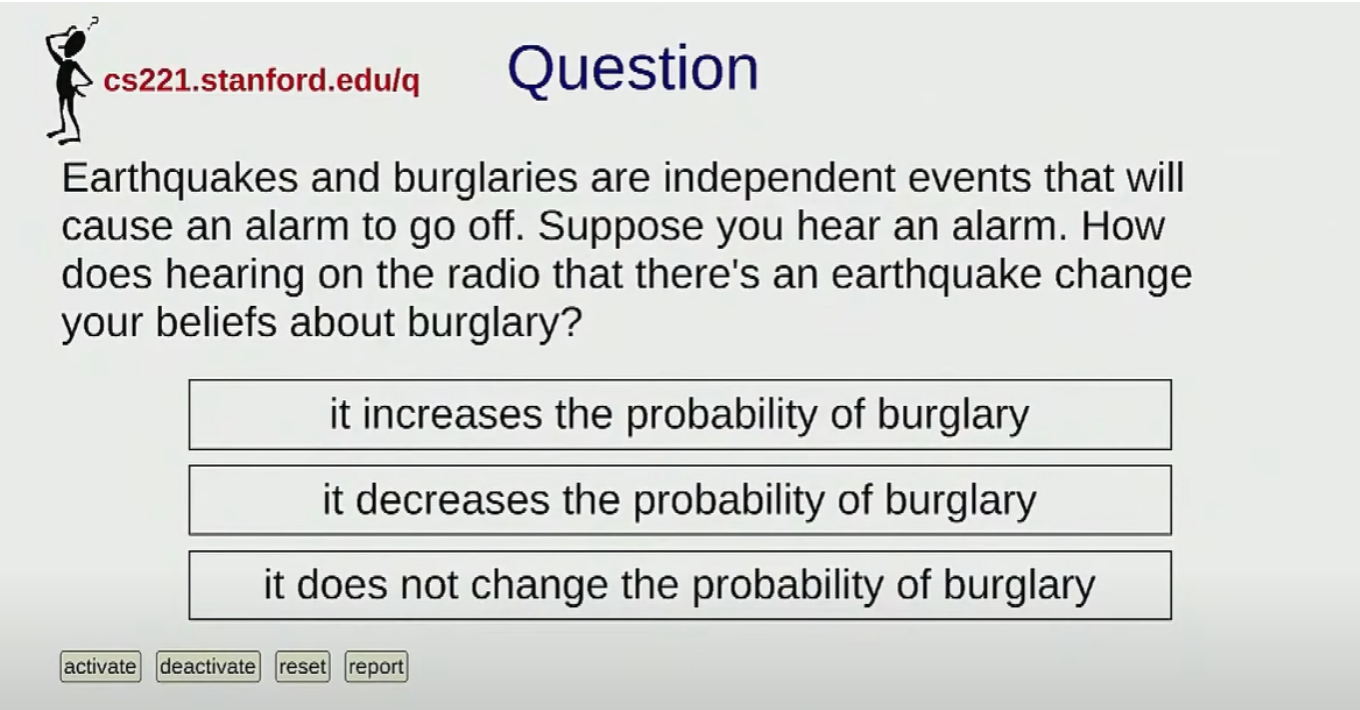
Think of joint distibution as a database

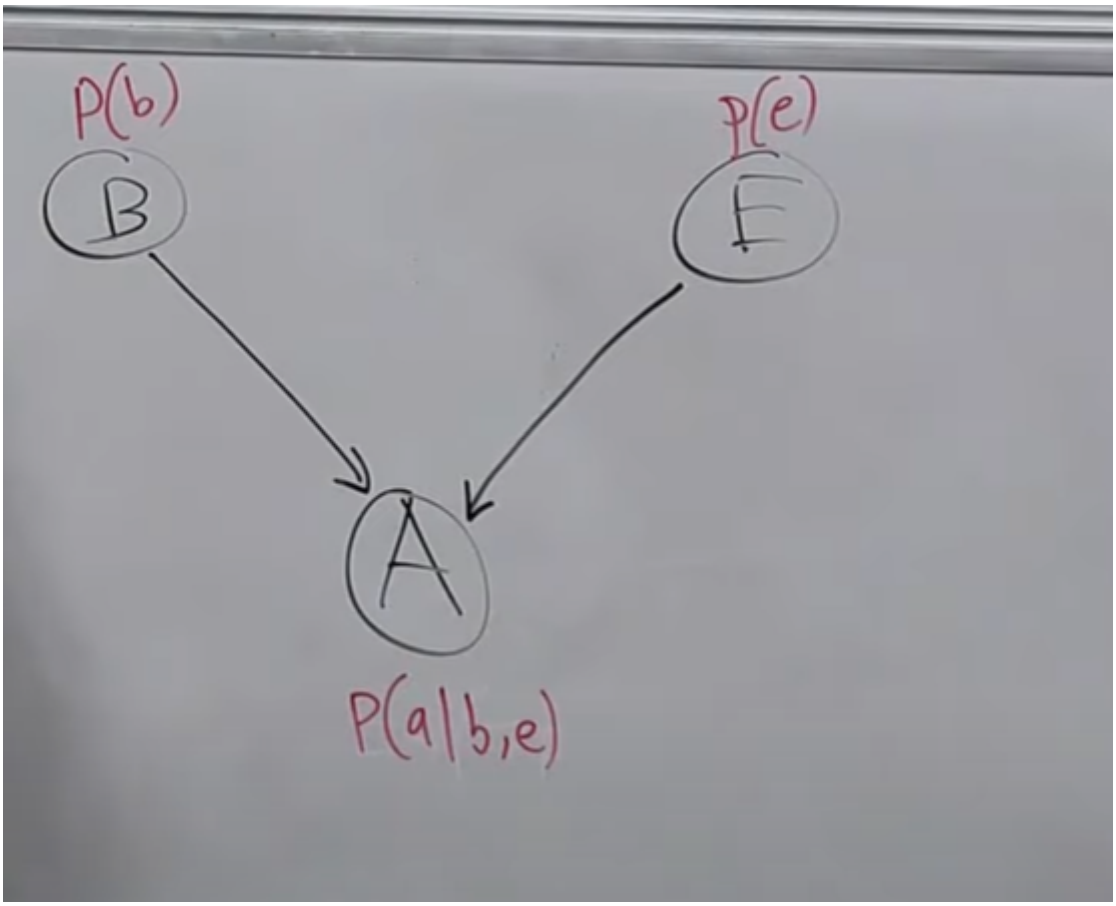
Probabilistic inference: You observe some evidence, that's what we know and what we like to find out is weather it's raining

Is that reaning under condition of we know it is autumn and traffic

Challenges: the joint distibution table can he really huge Cool thing about bayasian networks is that it allows us to define joint distibution using the language of factor graphs





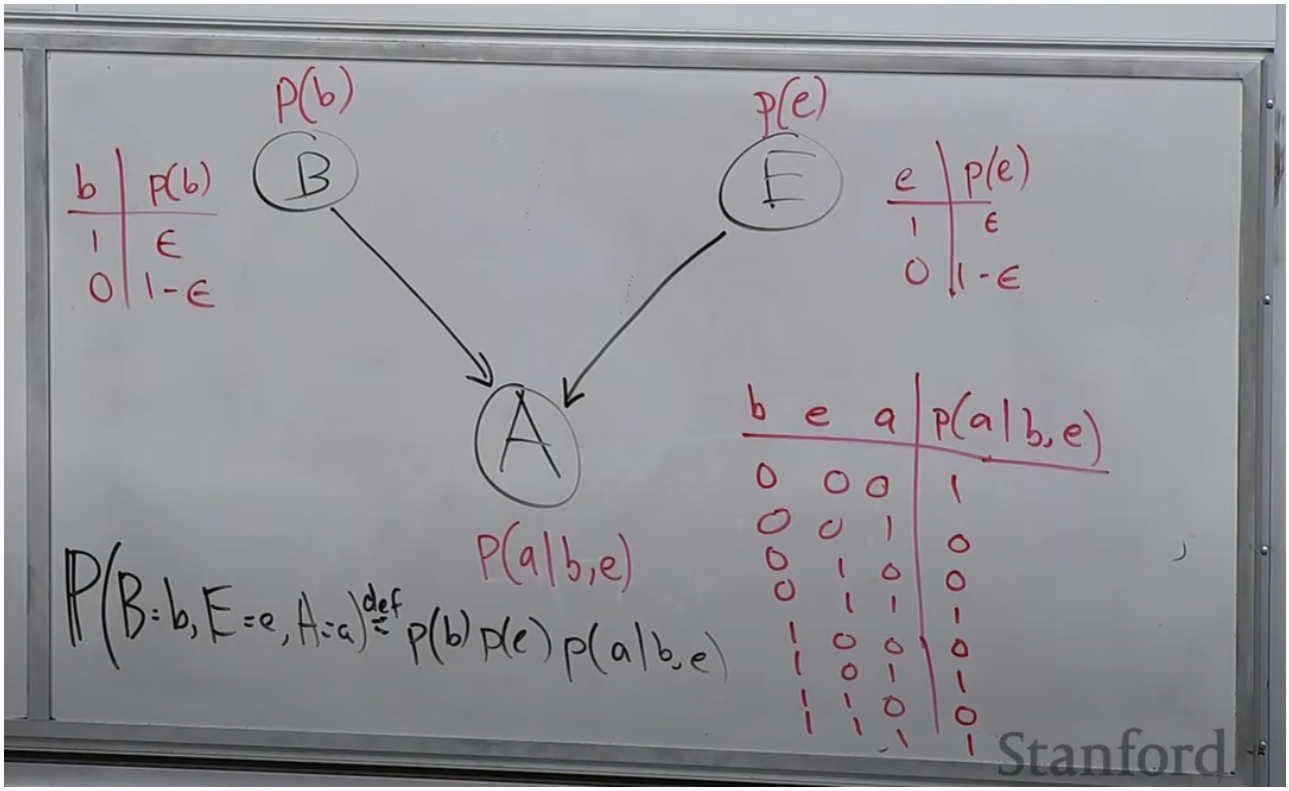


A: prob if alarm going off

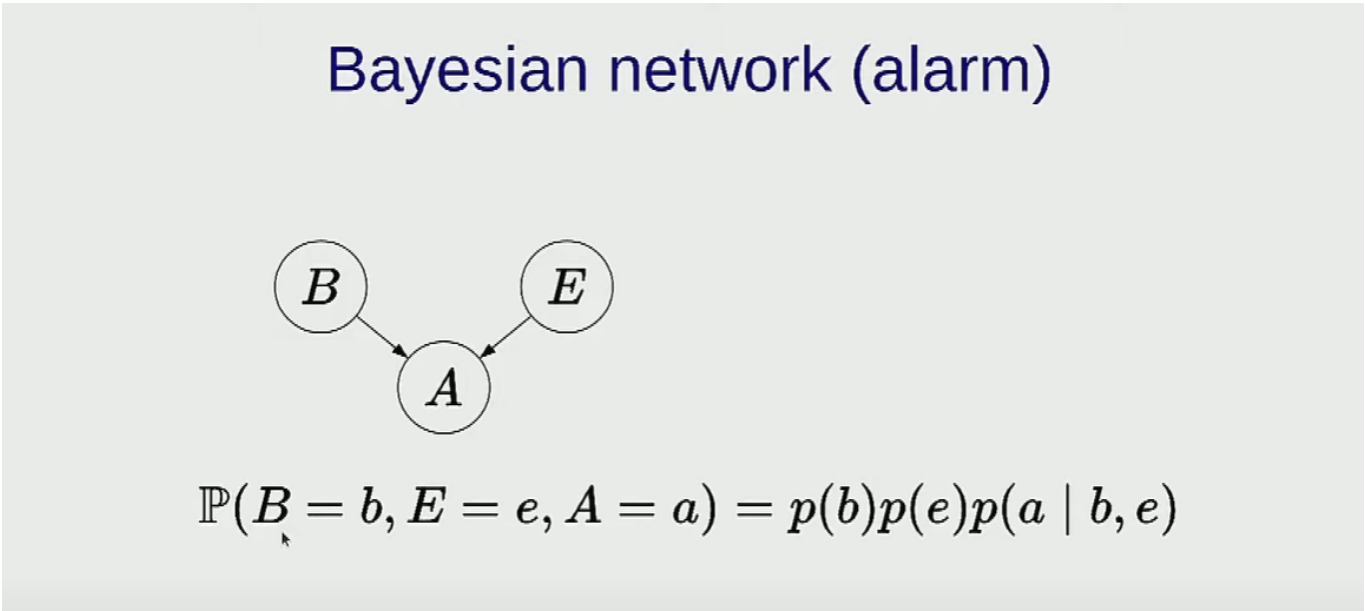
2. Define local conditional distribution

a local conditional distribution is: P of whatever that variable is, given its parents, parents are the variable directly point into it In this example the parents of a is b and e

1. define joint distribution



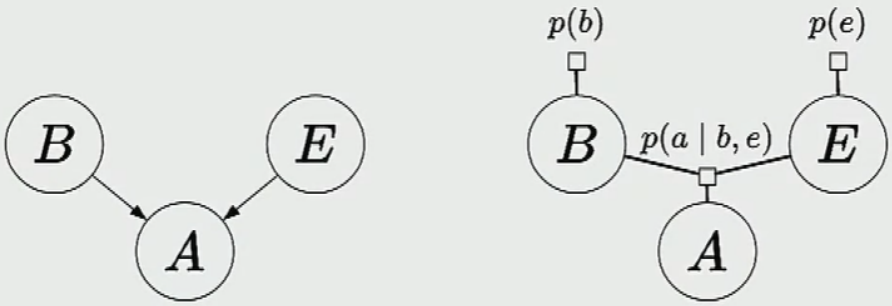
The small p stands for local conditional distribution, it's the thing we need to define (it is the ground truth cuz we defined it) The capital P stands for joint distribution



Okay this is bayasian networks, and what's the connection between this and factor graphs This looks like

```
weight = product of factors
```

Bayesian network (alarm)



$$\mathbb{P}(B = b, E = e, A = a) = p(b)p(e)p(a \mid b, e)$$

Bayesian networks are a special case of factor graphs!

Note: single factor that connects **all** parents!

In this special case: one factor per variable has a single factor connect all parents

Probabilistic inference (alarm)

Joint distribution:

b	e	a	$\mathbb{P}(B=b, E=e, A=a)$
0	0	0	$(1-\epsilon)^2$
0	0	1	0
0	1	0	0
0	1	1	$(1-\epsilon)\epsilon$
1	0	0	0
1	0	1	$\epsilon(1-\epsilon)$
1	1	0	0
1	1	1	ϵ^2

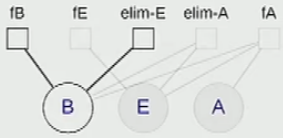
Queries: $\mathbb{P}(B)$? $\mathbb{P}(B \mid A=1)$? $\mathbb{P}(B \mid A=1, E=1)$?

[demo: $\epsilon = 0.05$]

Examples: [vote] [csp] [pair] [chain] [track] [alarm] [med] [dep] [delay] [mln] [new]
[Background] [Documentation]

```
// Alarm network (a Bayesian network)

// (B)urglary, (E)arthquake, (A)larm
variable('B', [0, 1])
variable('E', [0, 1])
variable('A', [0, 1])

// Small probability of something happening
factor('fB', 'B', {0: 0.95, 1: 0.05})
factor('fE', 'E', {0: 0.95, 1: 0.05})
// If something happens, alarm
factor('fA', 'B E A', function(b, e, a) {
  return (b || e) == (a == 1);
})

// Alarm went off: B,E no longer independent
condition('A', 1)

// No earthquake: explaining away phenomenon
condition('E', 1)

query('B')  // Was there a burglary?

sumVariableElimination({order: 'A E'})
```

Query: $\mathbb{P}(B)$
Algorithm: **variable elimination (sum)**



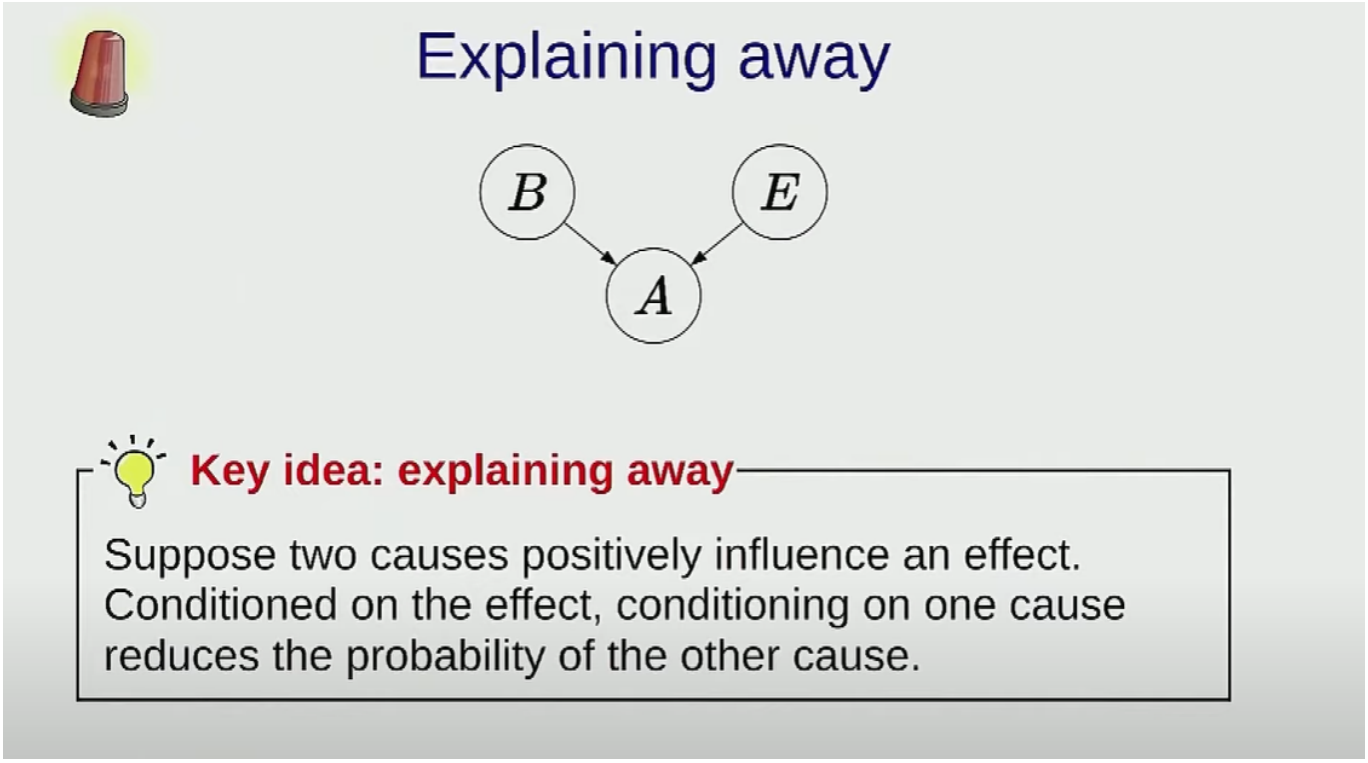
Algorithm done.
Final factor: final

B	final(B)	$\mathbb{P}(B)$
0	0.05	0.95
1	0	0.05

⇐

B	fB(B)
0	0.95
1	0.05

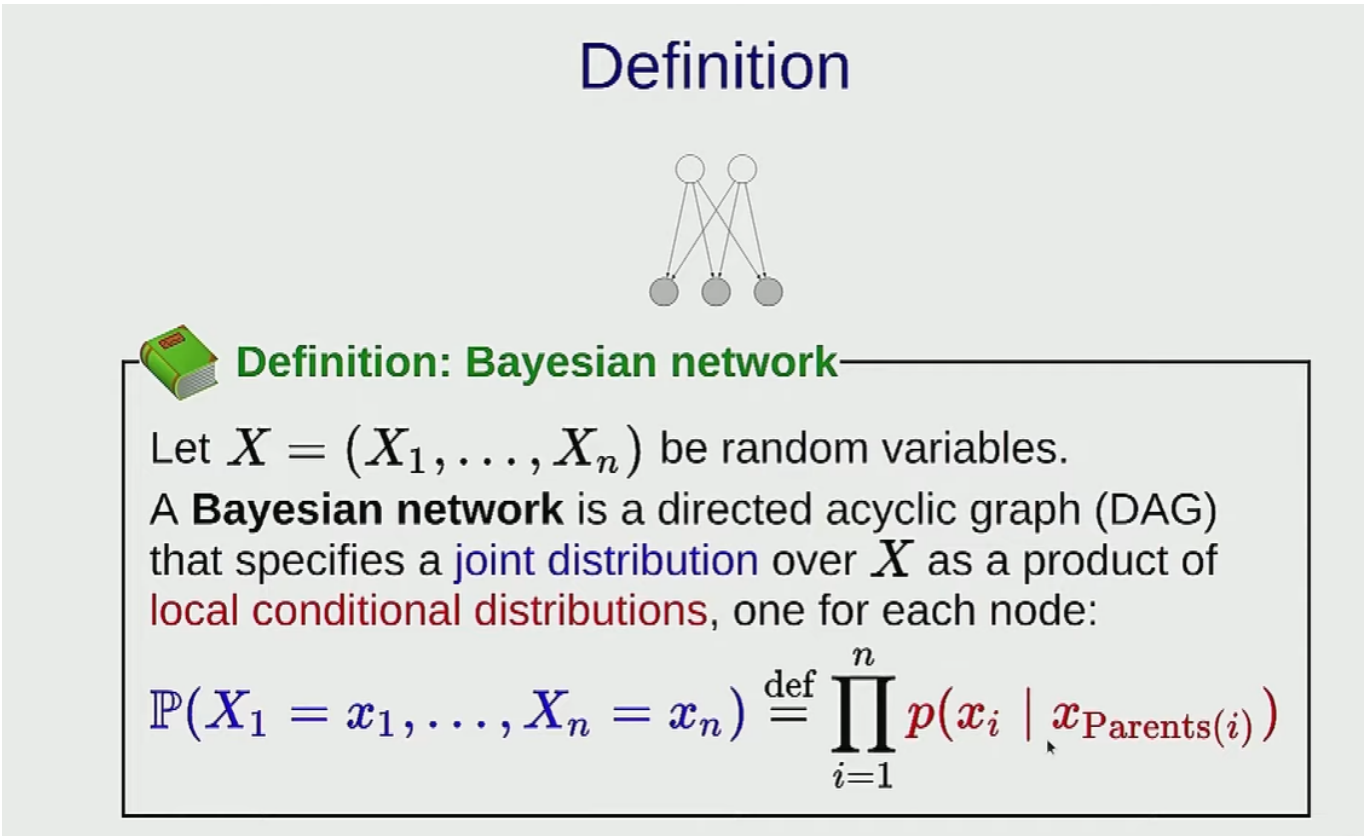
B	elim-E(B)
0	0.05
1	0.05



So given the alarms goes off and there's a bulargruy, decrease the prob of earthqeak

We might doubt that, E and B are independent. That's true but when we are conditioning on A, we already changed the independent structure of the model, which means when alarm goes off, E and B are no longer independent

Definition



Xparents(i) means the value assigned to parents of i

Special properties

Key difference from general factor graphs:

Key idea: locally normalized

All factors (local conditional distributions) satisfy:

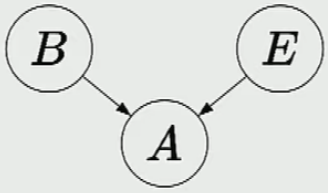
$$\sum_{x_i} p(x_i \mid x_{\text{Parents}(i)}) = 1 \text{ for each } x_{\text{Parents}(i)}$$

Implications:

- Consistency of sub-Bayesian networks
- Consistency of conditional distributions

if we summarize all possible values Xi can take on, it should be 1, and it is true for every settings of Xparents

Consistency of sub-Bayesian networks



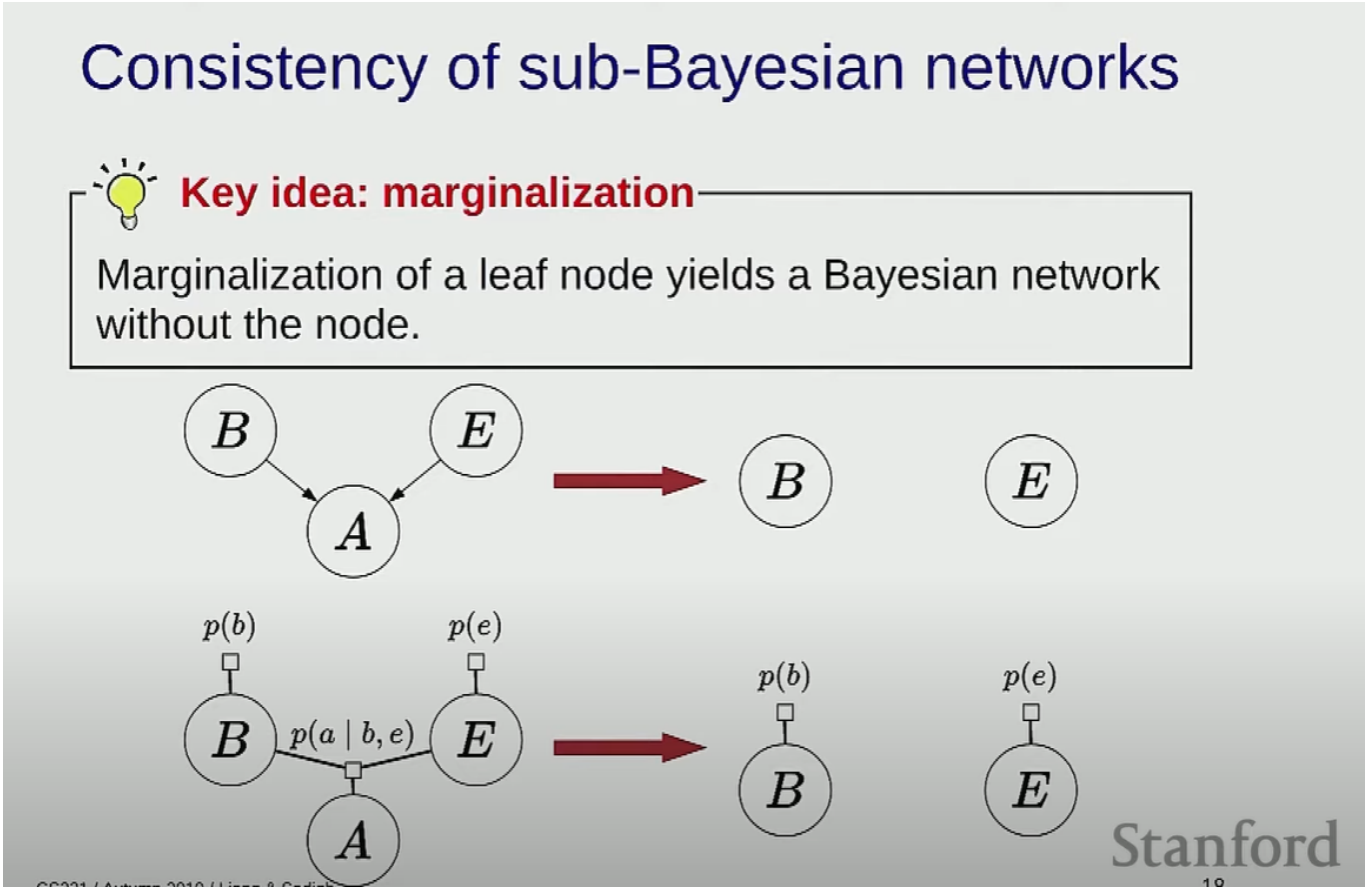
A short calculation:

$$\begin{aligned}\mathbb{P}(B=b,E=e) &\overset{\text{def}}{=} \textstyle\sum_a \mathbb{P}(B=b,E=e,A=a)\\ &\overset{\text{def}}{=} \textstyle\sum_a p(b)p(e)p(a \mid b,e)\\ &= p(b)p(e)\textstyle\sum_a p(a \mid b,e)\\ &= p(b)p(e)\end{aligned}$$

Stanford

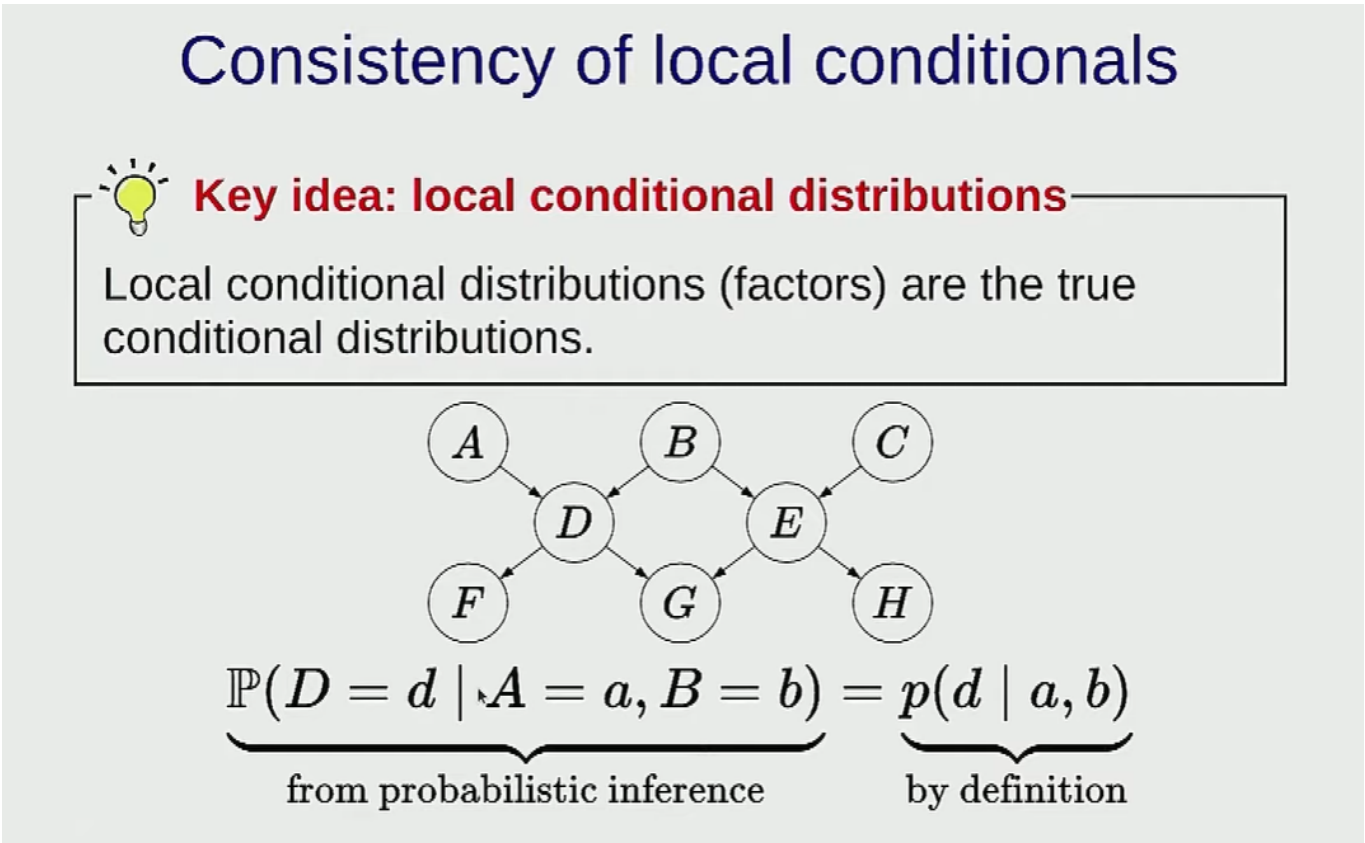
def means by the laws of probabiliy So by the laws of probabiliy we can take p(b)p(e) out side

We've made an algerbric operation graphcially

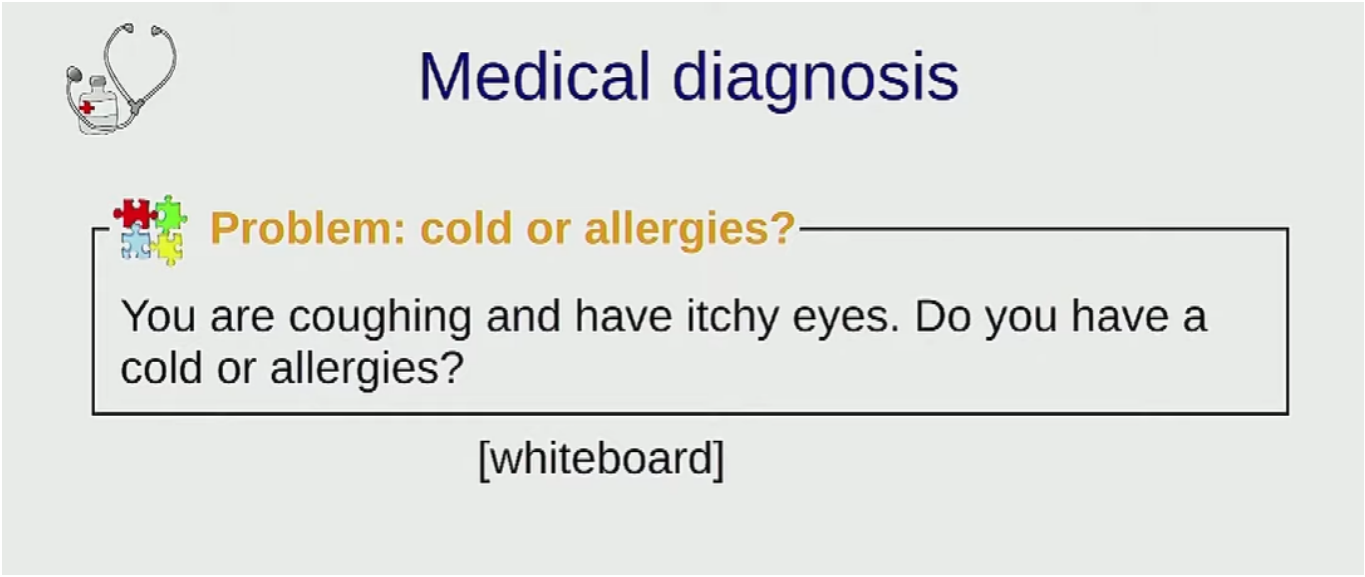


You marginalize a leaf node in Bayesian network, you can just drop it from the graph

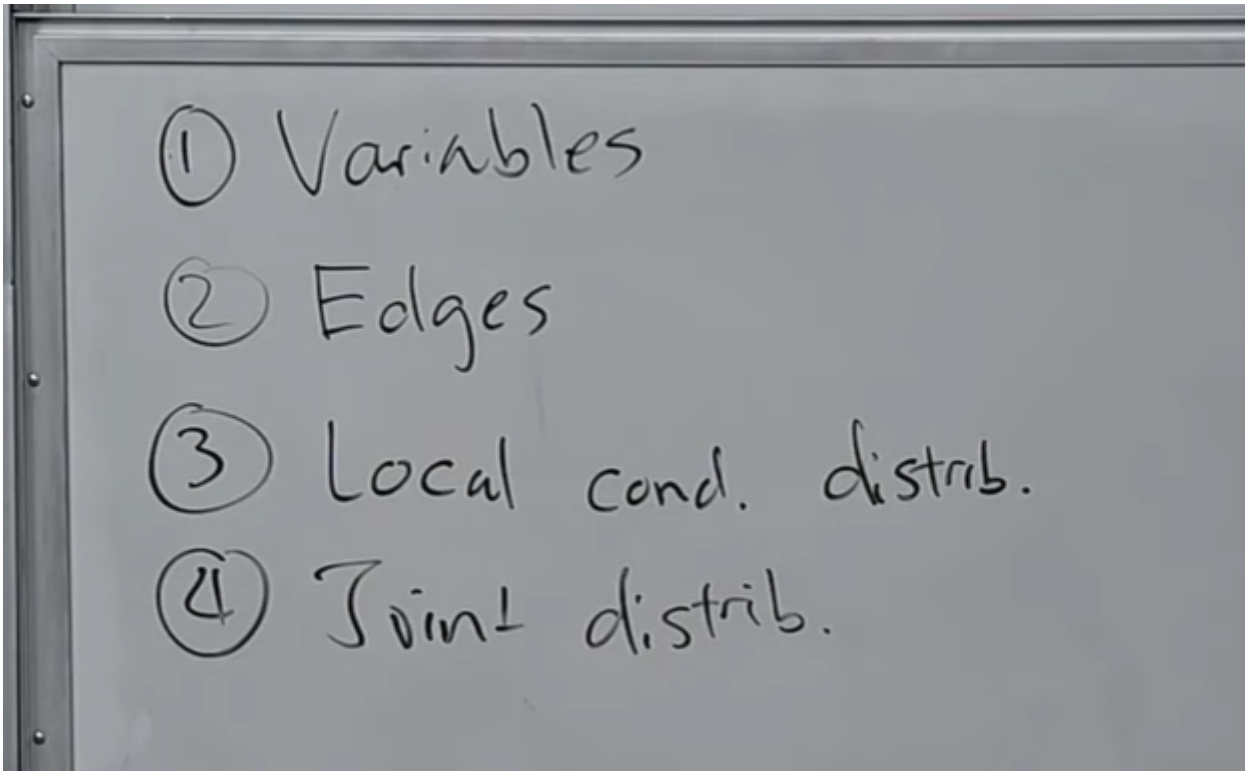
Consistency of local conditionals

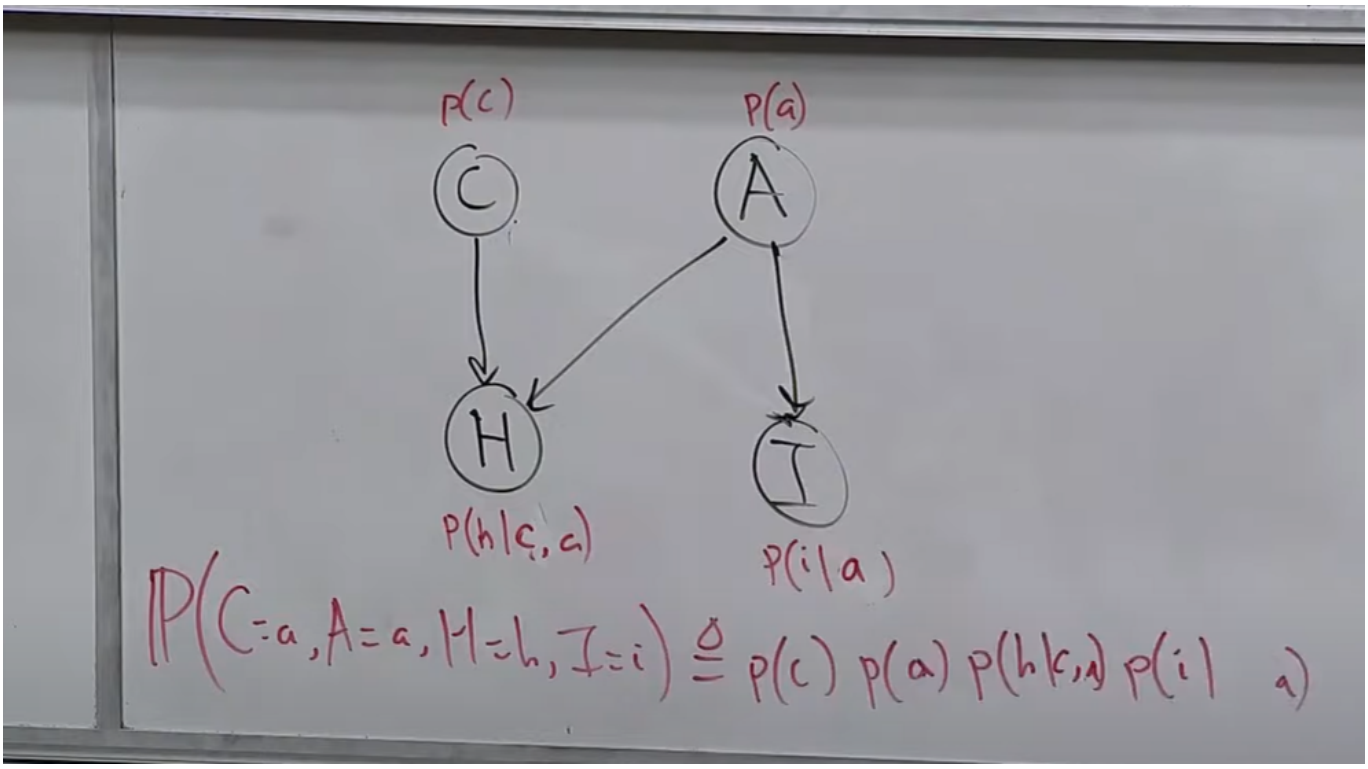


They are equal

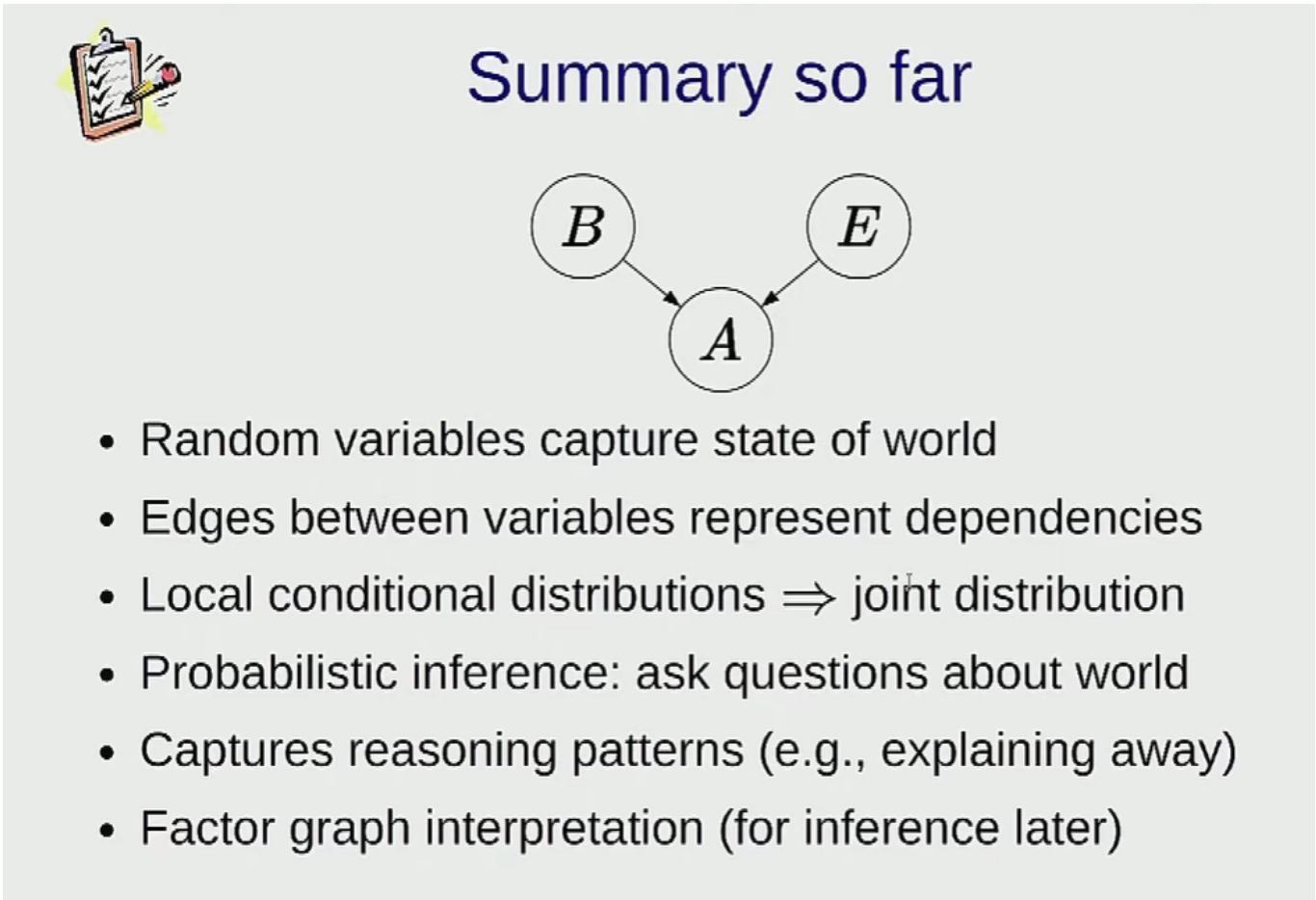


Let's define it with procedure

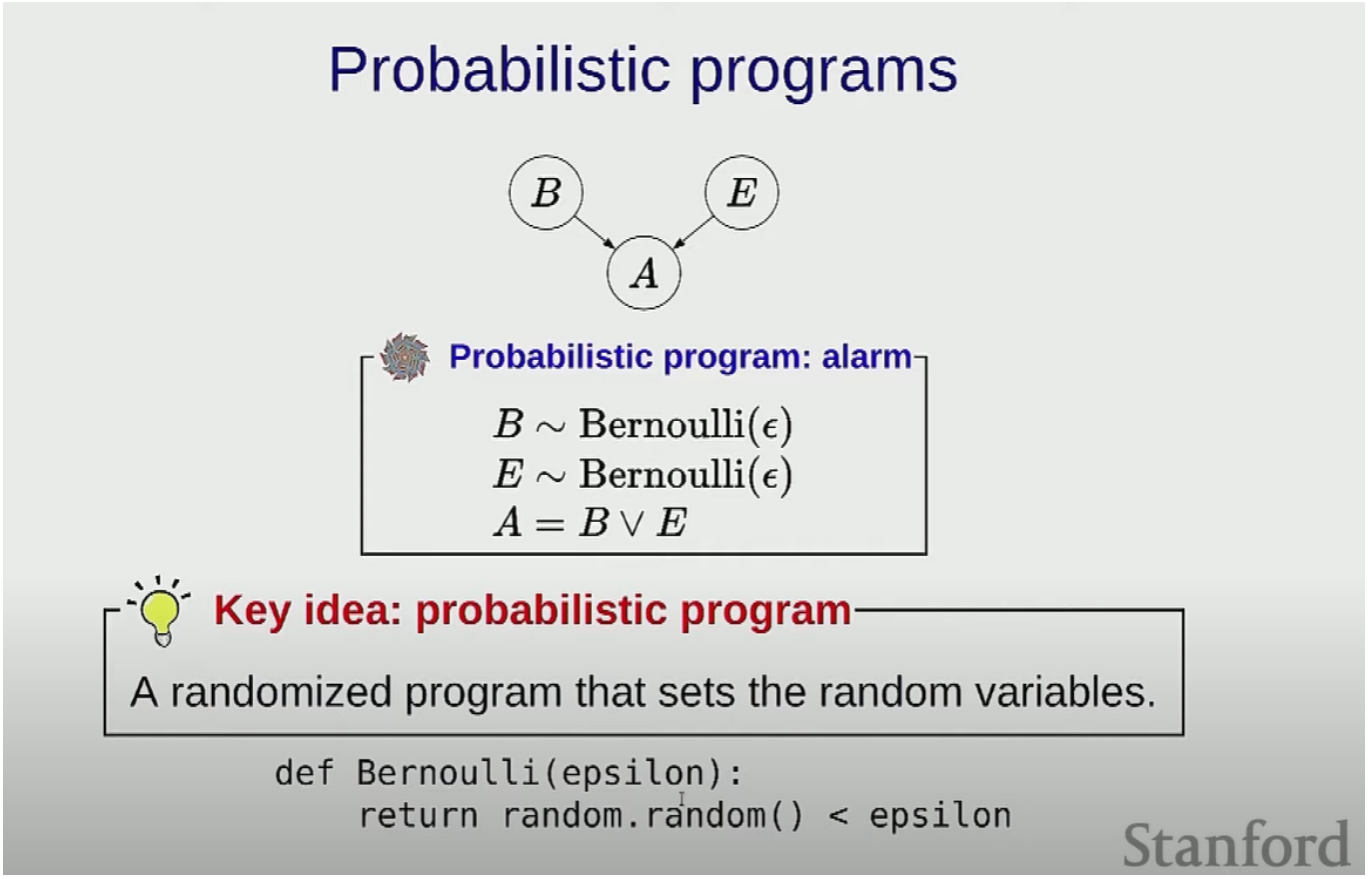




h: coughing

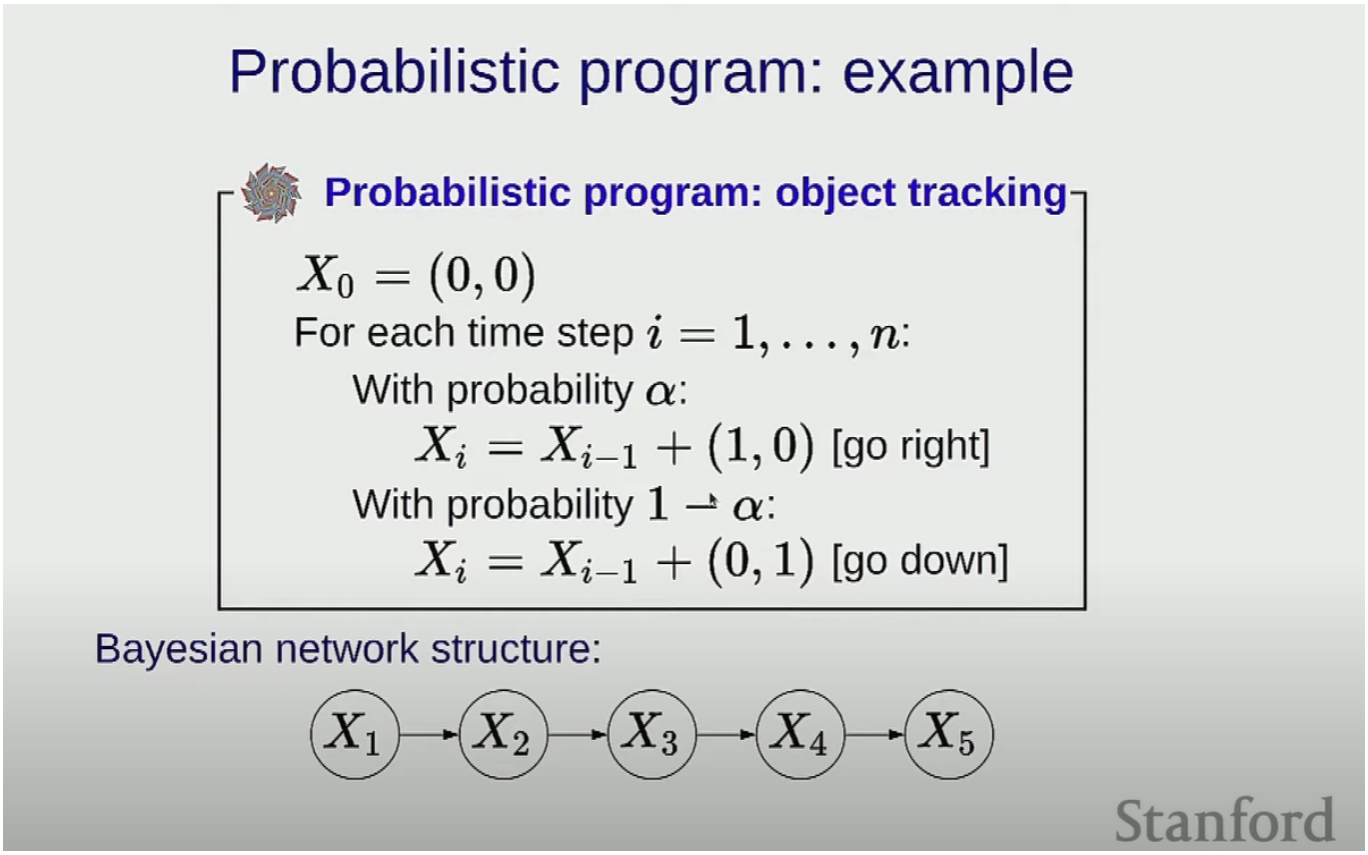


Probabilistic programs



Bernoulli: returns true with prob epsilon

define a distribution?

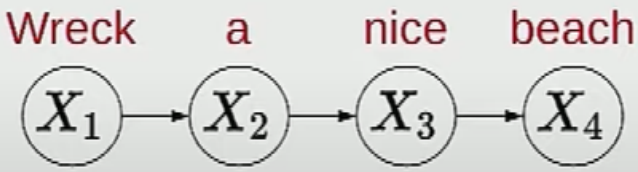


this program reduces to a particular bayesian network structure, where each Xi is only connected to Xi-1

Application: language modeling

Probabilistic program: Markov model

For each position $i = 1, 2, \ldots, n$:
Generate word $X_i \sim p(X_i \mid X_{i-1})$

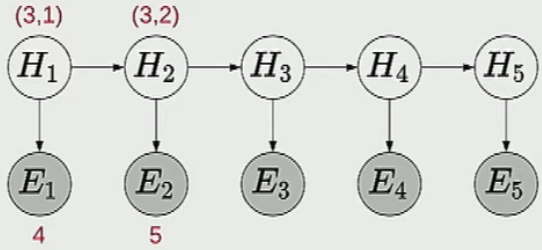


For every position, generate a word given previous word

Application: object tracking

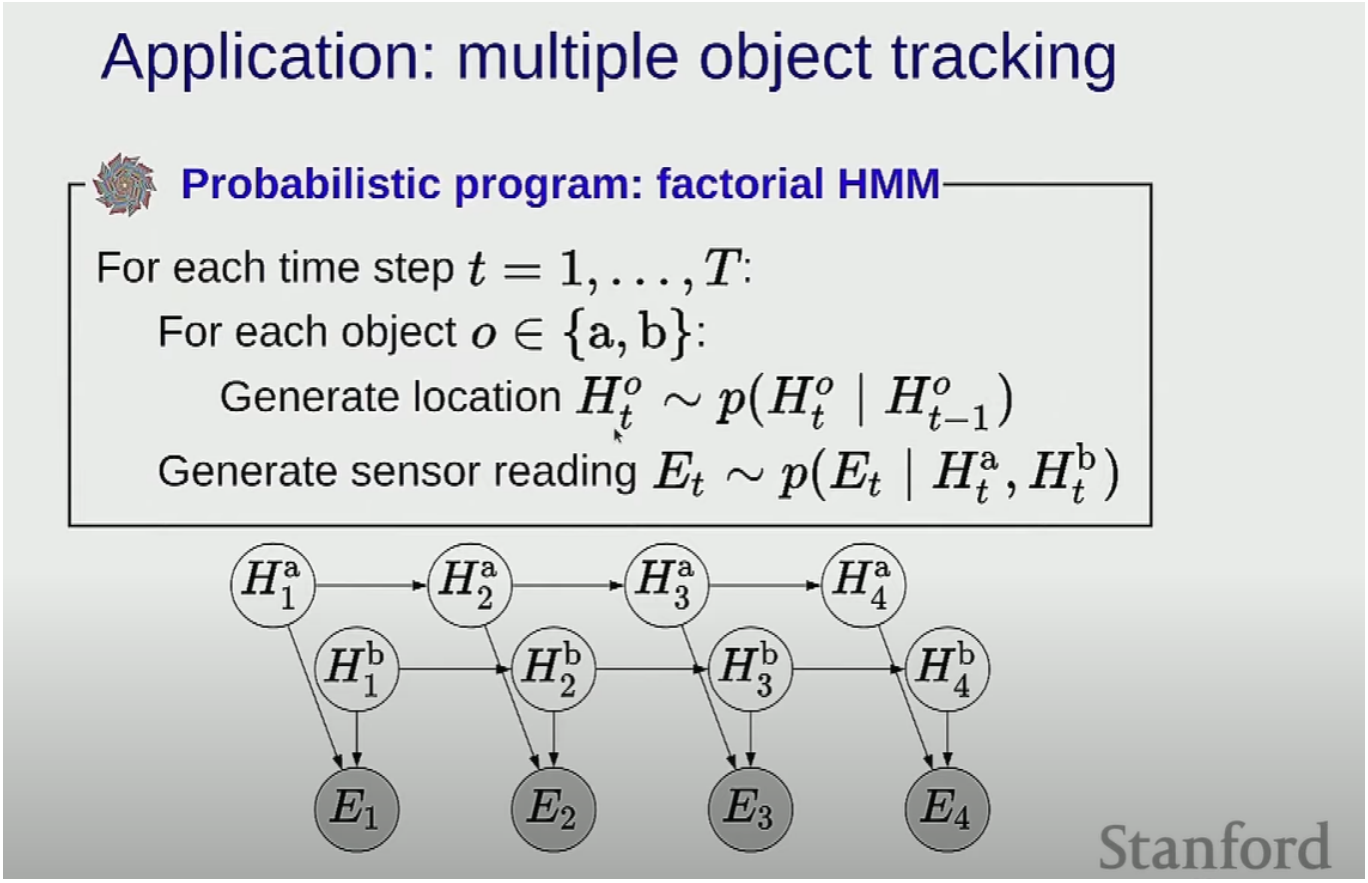
Probabilistic program: hidden Markov model (HMM)

For each time step $t = 1, \ldots, T$:
Generate object location $H_t \sim p(H_t \mid H_{t-1})$
Generate sensor reading $E_t \sim p(E_t \mid H_t)$

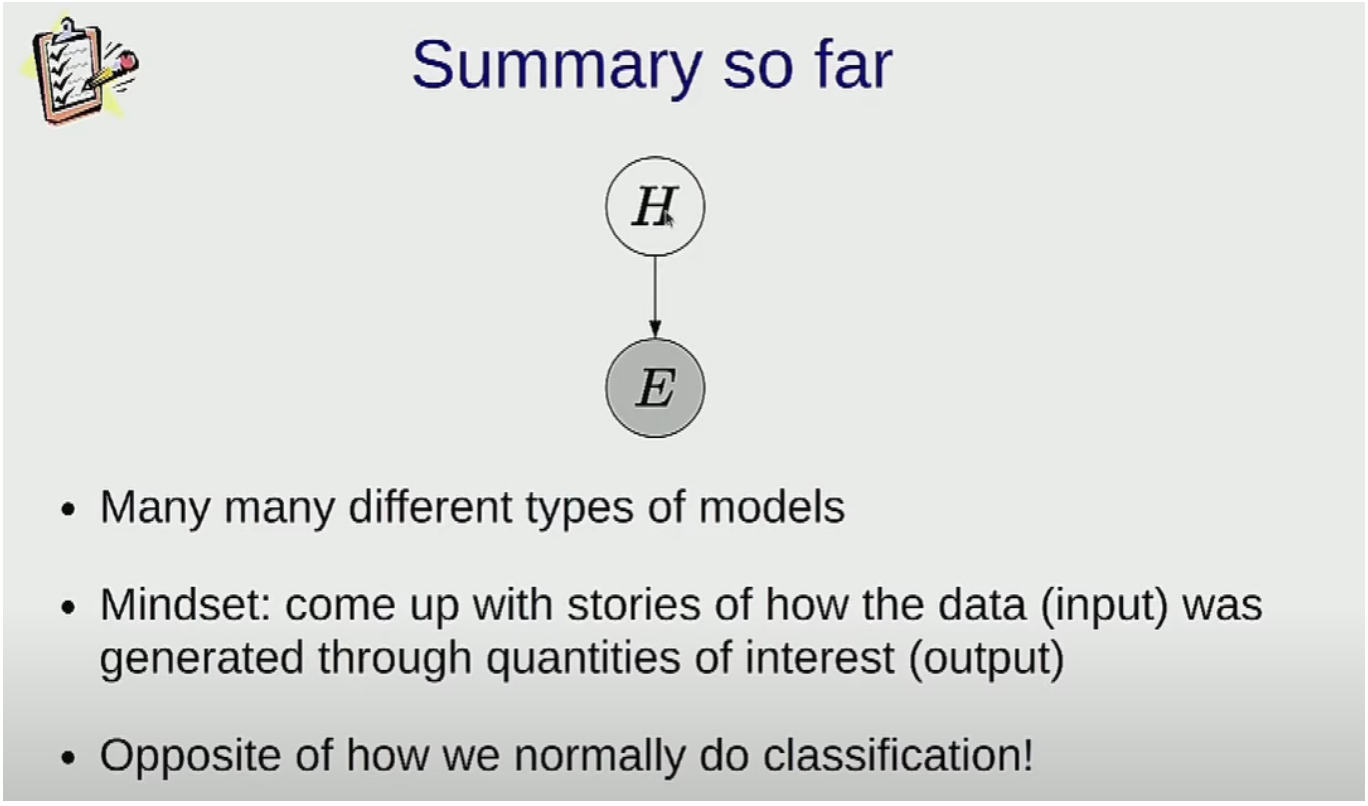


Inference: given sensor readings, where is the object?

Convention: If shade a variable, means u observe it If not shade a variable, means u can't/don't observe it



We have two object and two independent markov chains running At each time step, I only observe one sensor reading, and that sensor reading is gonna be some combination of the actual objects a and b



It's like another way around, we start with output We first define the model

A set of variables H, which u don't observe, that generates/causes a set of variables E, which you do observe

Inference

Example: Markov model

Query: $\mathbb{P}(X_3 = x_3 \mid X_2 = 5)$ for all x_3

Tedious way:

$$\propto \sum_{x_1, x_4} p(x_1)p(x_2 = 5 \mid x_1)p(x_3 \mid x_2 = 5)p(x_4 \mid x_3)$$

$$\propto \left(\sum_{x_1} p(x_1)p(x_2 = 5 \mid x_1) \right) p(x_3 \mid x_2 = 5)$$

$$\propto p(x_3 \mid x_2 = 5)$$

Fast way:

[whiteboard]

Stanford

General strategy

Query:

$$\mathbb{P}(Q \mid E = e)$$

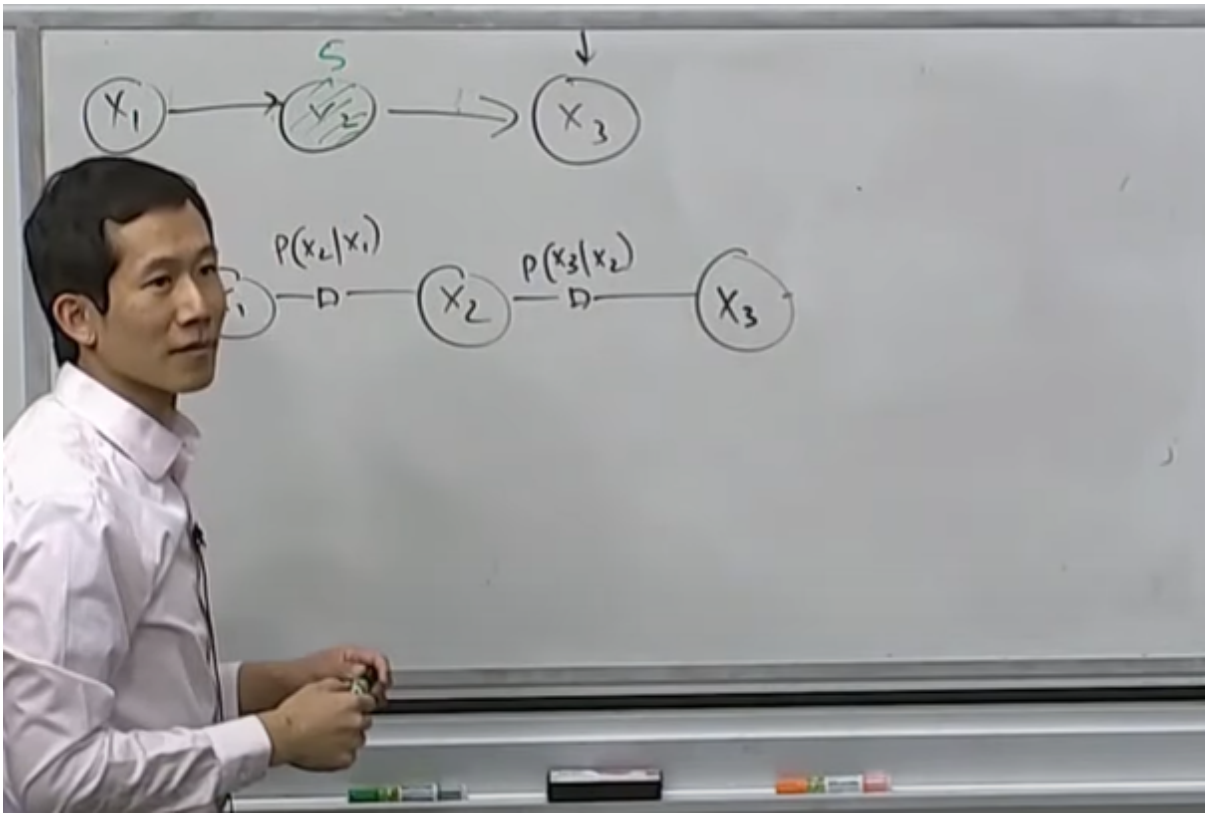
Algorithm: general probabilistic inference strategy

- Remove (marginalize) variables that are not ancestors of Q or E.
- Convert Bayesian network to factor graph.
- Condition on $E = e$ (shade nodes + disconnect).
- Remove (marginalize) nodes disconnected from Q.
- Run probabilistic inference algorithm (manual, variable elimination, Gibbs sampling, particle filtering).

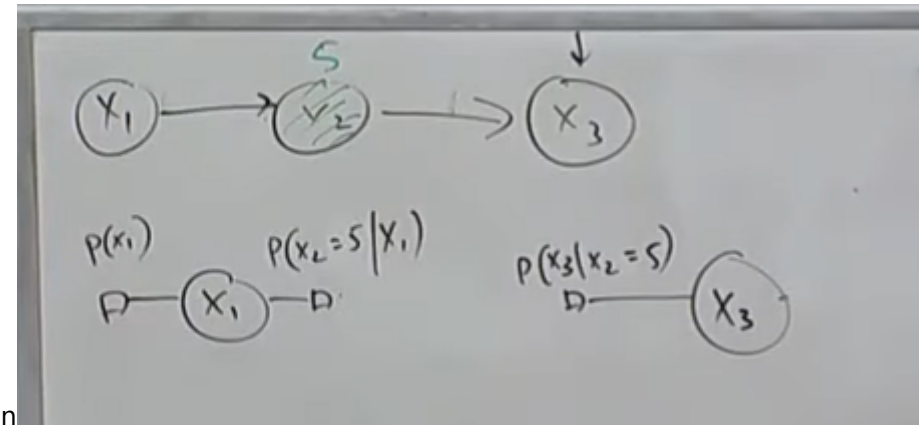
Stanford

1. We wanna remove as much as variables as I can
    1. Marginlize non-ancestors, meaning that anything upstream i can keep for now, anything down stream I can let go
2. Convert to a factor graph

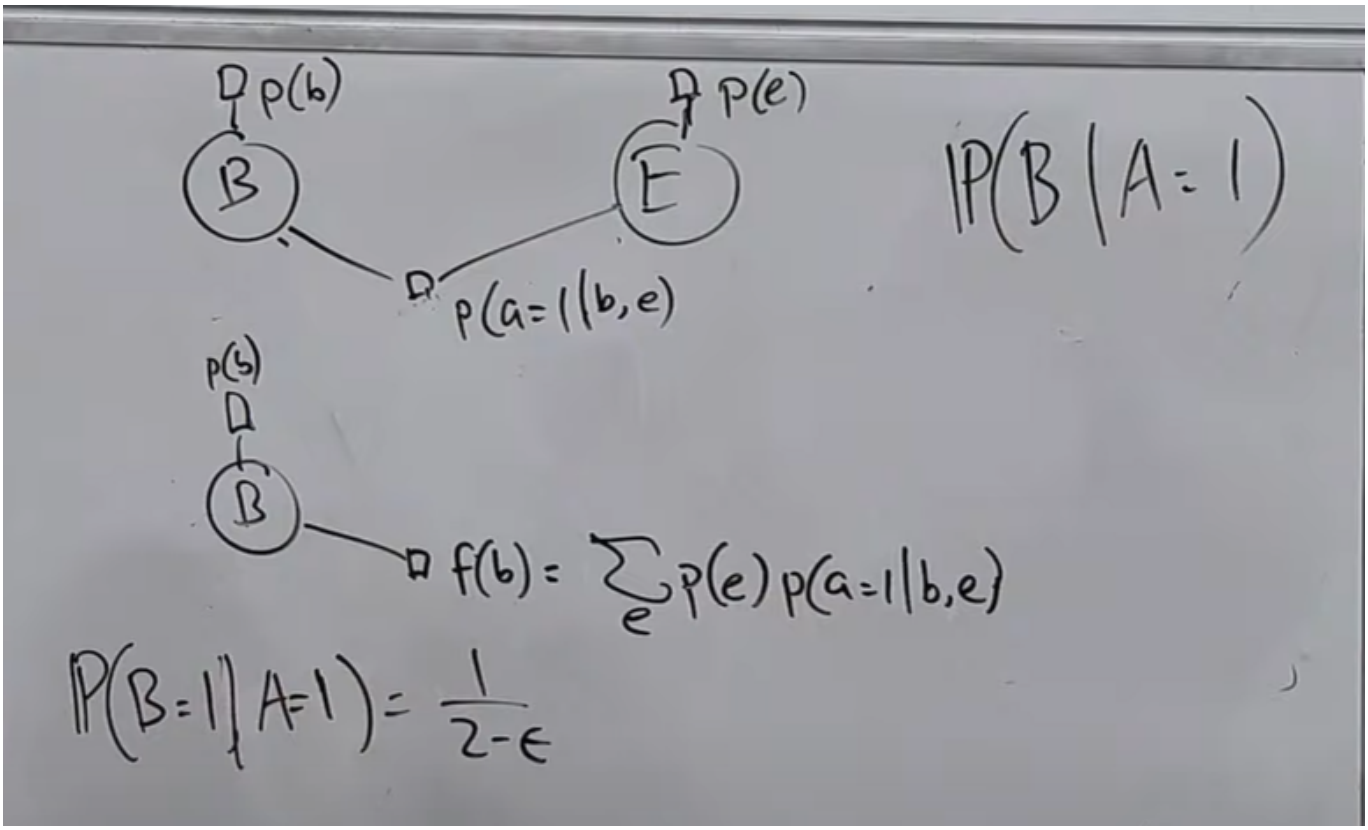
1. Remove the directions(cuz it's complex and easily to get confused), make things easy



3. Condition on evidence, we are conditioning on X2, so we wrap it put and change the factors to be a partial evaluation



4. Marginlize out the disconnected components (We care about X3 only, since they are disconnected, we can drop it since it's not related)
5. Do work, you might not just left with one node, in this step we need to solve the factor graph


P(b)
P(e)
B
E
P(a=1|b,e)
P(B|A=1)
P(b)
B
f(b) = Σ_e P(e) P(a=1|b,e)
P(B=1|A=1) = 1/(2-ε)