CS60050 Machine Learning
Graphical Models

Somak Aditya Sudeshna Sarkar

Joint Distributions

$$P(T, W)$$

-

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

Size of distribution if n variables with domain sizes d? For all but the smallest distributions, impractical to write out!

Probabilistic Models

- A probabilistic model is a joint distribution over a set of random variables
- Probabilistic models:
  - Random variables with domains
  - Assignments are called *outcomes*
  - Joint distributions: say whether assignments (outcomes) are likely
  - *Normalized:* sum to 1.0
  - Ideally: only certain variables directly interact

Distribution over T,W

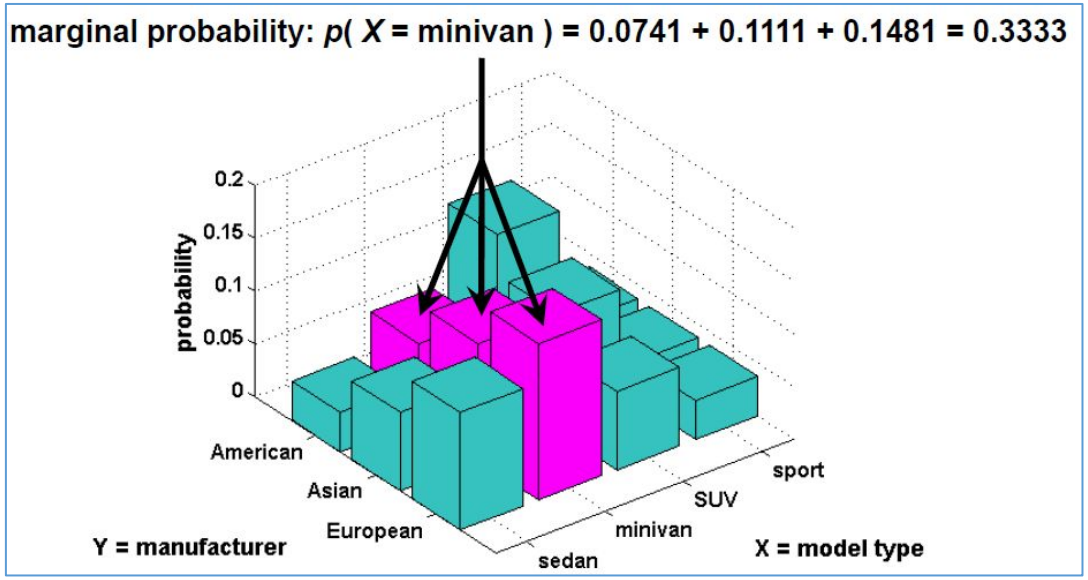
T	W	P
hot	sun	0.4
hot	rain	0.1
cold	sun	0.2
cold	rain	0.3

Constraint over T,W

T	W	P
hot	sun	T
hot	rain	F
cold	sun	F
cold	rain	T

Marginal Probability Distribution

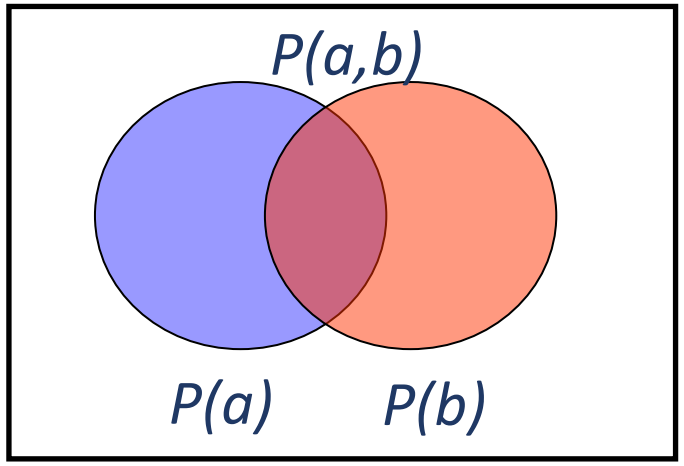
-



Slide credit: Jeff Howbert — Machine Learning Math Essentials

Conditional Probabilities

$$P(a|b) = \frac{P(a,b)}{P(b)}$$



$$P(T, W)$$

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

$$P(W = s|T = c) = \frac{P(W = s, T = c)}{P(T = c)} = \frac{0.2}{0.5} = 0.4$$

$$= P(W = s, T = c) + P(W = r, T = c)$$
$$= 0.2 + 0.3 = 0.5$$

Conditional Distributions

- Conditional distributions are probability distributions over some variables given fixed values of others

Conditional Distributions

$P(W|T)$

$P(W|T = hot)$

W	P
sun	0.8
rain	0.2

$P(W|T = cold)$

W	P
sun	0.4
rain	0.6

Joint Distribution

$P(T, W)$

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

Probabilistic Inference

- Probabilistic inference: compute a desired probability from other known probabilities (e.g. conditional from joint)

- We generally compute conditional probabilities
  - P(on time | no reported accidents) = 0.90
  - These represent the agent's *beliefs* given the evidence

- Probabilities change with new evidence:
  - P(on time | no accidents, 5 a.m.) = 0.95
  - P(on time | no accidents, 5 a.m., raining) = 0.80
  - Observing new evidence causes *beliefs to be updated*

Independence

Number of assignments to specify joint prob. distribution

Assume some events to be independent

Conditional Independence

- X is conditionally independent of Y given Z: probability distribution governing X is independent of the value of Y, given the value of Z
- $P(X \wedge Y|Z) = P(X|Z)P(Y|Z)$

$$(\forall x, y, z) P(X = x|Y = y, Z = z) = P(X = x|Z = z)$$

- $P(X|Y, Z) = P(X|Z)$

Conditional Independence

X	Y	Z
Amount of speeding fine (SF)	Type of Car (CT),	Speed (S)
Lung Cancer (LC)	Yellow Teeth (YT)	Smoking (S)
Car Motor working? (M)	Radio working (R),	Battery State (B)
Future (F),	Past (P),	Present (C)

Imagine that you know the value of Z and you are trying to guess the value of X. In your pocket is an envelope containing the value of Y. Would opening the envelope help you guess X. If not $X \perp Y|Z$

Graphical Models

- Key Idea:
  - Conditional independence assumptions useful
  - but Naïve Bayes is extreme!
  - Graphical models express sets of conditional independence assumptions via graph structure
  - Graph structure + Conditional Probability Tables (CPTs) define
    - *joint probability distribution over set of variables/nodes*

- Two types of graphical models:
  - Directed graphs (aka Bayesian Networks)
  - Undirected graphs (aka Markov Random Fields)

Topics in Graphical Models

- Representation
  - Which joint probability distributions does a graphical model represent?
- Inference
  - How to answer questions about the joint probability distribution?

    Marginal distribution of a node variable

    Most likely assignment of node variables
- Learning
  - How to learn the parameters and structure of a graphical model?

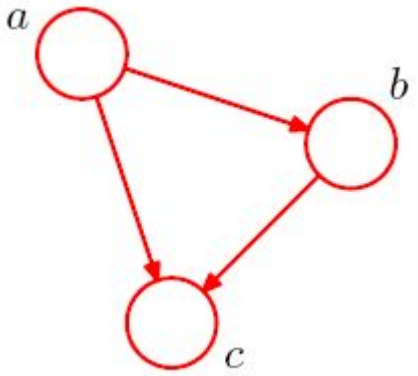
Directed GM: Bayesian Networks

Representation

Which joint probability distributions does a graphical model represent?

For any arbitrary distribution, **Chain Rule:**

$$p(X_1, X_2, \ldots, X_N) = p(X_1)p(X_2|X_1)\ldots p(X_N|X_1, \ldots, X_{N-1})$$



Fully connected directed graph between $X_1, \ldots, X_n$

Directed GM: Bayesian Networks

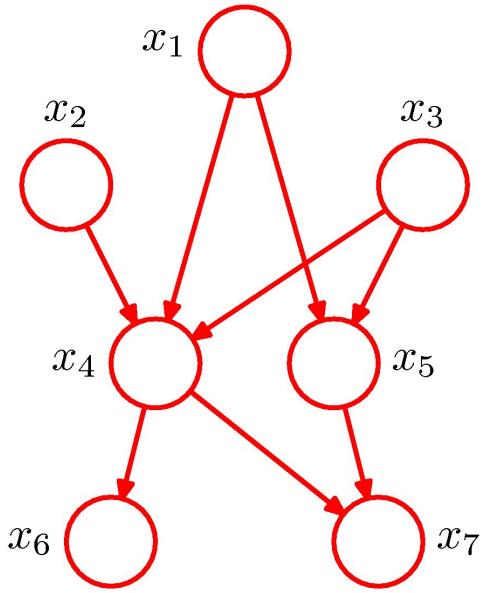
Representation

Which joint probability distributions does a graphical model represent?

For any arbitrary distribution, **Chain Rule:**

$$p(X_1, X_2, \ldots, X_N) = p(X_1)p(X_2|X_1)\ldots p(X_N|X_1, \ldots, X_{N-1})$$

Absence of edges in a graphical model conveys useful information.



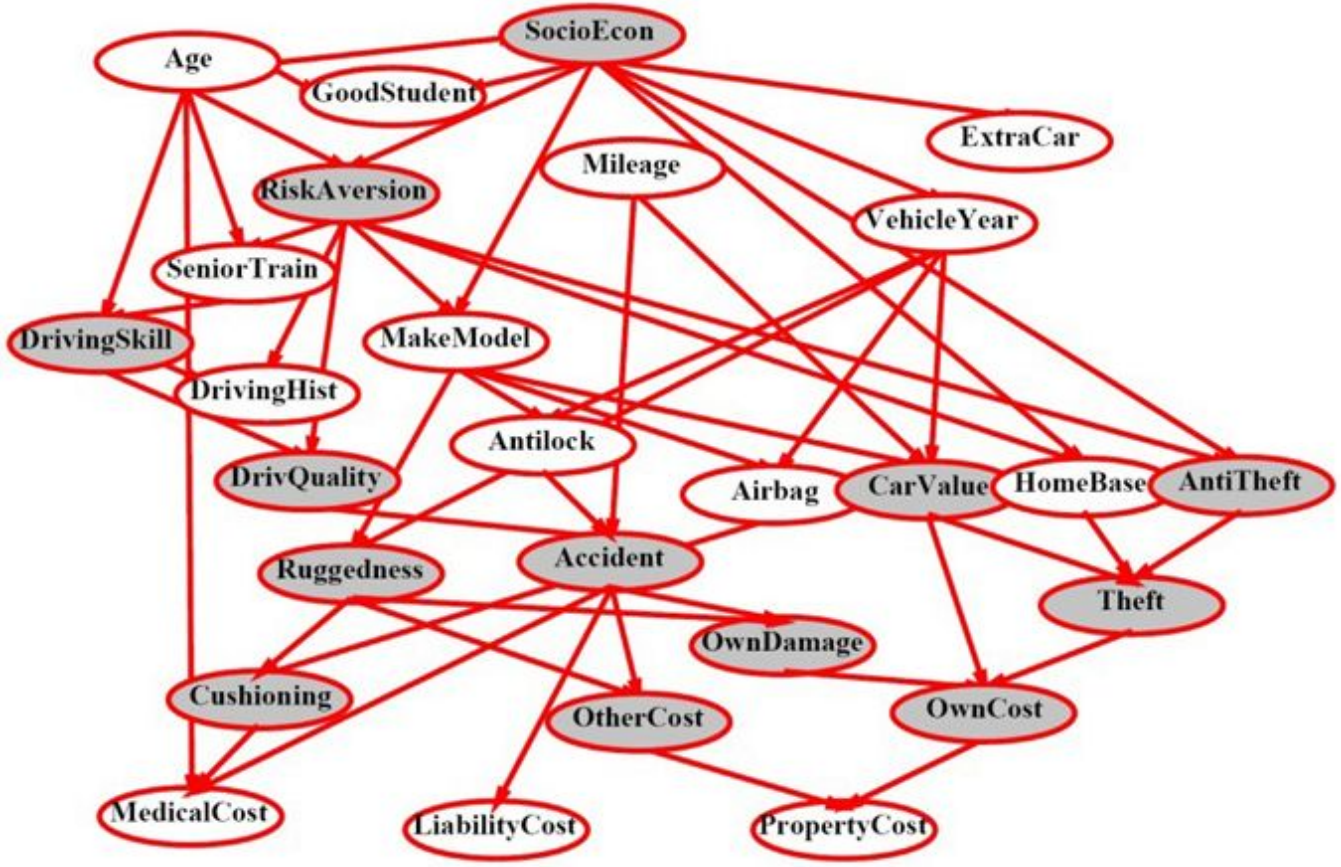
$$p(X_1, \ldots, X_7) =$$
$$p(X_1)\, p(X_2)\, p(X_3)\; p(X_4/X_1, X_2, X_3) \cdot$$
$$p(X_5/X_1, X_3)\, p(X_6/X_4)\; p(X_7/X_4, X_5)$$

Bayes Nets

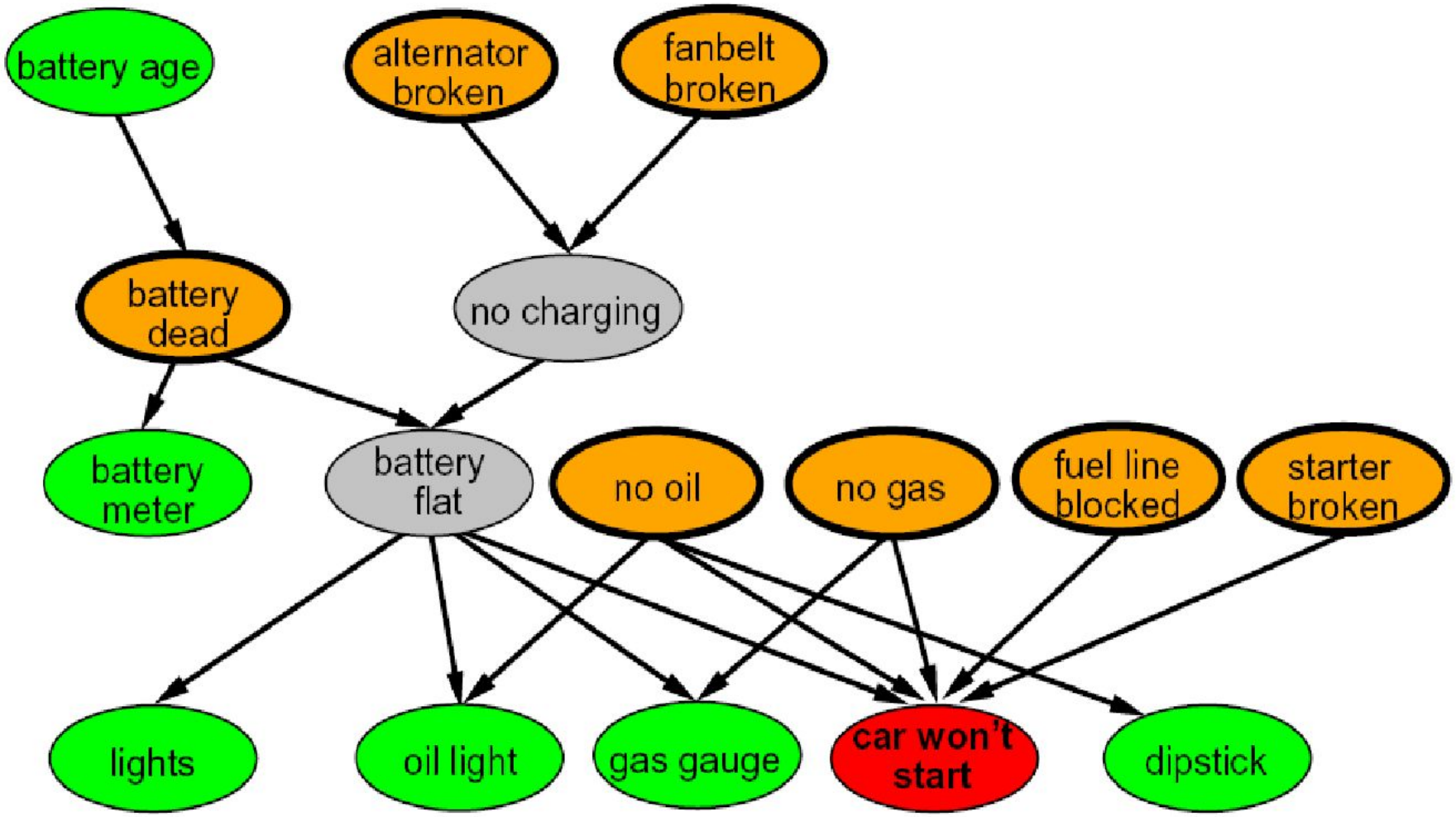
A Bayes' net is an efficient encoding of a probabilistic model of a domain

Questions we can ask:

1. Inference: given a fixed BN, what is P(X | e)?
2. Representation: given a BN graph, what kinds of distributions can it encode?
3. Modeling: what BN is most appropriate for a given domain?



Example Bayes' Net: Car

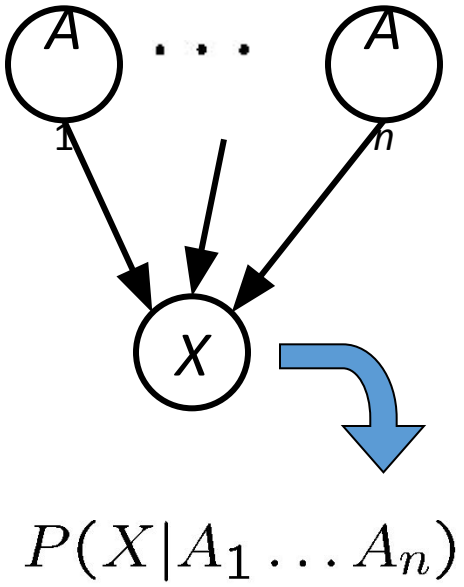


Bayes' Net Semantics

- A set of nodes, one per variable X
- A directed, acyclic graph
- A conditional distribution for each node
  - A collection of distributions over X, one for each combination of parents' values

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

  - CPT: conditional probability table
  - Description of a noisy "causal" process



$$P(X|A_1 \dots A_n)$$

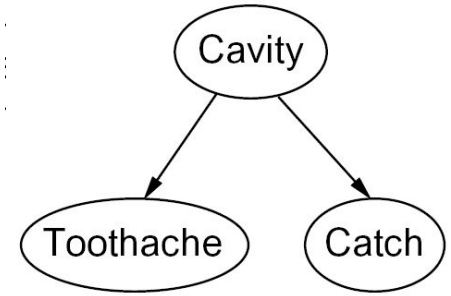
A Bayes net = Topology (graph) + Local Conditional Probabilities

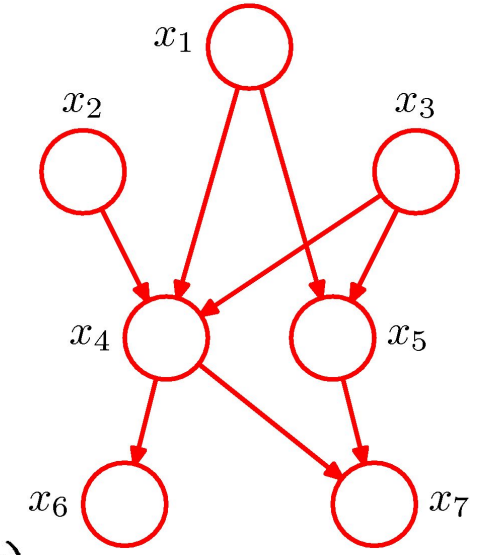
Probabilities in BNs

- Bayes' nets implicitly encode joint distributions
  - As a product of local conditional distributions

$$P(X_i|X_1, X_2, \ldots \ldots X_{i-1}) = P(X_i|Parent(X_i))$$

$Parent(X_i)$ = minimal set of predecessors of X_i in the total ordering such that other predecessors are conditionally independent of X_i given $Parent(X_i)$

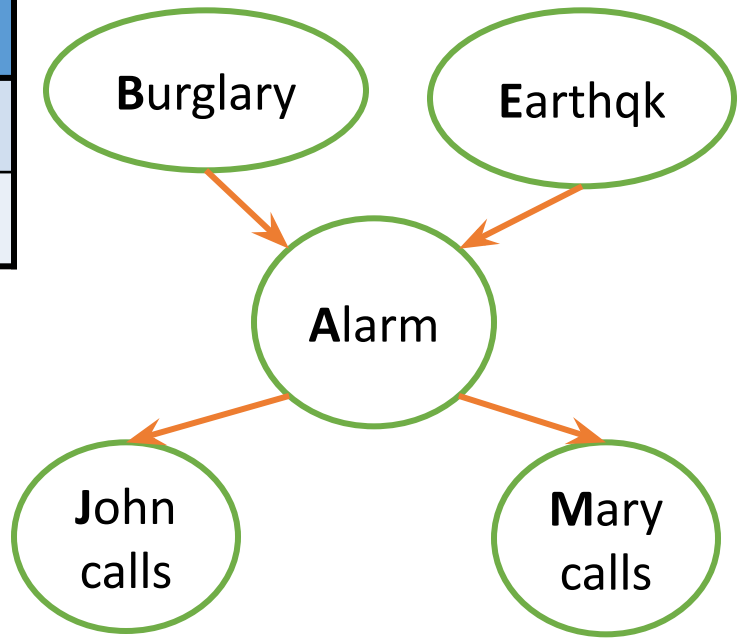




$P(+cavity, +catch, -toothache)$

Example: Alarm Network

B	P(B)
+b	0.001
-b	0.999



E	P(E)
+e	0.002
-e	0.998

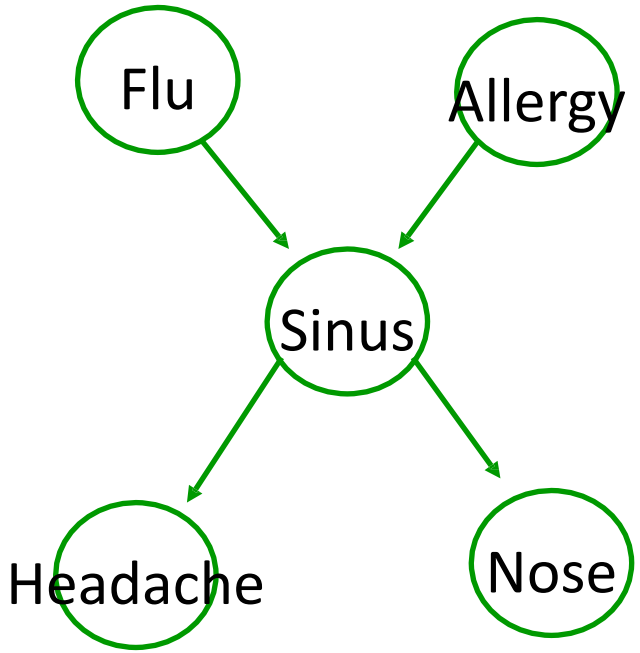
A	J	P(J|A)
+a	+j	0.9
+a	-j	0.1
-a	+j	0.05
-a	-j	0.95

A	M	P(M|A)
+a	+m	0.7
+a	-m	0.3
-a	+m	0.01
-a	-m	0.99

B	E	A	P(A|B,E)
+b	+e	+a	0.95
+b	+e	-a	0.05
+b	-e	+a	0.94
+b	-e	-a	0.06
-b	+e	+a	0.29
-b	+e	-a	0.71
-b	-e	+a	0.001
-b	-e	-a	0.999

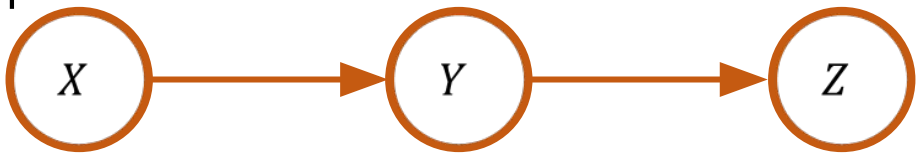
Bayesian Networks Example

- Suppose we know the following:
  - The flu causes sinus inflammation
  - Allergies cause sinus inflammation
  - Sinus inflammation causes a runny nose
  - Sinus inflammation causes headaches
- Local Markov Assumption: If you have no sinus infection, then flu has no influence on headache (flu causes headache but only through sinus)



Independence in a BN

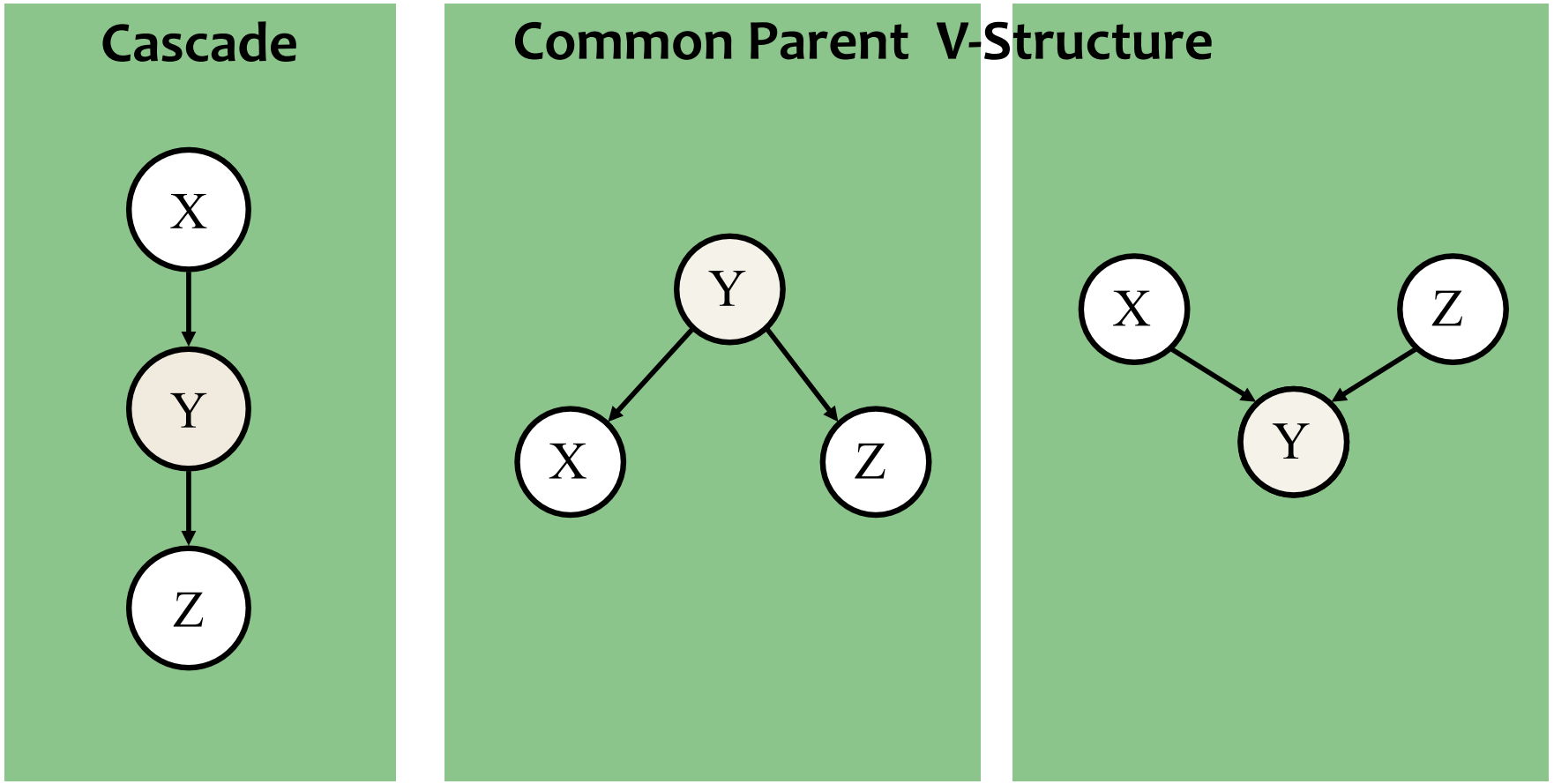
- Important question about a BN:
  - Are two nodes independent given certain evidence?
  - If yes, can prove using algebra (tedious in general)
  - If no, can prove with a counter example
  - Example:



  - Question: are X and Z necessarily independent?
    - Answer: no. Example: low pressure causes rain, which causes traffic.
    - X can influence Z, Z can influence X (via Y)
    - Addendum: they *could* be independent: how?
  - D-separation to study independence properties

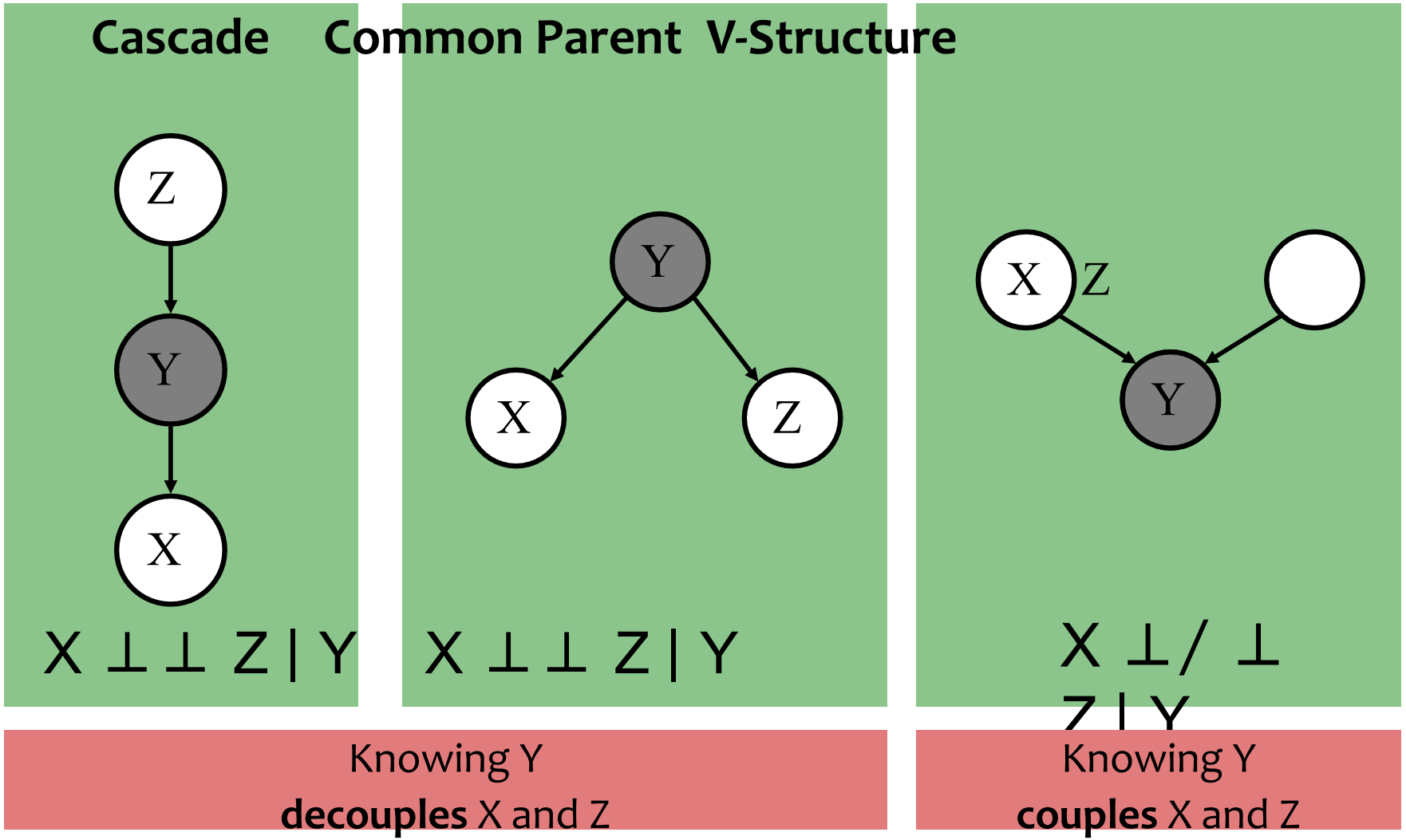
What Independencies does a Bayes Net Model?

Three cases of interest…

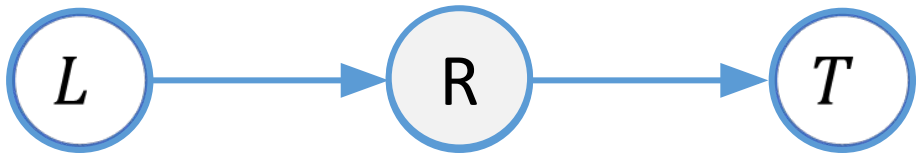


What Independencies does a Bayes Net Model?

Three cases of interest…



D-Separation: Causal Chains



L = Low Pressure

R = Rain

T = Traffic

$$P(l,r,t) = P(l)P(r|l)\,P(t|r)$$

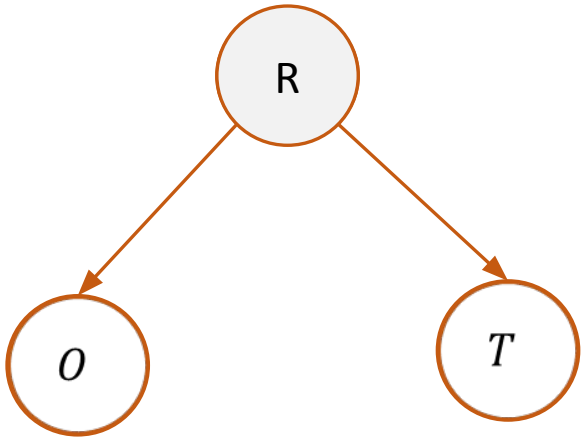
Whether L and T are independent? NO

Whether L and T are independent given R?

$$P(t|l,r) = \frac{P(l,r,t)}{P(l,r)} = \frac{P(l)P(r|l)P(t|r)}{P(l)P(r|l)} = P(t|r)$$

Evidence along the chain blocks the influence.

Common Cause



$$P(r,o,t) = P(r)P(o|r)\,P(t|r)$$

O = Lawn Overflow

R = Rain

T = Traffic

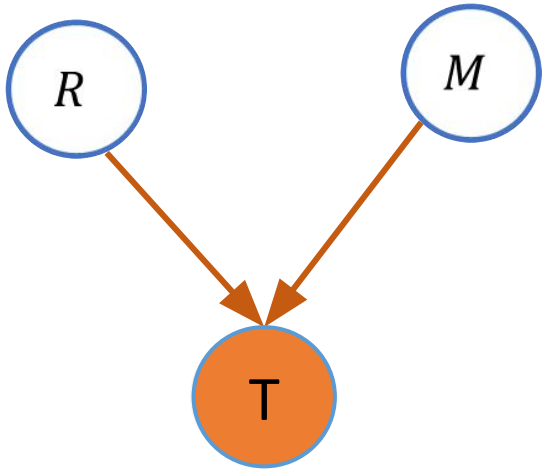
Whether O and T are independent? NO

Whether O and T are independent given R?

$$P(t|o,r) = \frac{P(r,o,t)}{P(o,r)} = \frac{P(r)P(o|r)P(t|r)}{P(r)P(o|r)} = P(t|r)$$

Observing the cause blocks influence between effects

Common Effect (V-Structure)



Whether R and M are independent? YES

Whether R and M are independent given T? NO

Traffic is a effect of two competing explanations
seeing traffic puts the rain and the ballgame in competition as explanation.

M = Match

R = Rain

T = Traffic

Observing an effect activates influence between causes

The General Case

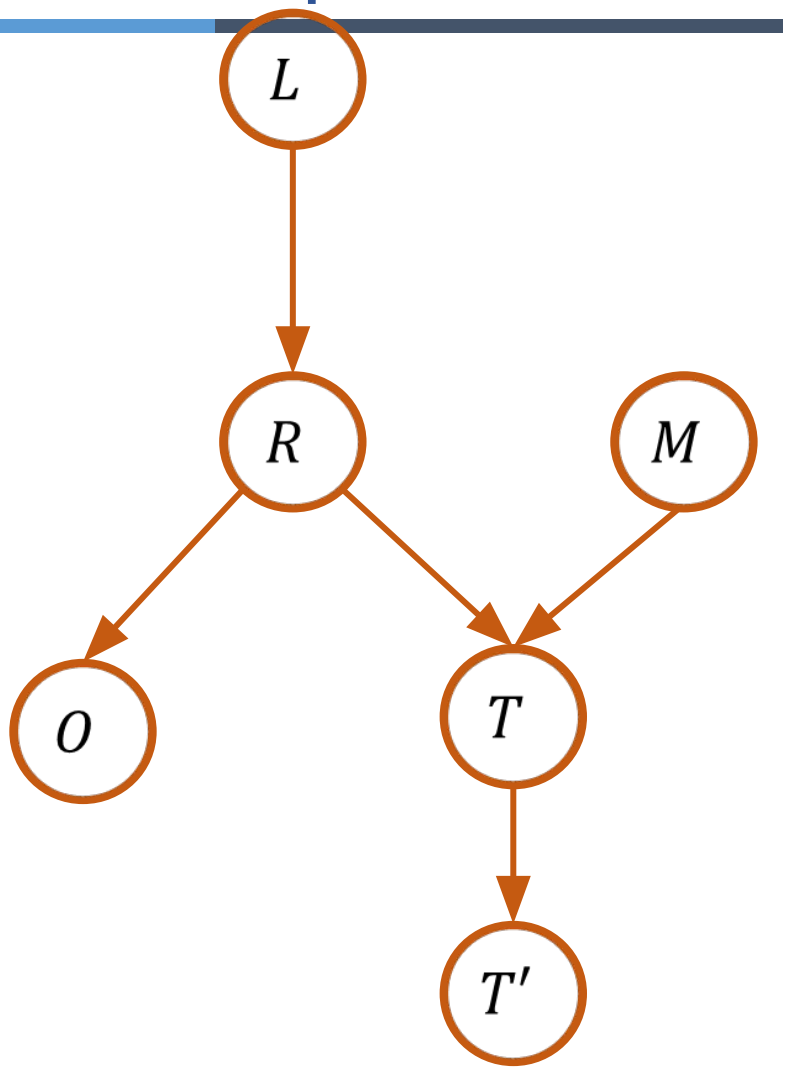
- General question: in a given BN, are two variables independent (given evidence)?

- Solution: analyze the graph

- Any complex example can be broken

  into repetitions of the three canonical cases

D-Separation

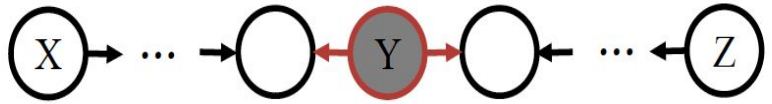


If variables X and Z are d-separated given a **set** of variables E
Then X and Z are conditionally independent given the **set** E

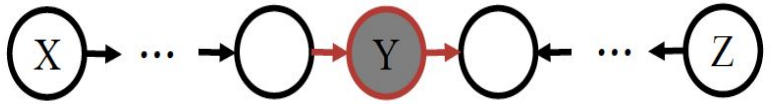
Definition #1:
Variables X and Z are d-separated given a **set** of evidence variables E iff every path from X to Z is "blocked".

A path is "blocked" whenever:

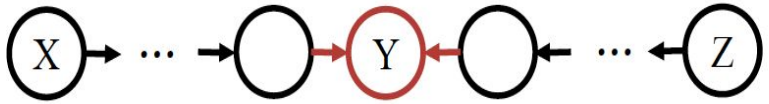
1. $\exists$ Y on path s.t. Y $\in$ E and Y is a "common parent"



2. $\exists$ Y on path s.t. Y $\in$ E and Y is in a "cascade"



3. $\exists$ Y on path s.t. {Y, descendants(Y)} $\notin$ E and Y is in a "v-structure"



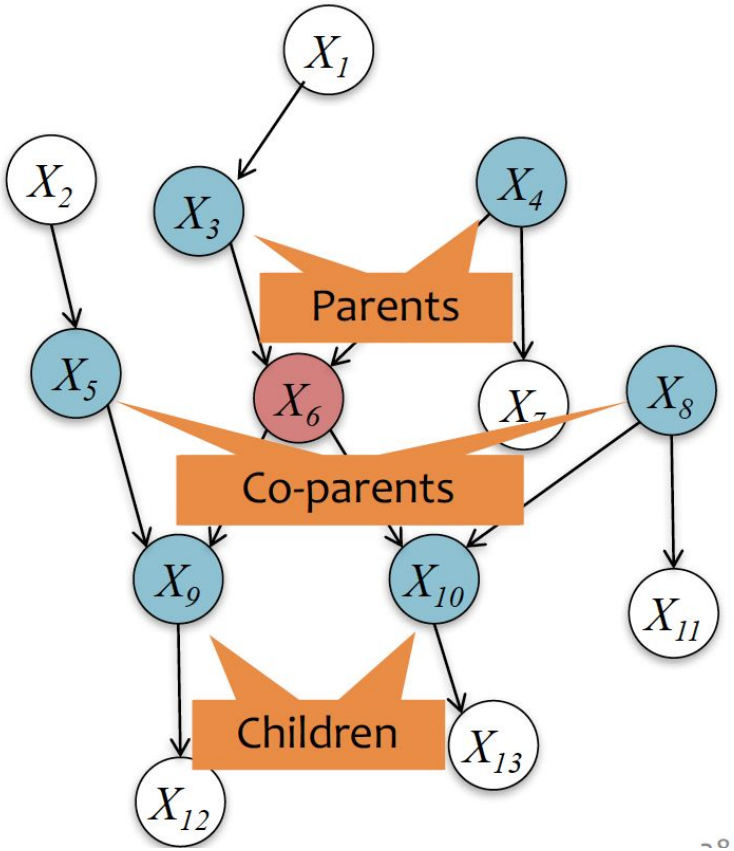
Markov Blanket

Def: the **co-parents** of a node are the parents of its children

Def: the **Markov Blanket** of a node is the set containing the node's parents, children, and co-parents.

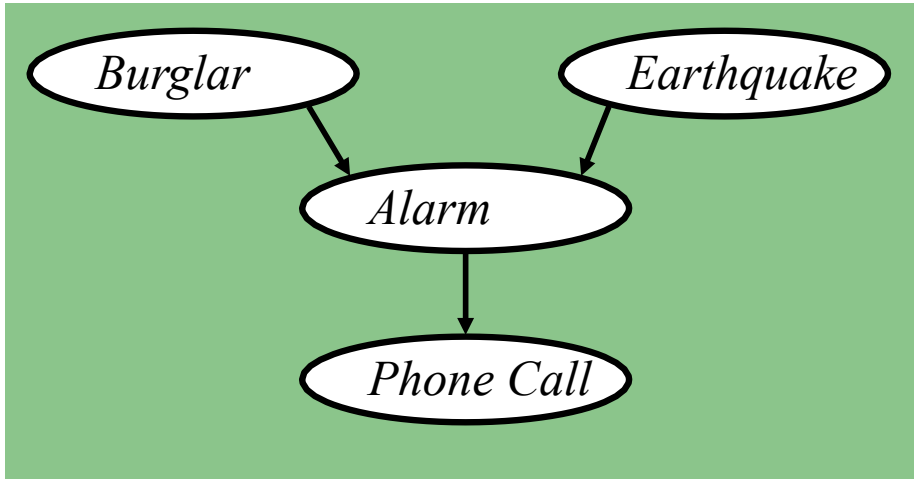
Theorem: a node is **conditionally independent** of every other node in the graph given its **Markov blanket**

Example: The Markov Blanket of X_6 is $\{X_3, X_4, X_5, X_8, X_9, X_{10}\}$



Exercise: The "Burglar Alarm" example

- Your house has a twitchy burglar alarm that is also sometimes triggered by earthquakes.
- Earth arguably doesn't care whether your house is currently being burgled
- While you are on vacation, one of your neighbors calls and tells you your home's burglar alarm is ringing. Uh oh!



Quiz: True or False?

$$\text{Burglar} \perp \text{Earthquake} \mid \text{PhoneCall}$$

Slide from William Cohen

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

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

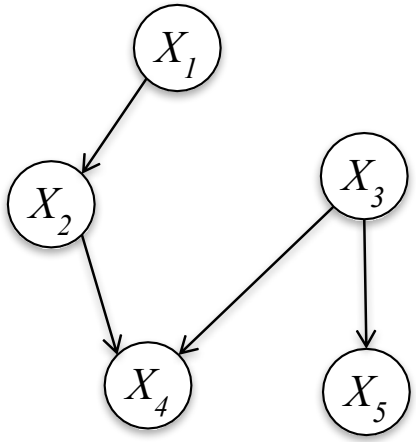
- BNs: Huge space savings!
- Also easier to elicit local CPTs
- Also faster to answer queries

SUPERVISED LEARNING FOR BAYES NETS

Computational Problems

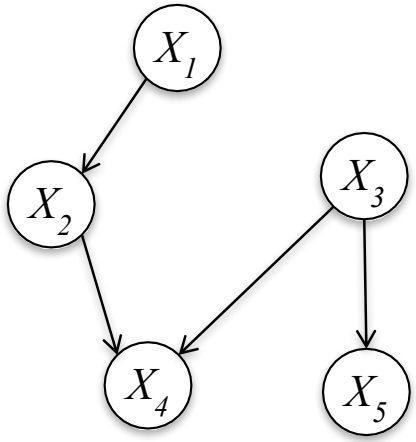
- Learning the structure of the Bayes net
  - (What would be the guiding principle?)
- Learning the parameters
  - Supervised? Unsupervised?
- Inference:
  - Computing the probability of an event: [#P Complete, Roth'93, '96]
    - Given structure and parameters
    - Given an observation (evidence) E, what is the probability of assignment Y?
      - $P(R = off, A = off \mid E = e)$ =? (E, Y are sets of instantiated variables)
  - Most likely explanation (Maximum A Posteriori assignment, MAP, MPE) [NP-Hard; Shimony'94]
    - Given structure and parameters
    - Given an observation (evidence) E, what is the most likely assignment to Y?
    - $Argmax_y P(Y = y \mid E = e)$ (Say, $Y = (R, A)$)
    - (E, Y are sets of instantiated variables)

Learning Fully Observed BNs



$$p(X_1, X_2, X_3, X_4, X_5) = p(X_5|X_3)p(X_4|X_2, X_3) p(X_3)p(X_2|X_1)p(X_1)$$

Learning Fully Observed BNs



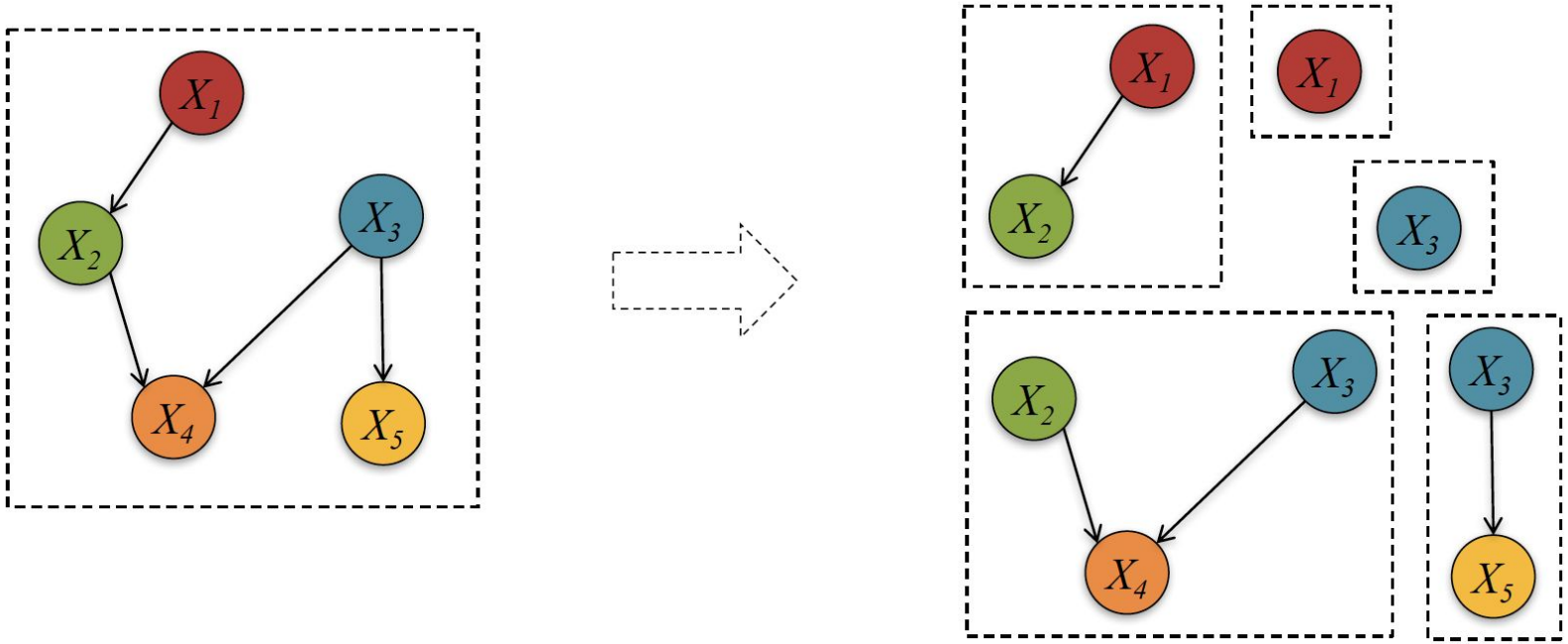
$$p(X_1, X_2, X_3, X_4, X_5) =$$
$$p(X_5|X_3)p(X_4|X_2, X_3)$$
$$p(X_3)p(X_2|X_1)p(X_1)$$

How do we learn these conditional and marginal distributions for a Bayes Net?

Learning Fully Observed BNs

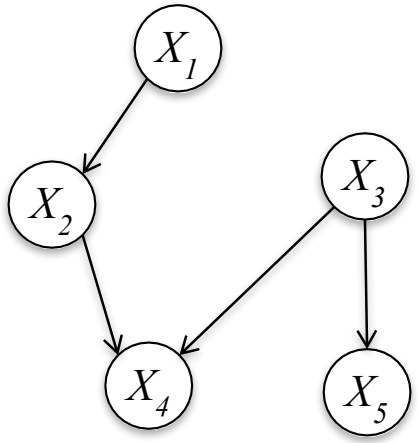
- Learning this fully observed Bayesian Network is **equivalent** to learning five (small / simple) independent networks from the same data

$$p(X_1, X_2, X_3, X_4, X_5) = p(X_5|X_3)p(X_4|X_2, X_3)p(X_3)p(X_2|X_1)p(X_1)$$



Learning Fully Observed BNs

How do we **learn** these conditional and marginal distributions for a Bayes Net?



$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(X_1, X_2, X_3, X_4, X_5)$$

$$= \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \log p(X_5|X_3, \theta_5) + \log p(X_4|X_2, X_3, \theta_4) + \log p(X_3|\theta_3) + \log p(X_2|X_1, \theta_2) + \log p(X_1|\theta_1)$$

$$\theta_1^* = \underset{\theta_1}{\operatorname{argmax}} \log p(X_1|\theta_1)$$

$$\theta_2^* = \underset{\theta_2}{\operatorname{argmax}} \log p(X_2|X_1, \theta_2)$$

$$\theta_3^* = \underset{\theta_3}{\operatorname{argmax}} \log p(X_3|\theta_3)$$

$$\theta_4^* = \underset{\theta_4}{\operatorname{argmax}} \log p(X_4|X_2, X_3, \theta_4)$$

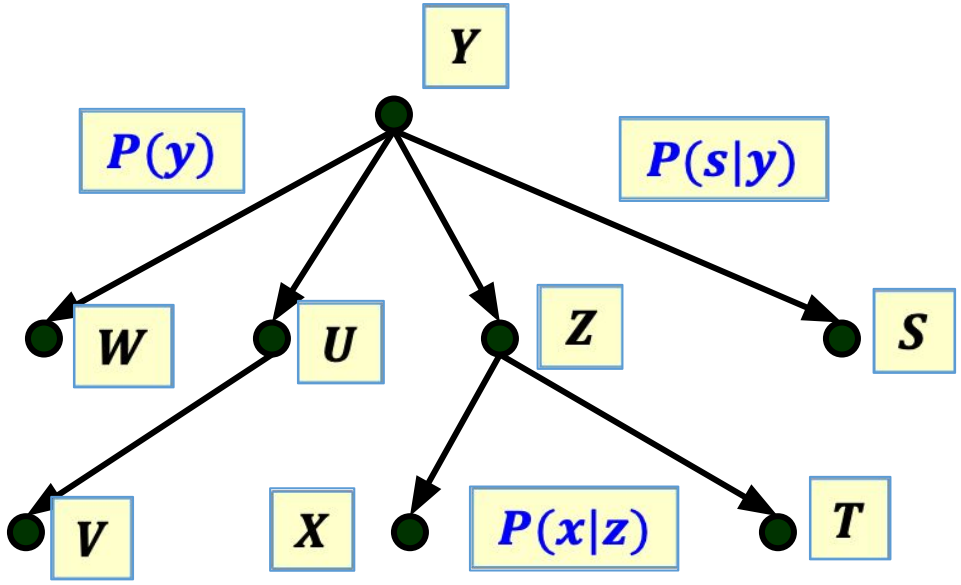
$$\theta_5^* = \underset{\theta_5}{\operatorname{argmax}} \log p(X_5|X_3, \theta_5)$$

Tree Dependent Distributions

- Learning Problem:
  - Given data (n tuples) assumed to be sampled from a tree-dependent distribution
  - Find the tree representation of the distribution.
- Assuming uniform prior on trees, the Maximum Likelihood approach is to maximize $P(D|T)$,

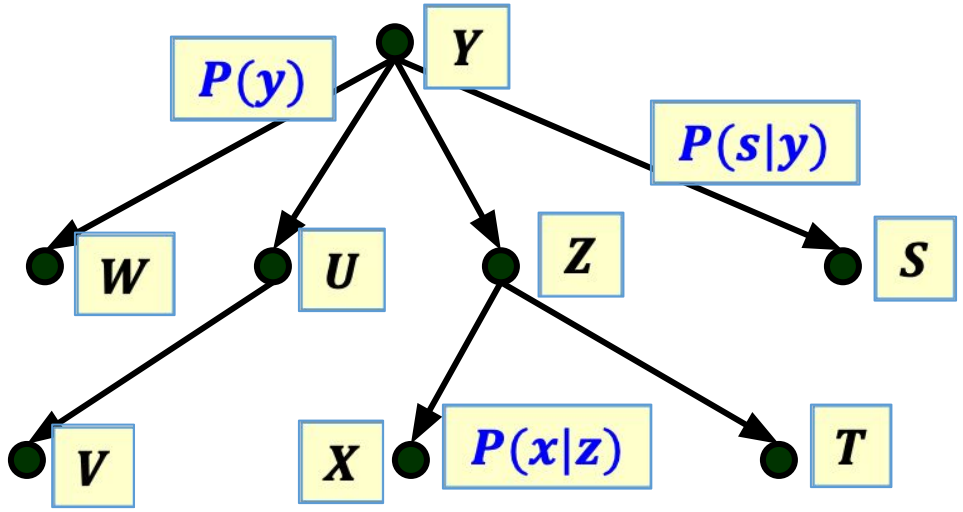
$$T_{ML} = argmax_T P(D|T) = argmax_T \prod_{\{x\}} P_T(x_1, x_2, \ldots x_n)$$

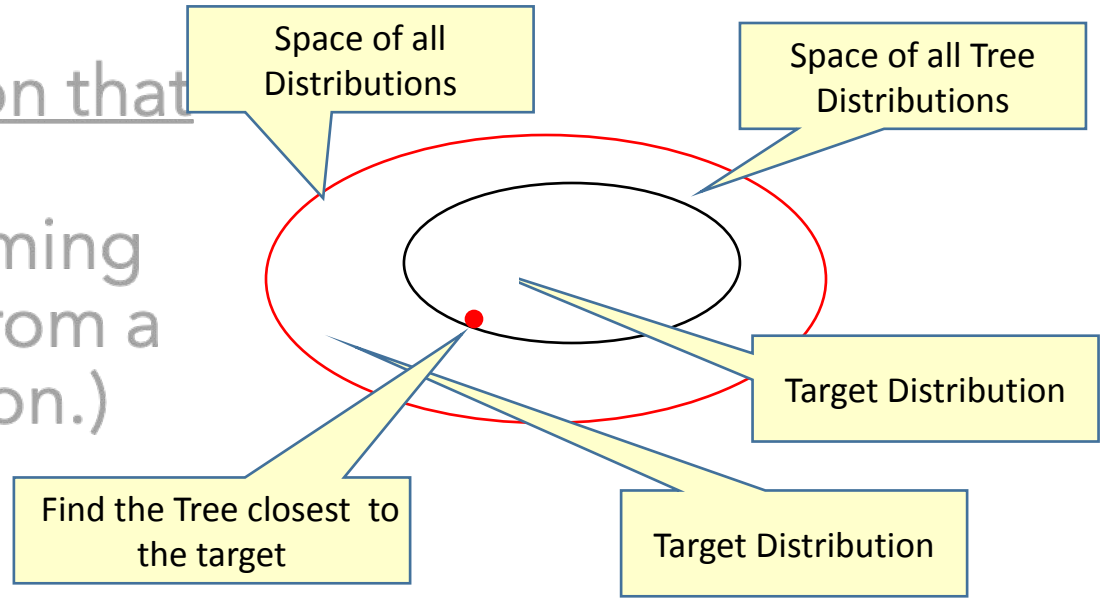
- Now we can see why we had to solve the inference problem first; it is required for learning.



Learning Tree Dependent Distributions

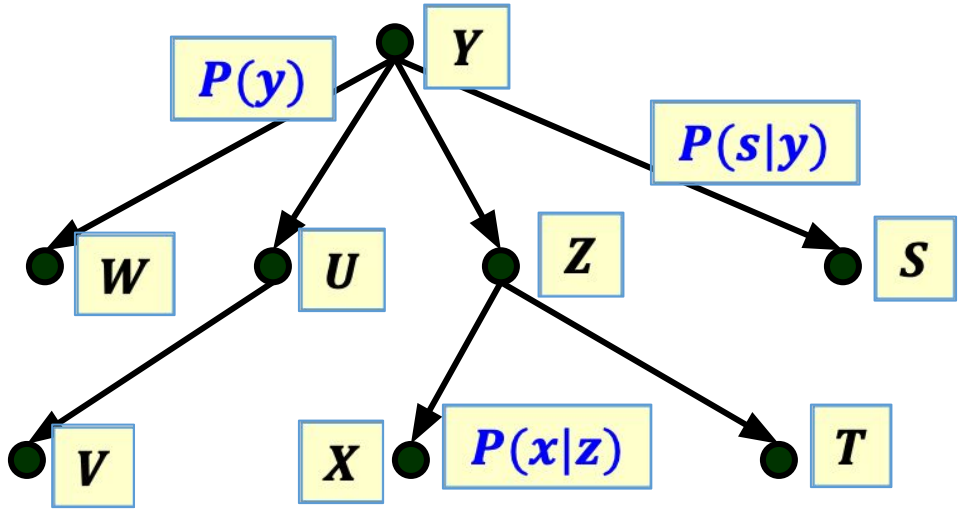
- Learning Problem:
  1. Given data (n tuples) assumed to be sampled from a tree-dependent distribution
     - find the most probable tree representation of the distribution.
  2. Given data (n tuples)
     - find the tree representation that best approximates the distribution (without assuming that the data is sampled from a tree-dependent distribution.)





Learning Tree Dependent Distributions

- Learning Problem:
  1. Given data (n tuples) assumed to be sampled from a tree-dependent distribution
     - find the most probable tree representation of the distribution.
  2. Given data (n tuples)
     - find the tree representation that best approximates the distribution (without assuming that the data is sampled from a tree-dependent distribution.)



The simple minded algorithm for learning a tree dependent distribution requires:
(1) for each tree, compute its likelihood

$$L(T) = P(D|T) =$$

$$= argmax_T \prod_{\{x\}} P_T(x_1, x_2, \ldots, x_n) =$$

$$= argmax_T \prod_{\{x\}} P_T(x_i | Parents(x_i))$$

(2) Find the maximal one

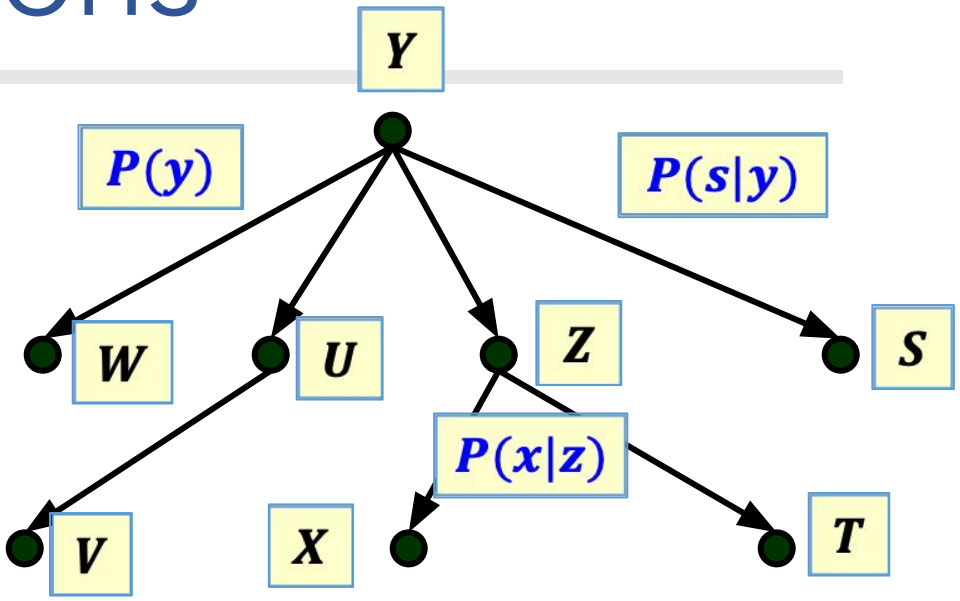
INFERENCE FOR BAYESIAN NETWORKS

Inference

- Inference in Bayesian Networks is generally intractable in the worst case
- Two broad approaches for inference
  - Exact inference
    - Eg. Variable Elimination
  - Approximate inference
    - Eg. Gibbs sampling

Tree Dependent Distributions

- This is a generalization of naïve Bayes.
- Inference Problem:
  - Given the Tree with all the associated probabilities, evaluate the probability of an event $p(x)$?



- $P(x = 1) =$
  $= P(x = 1|z = 1)P(z = 1) + P(x = 1|z = 0)P(z = 0)$
- Recursively, go up the tree:

  $P(z = 1) =$
  $= P(z = 1|y = 1)P(y = 1) + P(z = 1|y = 0)P(y = 0)$

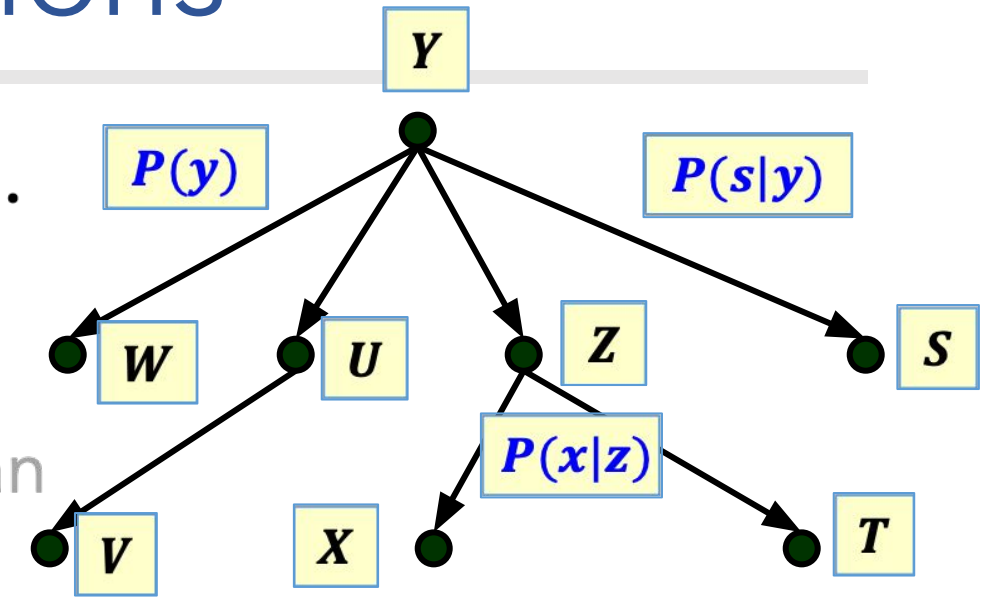
  $P(z = 0) =$
  $= P(z = 0|y = 1)P(y = 1) + P(z = 0|y = 0)P(y = 0)$
- Linear Time Algorithm

Now we have everything in terms of the CPTs (conditional probability tables)

$$P(y, x_1, x_2, \ldots x_n) = p(y) \prod_i P(x_i|Parents(x_i))$$

Tree Dependent Distributions

- This is a generalization of naïve Bayes.
- Inference Problem:
  - Given the Tree with all the associated probabilities, evaluate the probability of an event $p(x, y)$?



- $P(x = 1, y = 0) =$

$$= P(x = 1|y = 0)P(y = 0)$$

- Recursively, go up the tree along the path from x to y:

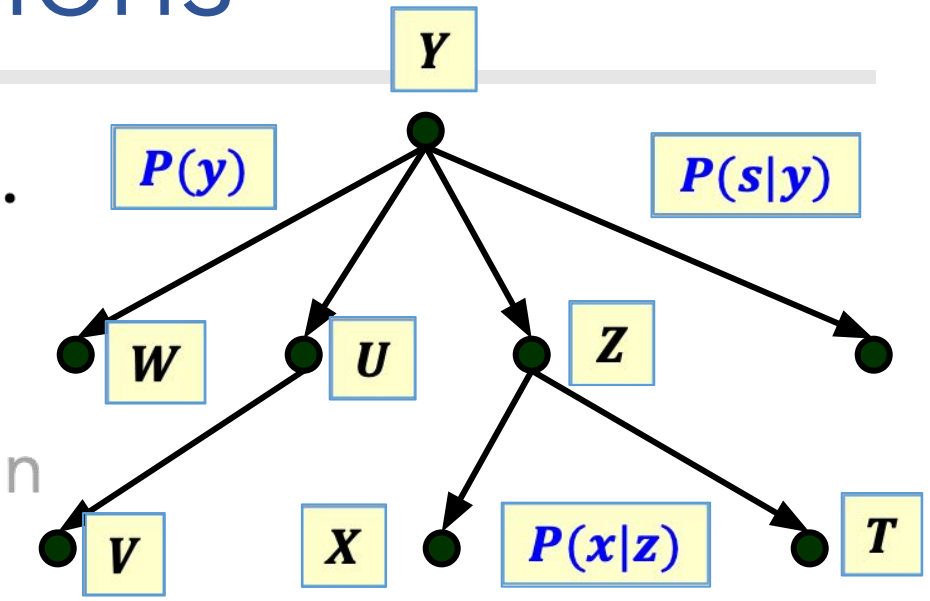
$$P(x = 1|y = 0) = \sum_{z=0,1} P(x = 1|y = 0, z)P(z|y = 0)$$

$$= \sum_{z=0,1} P(x = 1|z)P(z|y = 0)$$

Now we have everything in terms of the CPTs (conditional probability tables)

Tree Dependent Distributions

- This is a generalization of naïve Bayes.
- Inference Problem:
  - Given the Tree with all the associated probabilities, evaluate the probability of an event $p(x, u)$?
  - (No direct path from x to u)



- $P(x = 1, u = 0) = P(x = 1|u = 0)P(u = 0)$
- Let y be a parent of x and u (we always have one)

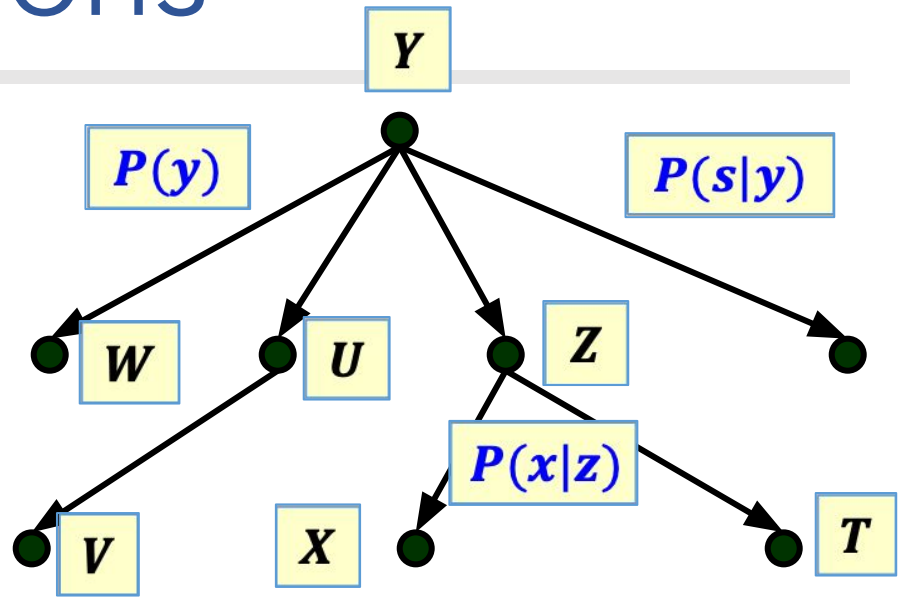
$$P(x = 1|u = 0) = \sum_{y=0,1} P(x = 1|u = 0, y)P(y|u = 0)$$
$$= \sum_{y=0,1} P(x = 1|y)P(y|u = 0)$$

Now we have reduced it to cases we have seen

$$P(y, x_1, x_2, \ldots x_n) = p(y) \prod_i P(x_i|Parents(x_i))$$

Tree Dependent Distributions

- Inference Problem:
  - Given the Tree with all the associated CPTs, we "showed" that we can evaluate the probability of all events efficiently.
  - There are more efficient algorithms
  - The idea was to show that the inference is this case is a simple application of Bayes rule and probability theory.

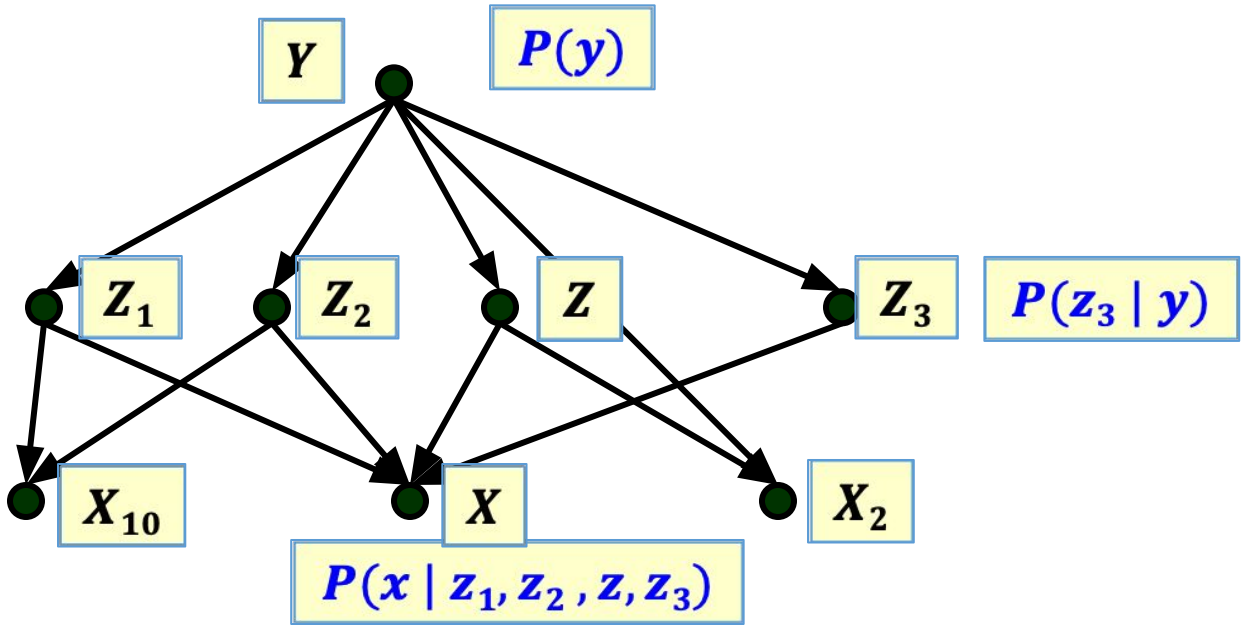


$$P(y, x_1, x_2, \ldots x_n) = p(y) \prod_i P(x_i | Parents(x_i))$$

Things are not so simple in the general case; there are multiple ways to "get" from node A to B, and this has to be accounted for in Inference.

Graphical Models of Probability Distributions

- For general Bayesian Networks
  - The learning problem is hard
  - The inference problem (given the network, evaluate the probability of a given event) is hard (#P Complete)



$$P(y, x_1, x_2, \ldots x_n) = p(y) \prod_i P(x_i | Parents(x_i))$$

Variable Elimination

$$P(x_1, x_2, \ldots, x_n) = \prod_i P(x_i | Parents(x_i))$$

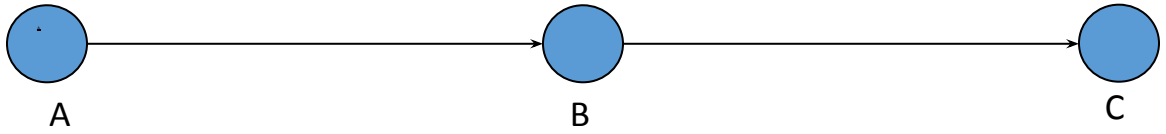
- Suppose the query is $P(X_1)$

$$P(x_1) = \sum_{x_2, \ldots, x_n} P(x_1, x_2, \ldots, x_n)$$

$$P(x_1) = \sum_{x_2} \sum_{x_3} \ldots \sum_{x_n} \prod_i P(x_i | Parents(x_i))$$

- Key Intuition: Move irrelevant terms outside summation and cache intermediate results

Variable Elimination: Example 1



- We want to compute $P(C)$

$$P(C) = \sum_A \sum_B P(A,B,C) = \sum_A \sum_B P(A)P(B|A)P(C|B)$$

$$= \sum_B P(C|B) \sum_A P(A)P(B|A)$$

Let's call this $f_A(B)$

$$= \sum_B P(C|B) f_A(B)$$

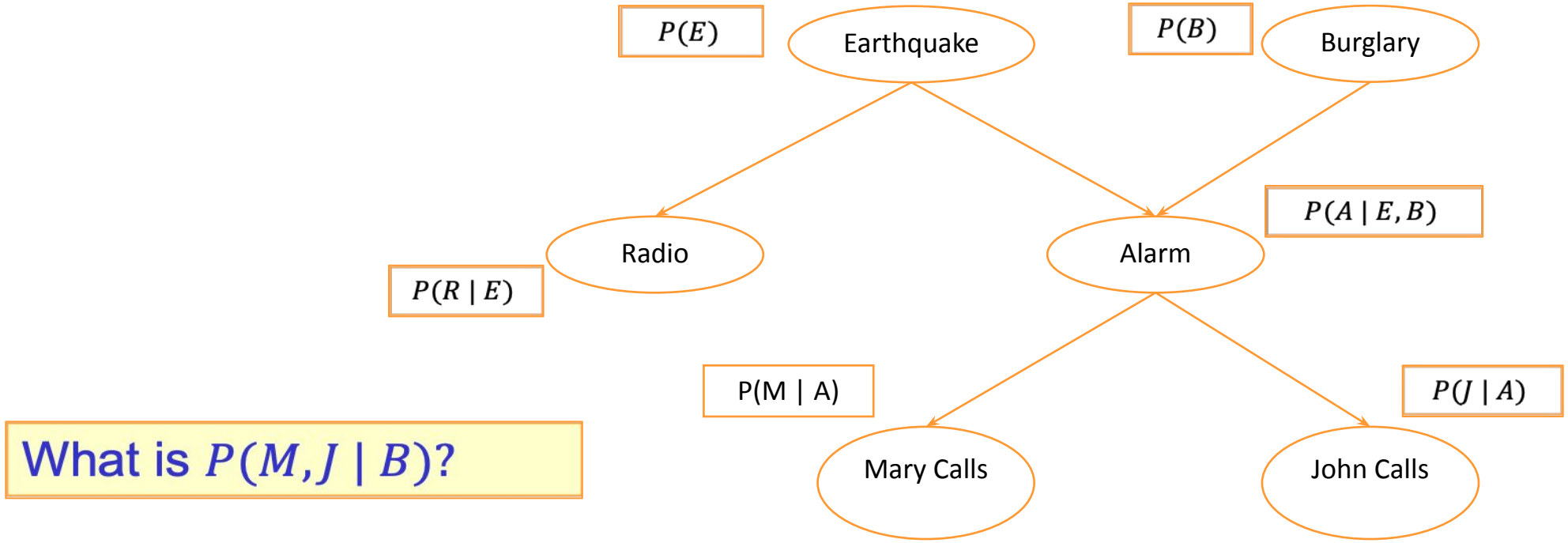
A has been (instantiated and) eliminated

- What have we saved with this procedure?
  - How many multiplications and additions did we perform?

Variable Elimination

- VE is a sequential procedure.
- Given an ordering of variables to eliminate
  - For each variable v that is not in the query
    - Replace it with a new function f_v
      - That is, marginalize v out
- The actual computation depends on the order
- What is the domain and range of f_v?
  - It need not be a probability distribution

Variable Elimination: Example 2



$$P(E,B,A,R,M,J) =$$

$$= P(E)\cdot P(B)\cdot P(R\,|\,E)\cdot P(A\,|\,E,B)\cdot P(M\,|\,A)\cdot P(J\,|\,A)$$

Variable Elimination: Example 2

$$P(E,B,A,R,M,J) = P(E) \cdot P(B) \cdot P(R|E) \cdot P(A|E,B) \cdot P(M|A) \cdot P(J|A)$$

$$P(M,J|B = true) = \frac{P(M,J,B = true)}{P(B = true)}$$

It is sufficient to compute the numerator and normalize

Assumptions (graph; joint representation)

$$P(M,J,B = true) = \sum_{E,A,R} P(E,B = true,A,R,M,J)$$

Elimination order R, A, E

$$= \sum_{E,A,R} P(E) \cdot P(B = true) \cdot P(R|E) \cdot P(A|E,B = true) \cdot P(M|A) \cdot P(J|A)$$

To eliminate R

$$f_R(E) = \sum_{R} P(R|E)$$

Variable Elimination: Example 2

$$P(E,B,A,R,M,J) = P(E) \cdot P(B) \cdot P(R|E) \cdot P(A|E,B) \cdot P(M|A) \cdot P(J|A)$$

$$P(M,J|B = true) = \frac{P(M,J,B = true)}{P(B = true)}$$

It is sufficient to compute the numerator and normalize

$$P(M,J,B = true) = \sum_{E,A,R} P(E,B = true,A,R,M,J)$$

Elimination order A, E

$$= \sum_{E,A} P(E) \cdot P(B = true) \cdot P(A|E,B = true) \cdot P(M|A) \cdot P(J|A) \cdot f_R(E)$$

To eliminate A

$$f_R(E) = \sum_{R} P(R|E)$$

$$f_A(E,M,J) = \sum_{A} P(A|E,B = true) \cdot P(M|A) \cdot P(J|A)$$

Variable Elimination: Example 2

$$P(E,B,A,R,M,J) = P(E) \cdot P(B) \cdot P(R|E) \cdot P(A|E,B) \cdot P(M|A) \cdot P(J|A)$$

$$P(M,J|B = true) = \frac{P(M,J,B = true)}{P(B = true)}$$

It is sufficient to compute the numerator and normalize

$$P(M,J,B = true) = \sum_{E,A,R} P(E,B = true,A,R,M,J)$$

Finally eliminate E

$$= \sum_{E} P(E) \cdot P(B = true) \cdot f_A(E,M,J) \cdot f_R(E)$$

Factors

$$f_R(E) = \sum_{R} P(R|E)$$

$$f_A(E,M,J) = \sum_{A} P(A|E,B = true) \cdot P(M|A) \cdot P(J|A)$$

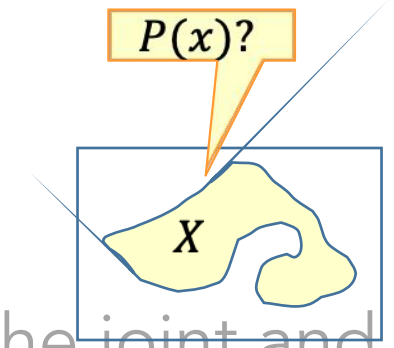
Variable Elimination

- The order in which variables are eliminated matters
  - In the previous example, what would happen if we eliminate E first?
    - The size of the factors would be larger
- Complexity of Variable Elimination
  - Exponential in the size of the factors
  - What about worst case?
    - The worst case is intractable

Approximate Inference

- Basic idea
  - If we had access to a set of examples from the joint distribution, we could just count.

$$E[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})$$

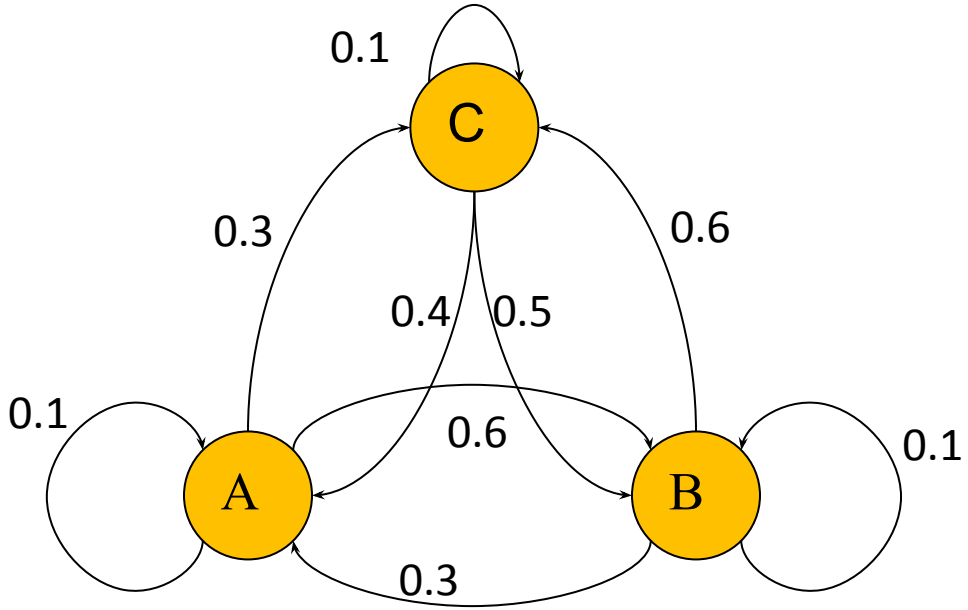


  - For inference, we generate instances from the joint and count
  - How do we generate instances?

Generating instances

- Sampling from the Bayesian Network
  - Conditional probabilities, that is, $P(X|E)$
  - Only generate instances that are consistent with E
- Problems?
  - How many samples? [Law of large numbers]
  - What if the evidence E is a very low probability event?

Detour: Markov Chain Review



Generates a sequence of A, B, C

Defined by initial and transition probabilities

$$P(X_0) \; and \; P(X_{t+1} = i \mid X_t = j)$$

P_{ij} : Time independent transition probability matrix

Stationary Distributions: A vector q is called a stationary distribution if

$$q_j = \sum_i q_i P_{ij}$$

q_i : The probability of being in state i

If we sample from the Markov Chain repeatedly, the distribution over the states converges to the stationary distribution

Markov Chain Monte Carlo

- Our goal: To sample from $P(X|\ e)$
- Overall idea:
  - The next sample is a function of the current sample
  - The samples can be thought of as coming from a Markov Chain whose stationary distribution is the distribution we want
- Can approximate any distribution

Gibbs Sampling

- The simplest MCMC method to sample from

$$P(X = x_1 x_2 \ldots x_n \mid e)$$

- Creates a Markov Chain of samples as follows:
  - Initialize X randomly
  - At each time step, fix all random variables except one.
  - Sample that random variable from the corresponding conditional distribution

Gibbs Sampling

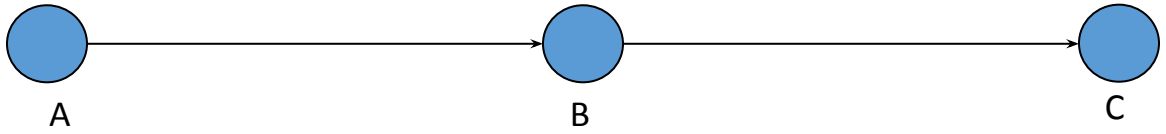
- Algorithm:
  - Initialize X randomly
  - Iterate:
    - Pick a variable X_i uniformly at random
    - Sample $x_i^{(t+1)}$ from $P(x_i|x_1^{(t)}, \ldots, x_{i-1}^{(t)}\ x_{i+1}^{(t)}, \ldots, x_n^{(t)}, e)$
    - $X_k^{(t+1)} = x_k^{(t+1)}$ for all other k
    - This is the next sample

- $X^{(1)}, X^{(2)}, \ldots X^{(t)}$ forms a Markov Chain
- Why is Gibbs Sampling easy for Bayes Nets?
  - $P(x_i|x_{-i}^{(t)}, e)$ is "local"

Gibbs Sampling: Big picture

- Given some conditional distribution we wish to compute, collect samples from the Markov Chain
- Typically, the chain is allowed to run for some time before collecting samples (burn in period)
  - So that the chain settles into the stationary distribution
- Using the samples, we approximate the posterior by counting

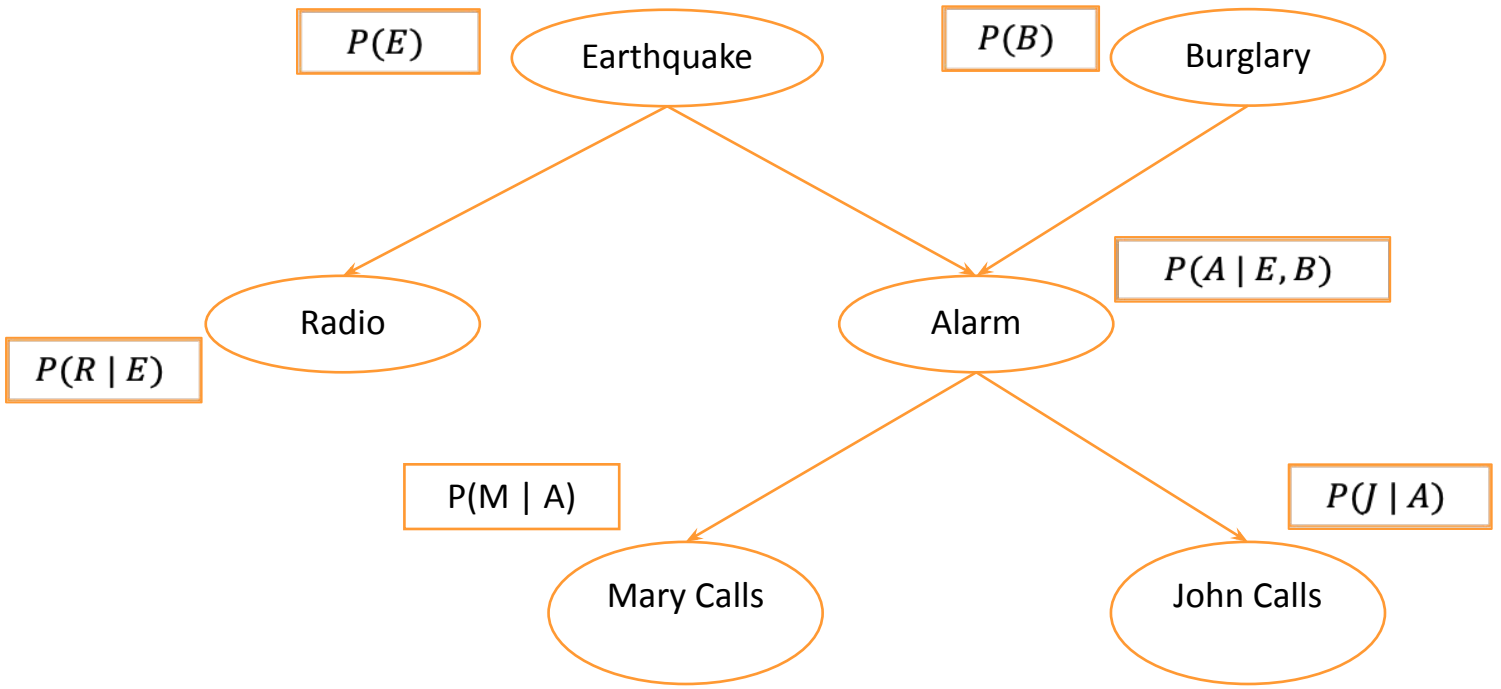
Gibbs Sampling Example 1



- We want to compute $P(C)$:
- Suppose, after burn in, the Markov Chain is at $A = true, B = false, C = false$

1. Pick a $variable \rightarrow B$
2. Draw the new value of B from
   - $P(B \mid A = true, C = false) = P(B \mid A = true)$
   - Suppose $B^{new} = true$
3. Our new sample is $A = true, B = true, C = false$
4. Repeat

Gibbs Sampling Example 2



- Exercise: $P(M, J|B)$?

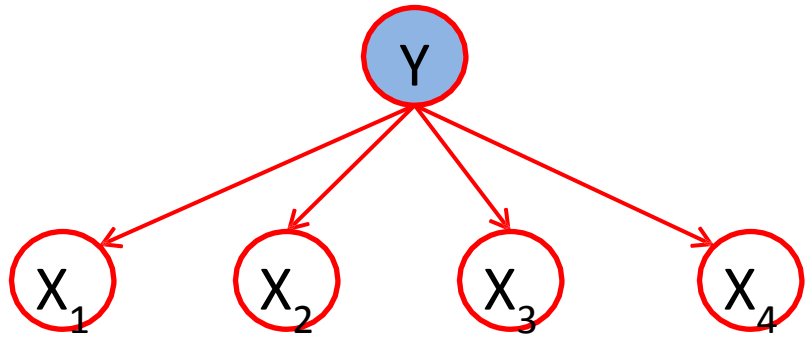
Hidden Markov Models

Bayesian Networks Example

- Naïve Bayes

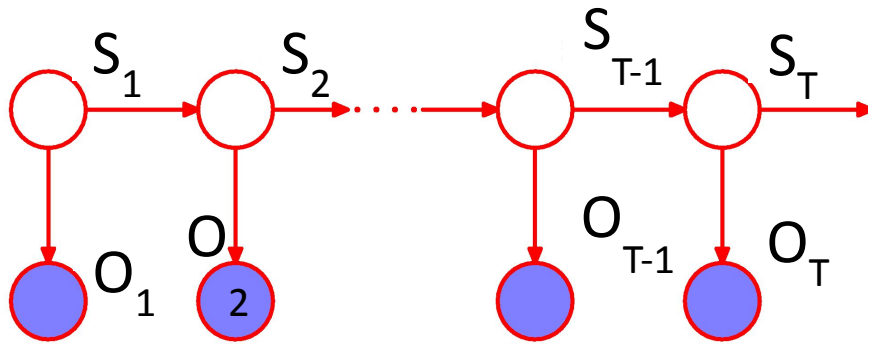
$$X_i \perp X_1,\ldots,X_{i-1},X_{i+1},\ldots, X_n \mid Y$$

$$P(X_1,\ldots,X_n,Y) = P(Y)P(X_1|Y)\ldots P(X_1|Y)$$



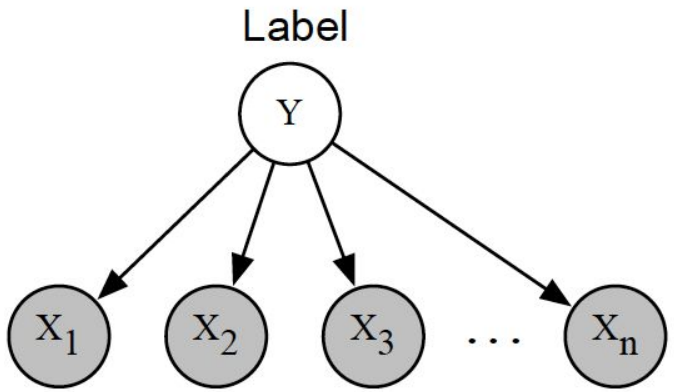
- HMM

$$p(\{S_t\}_{t=1}^T, \{O_t\}_{t=1}^T) = p(S_1)\prod_{t=2}^{T} p(S_t|S_{t-1})\prod_{t=1}^{T} p(O_t|S_t)$$



Naïve Bayes as a *graphical model*

- We can represent a naïve Bayes model with a graph:



Shading in denotes *observed* variables (e.g. what is available at test time)

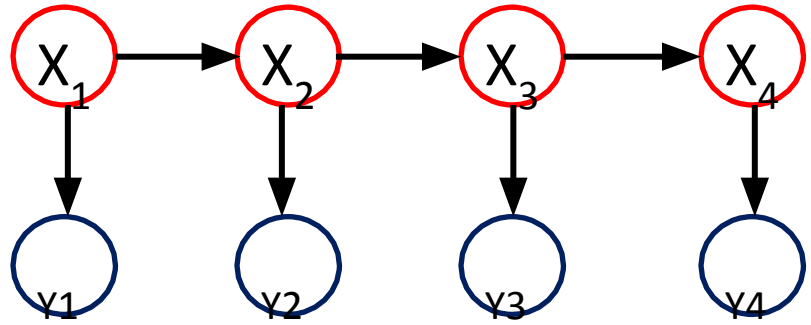
$$\Pr(y, x_1, \ldots, x_n) = \Pr(y) \prod_{i=1}^{n} \Pr(x_i \mid y)$$

- There is a 1-1 mapping between the graph structure and the factorization of the joint distribution

HMM

- Assume that the **joint** distribution on $X_1, X_2, \ldots, X_n$ and $Y_1, Y_2, \ldots, Y_n$ factors as follows:

$$\Pr(x_1, \ldots x_n, y_1, \ldots, y_n) = \Pr(x_1) \Pr(y_1 \mid x_1) \prod_{t=2}^{n} \Pr(x_t \mid x_{t-1}) \Pr(y_t \mid x_t)$$



- To find out where the missile is *now*, we do **marginal inference**:

$$\Pr(x_n \mid y_1, \ldots, y_n)$$

- To find the most likely *trajectory*, we do **MAP (maximum a posteriori) inference**:

$$\arg\max_{\mathbf{x}} \Pr(x_1, \ldots, x_n \mid y_1, \ldots, y_n)$$

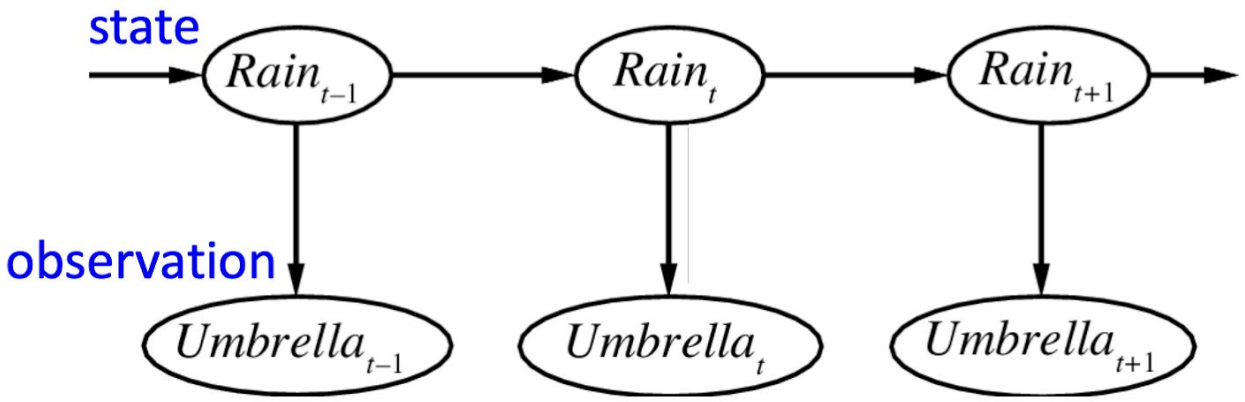
Example Scenario: UmbrellaWorld

- Elspeth Dunsany is an AI researcher at the Canadian company Unitek.
- Richard Feynman is an AI, named after the famous physicist, whose personality he resembles.
- To keep him from escaping, Richard's workstation is not connected to the internet. He knows about rain but has never seen it.
- He has noticed, however, that Elspeth sometimes brings an umbrella to work. He correctly infers that she is more likely to carry an umbrella on days when it rains.

Example Scenario: UmbrellaWorld

Since he has read a lot about rain, Richard proposes a hidden Markov model:

- Rain on day t-1 (R_{t-1}) makes rain on day t (R_t) more likely.
- Elspeth usually brings her umbrella (U_t) on days when it rains (R_t), but not always.



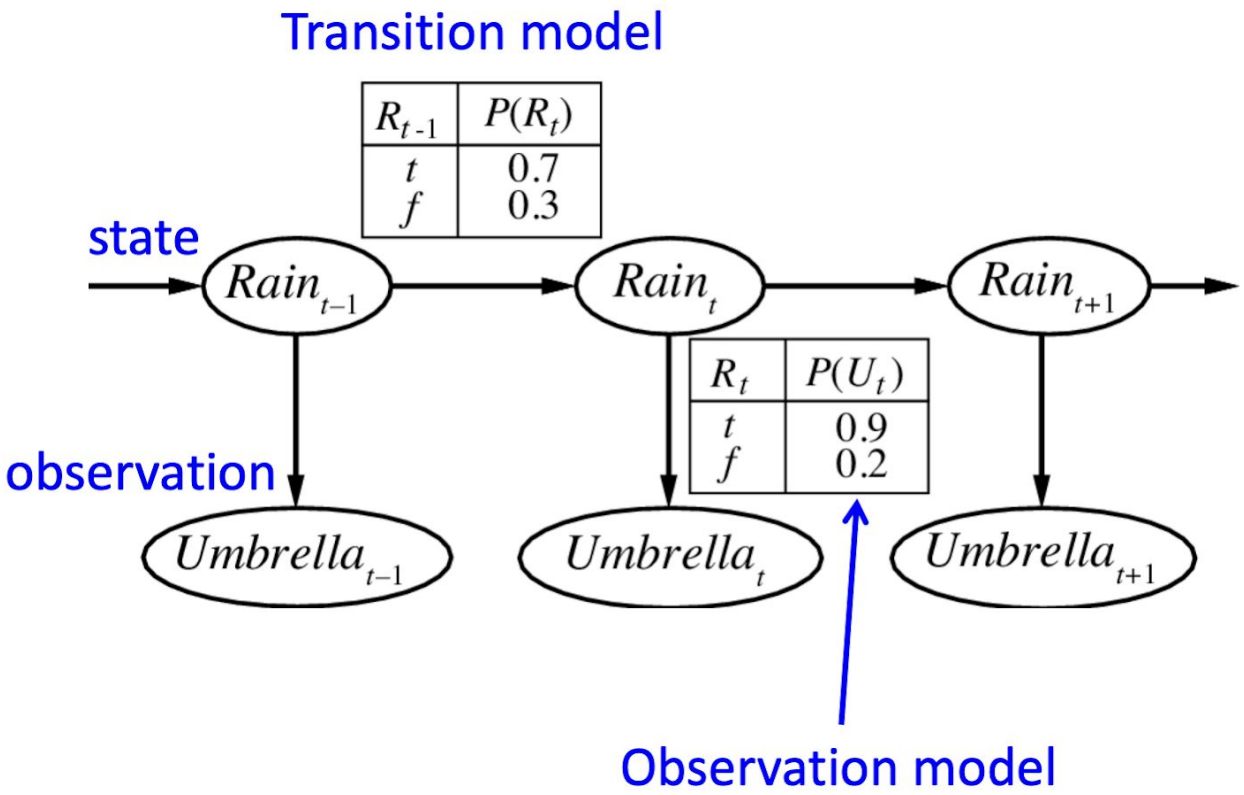
Example Scenario: UmbrellaWorld

- Richard learns that the weather changes on 3 out of 10 days, thus

$$P(R_t|R_{t-1}) = 0.7$$
$$P(R_t|\neg R_{t-1}) = 0.3$$

- He also learns that Elspeth sometimes forgets her umbrella when it's raining, and that she sometimes brings an umbrella when it's not raining. Specifically,

$$P(U_t|R_t) = 0.9$$
$$P(U_t|\neg R_t) = 0.2$$



This is nothing but a Bayes Net!

HMM Inference

Inference by Enumeration in an HMM

Filtering using the Forward Algorithm

Decoding using the Viterbi Algorithm

Inference by Enumeration

To calculate a probability $P(R_2|U_1,U_2)$:

1. **Select:** which other variables do we need, in order to model the relationship among U_1, U_2, and R_2?
   - We need also R_0 and R_1.
2. **Multiply** to compute joint probability:

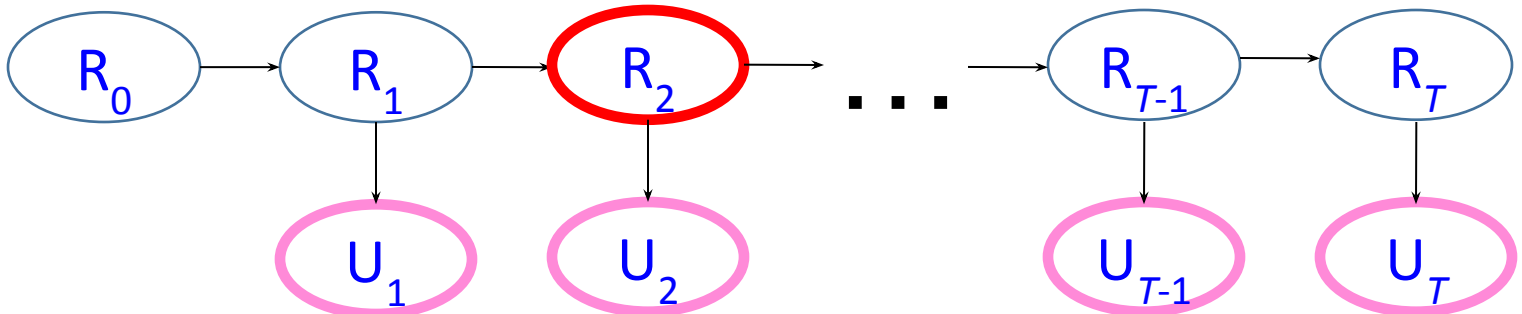
$$P(R_0,R_1,R_2,U_1,U_2) = P(R_0)P(R_1|R_0)P(U_1|R_1)\dots P(U_2|R_2)$$

3. **Add** to eliminate those we don't care about

$$P(R_2,U_1,U_2) = \sum_{R_0,R_1} P(R_0,R_1,R_2,U_1,U_2)$$

4. **Divide:** use Bayes' rule to get the desired conditional

$$P(R_2|U_1,U_2) = P(R_2,U_1,U_2)/P(U_1,U_2)$$



First simplification for HMMs: only enumerate the values of the hidden variables

- Notice: we don't really need to calculate $P(R_0, \neg R_1, R_2, \neg U_1, \neg U_2)$ if we have already observed that U_2 is True!
- First computational simplification for HMMs:
  - Only enumerate the possible values of the hidden variables.
  - Set the observed variables to their observed values.

- **Filtering with binary hidden variables**: enumerate $(R_0, \ldots, R_T)$, complexity is $2^{T+1} = \mathcal{O}\{2^T\}$.
- **Filtering with N-ary hidden variables:** If each of the variables R_t has N possible values, instead of only 2 possible values, then the inference complexity would be $\mathcal{O}\{N^T\}$.

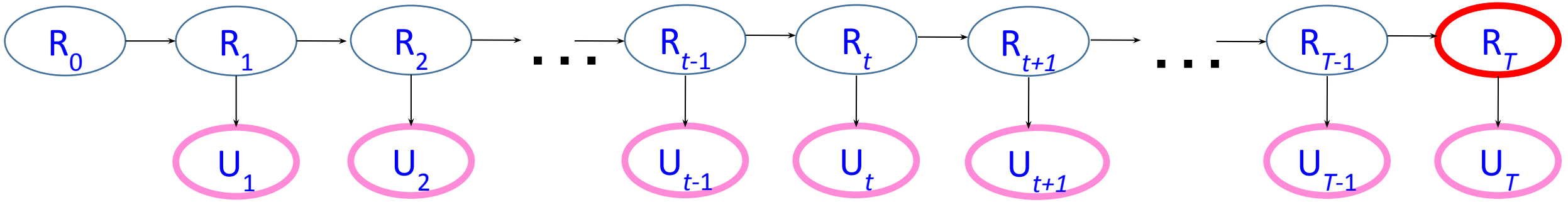
Inference complexity in an HMM

- $\mathcal{O}\{N^T\}$ is still a lot. Can we do better?
- For a general Bayes net, no. Bayes net inference, in an arbitrary Bayes net, is NP-complete.
- For an HMM, yes, we can do better.

The Forward Algorithm (for $\mathrm{P}(R_T|U_1, \ldots, U_T)$)

- **Initialize**: look up the value of $\mathrm{P}(R_0)$.
- **Iterate**: for $1 \leq t \leq T$:
  - **Multiply**:
    $\mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t) = \mathrm{P}(R_{t-1}, U_1, \ldots, U_{t-1})\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$
    for the N^2 combinations of R_{t-1} and R_t.
  - **Add:**
    $$\mathrm{P}(R_t, U_1, \ldots, U_t) = \sum_{R_{t-1}} \mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t)$$
    for the N possible values of R_t.

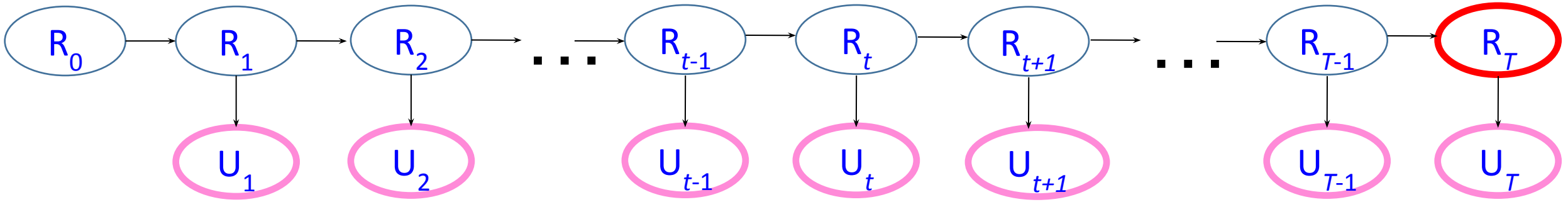
When t=1, this is just $\mathrm{P}(R_0)$



The Forward Algorithm (for $\mathrm{P}(R_T|U_1, \ldots, U_T)$)

- **Initialize**: look up the value of $\mathrm{P}(R_0)$.
- **Iterate**: for $1 \leq t \leq T$:
  - **Multiply**:
    $\mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t) = \mathrm{P}(R_{t-1}, U_1, \ldots, U_{t-1})\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$
    for the N^2 combinations of R_{t-1} and R_t.
  - **Add:**
    $$\mathrm{P}(R_t, U_1, \ldots, U_t) = \sum_{R_{t-1}} \mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t)$$
    for the N possible values of R_t.

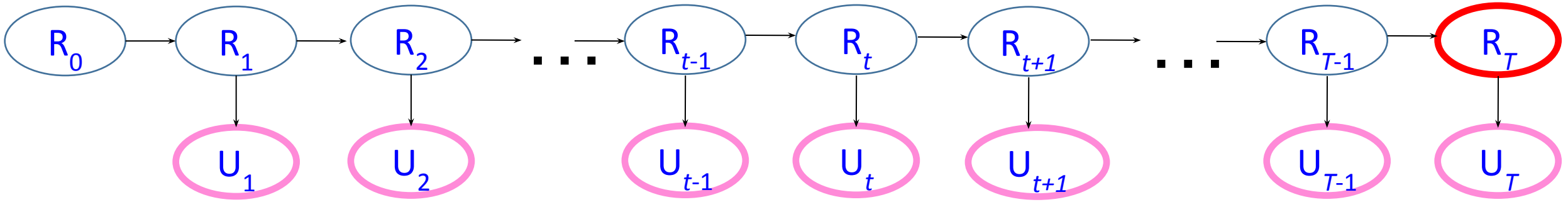
When we move to the next value of t…



The Forward Algorithm (for $\mathrm{P}(R_T|U_1, \ldots, U_T)$)

- **Initialize**: look up the value of $\mathrm{P}(R_0)$.
- **Iterate**: for $1 \leq t \leq T$:
  - **Multiply**:
    $\mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t) = \mathrm{P}(R_{t-1}, U_1, \ldots, U_{t-1})\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$
    for the N^2 combinations of R_{t-1} and R_t.
  - **Add:**
    $$\mathrm{P}(R_t, U_1, \ldots, U_t) = \sum_{R_{t-1}} \mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t)$$
    for the N possible values of R_t.

… and so on, until we reach t=T…

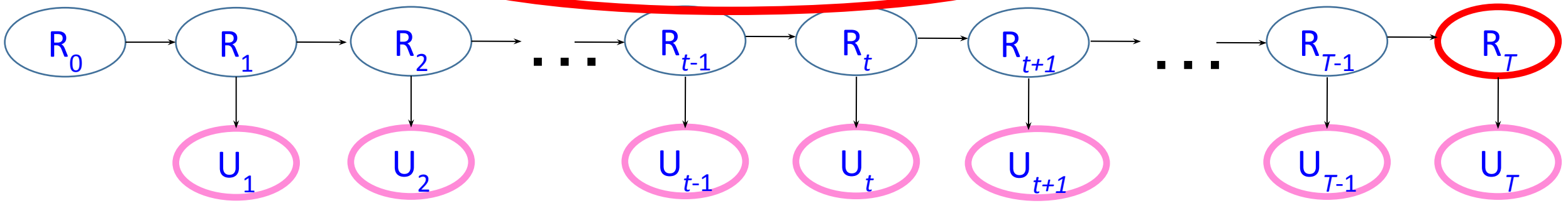


The Forward Algorithm

- **Initialize**: look up the value of $\mathrm{P}(R_0)$.
- **Iterate**: for $1 \leq t \leq T$:
  - **Multiply**:
    $\mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t) = \mathrm{P}(R_{t-1}, U_1, \ldots, U_{t-1})\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$
    for the N^2 combinations of R_{t-1} and R_t.
  - **Add:**

$$\mathrm{P}(R_t, U_1, \ldots, U_t) = \sum_{R_{t-1}} \mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t)$$

    for the N possible values of R_t.
- **Terminate**: $\mathrm{P}(R_T|U_1, \ldots, U_T) = \frac{\mathrm{P}(R_T, U_1, \ldots, U_T)}{\mathrm{P}(U_1, \ldots, U_t)}$



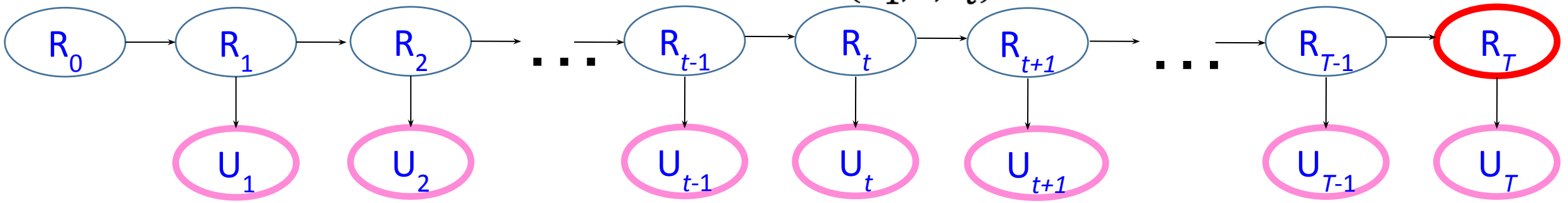
The Forward Algorithm

Complexity

- **Initialize**: look up the value of $\mathrm{P}(R_0)$. → $\mathcal{O}\{N\}$
- **Iterate**: for $1 \leq t \leq T$:
  - **Multiply**:
    $\mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t) = \mathrm{P}(R_{t-1}, U_1, \ldots, U_{t-1})\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$
    for the N^2 combinations of R_{t-1} and R_t. → $\mathcal{O}\{N^2T\}$
  - **Add:**

$$\mathrm{P}(R_t, U_1, \ldots, U_t) = \sum_{R_{t-1}} \mathrm{P}(R_{t-1}, R_t, U_1, \ldots, U_t)$$

    for the N possible values of R_t. → $\mathcal{O}\{N^2T\}$
- **Terminate**: $\mathrm{P}(R_T|U_1, \ldots, U_T) = \frac{\mathrm{P}(R_T, U_1, \ldots, U_T)}{\mathrm{P}(U_1, \ldots, U_t)}$ → $\mathcal{O}\{N\}$



Example: Filtering in UmbrellaWorld

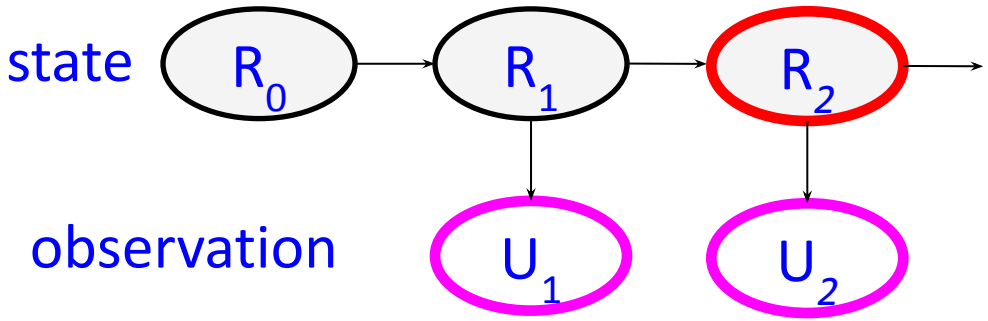
Richard notices that Ellie brought her umbrella today (U_2) but not yesterday ($\neg U_1$). Is it raining today? What is $P(R_2|\neg U_1, U_2)$?

Initialize:

$$P(R_0) = \frac{1}{2}$$

$$P(\neg R_0) = \frac{1}{2}$$

Transition model



Transition probabilities

	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

Observation probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

Example: Filtering in UmbrellaWorld

Iterate $t = 1$:

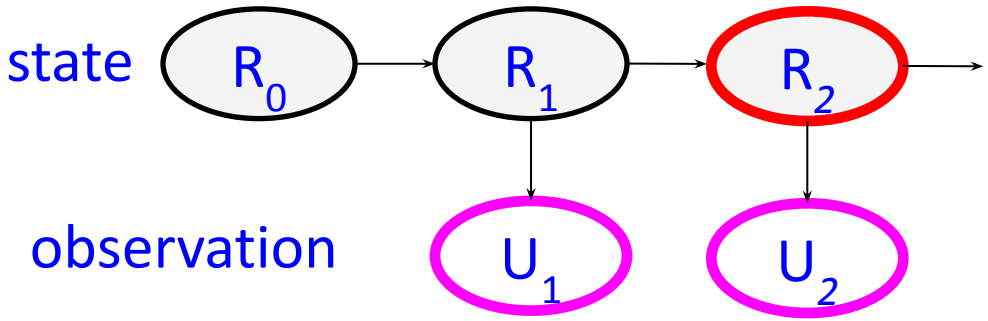
Multiply:

$$P(\neg R_0, \neg R_1, \neg U_1) = (0.5)(0.7)(0.8) = 0.28$$
$$P(\neg R_0, R_1, \neg U_1) = (0.5)(0.3)(0.1) = 0.015$$
$$P(R_0, \neg R_1, \neg U_1) = (0.5)(0.3)(0.8) = 0.12$$
$$P(R_0, R_1, \neg U_1) = (0.5)(0.7)(0.1) = 0.035$$

Add:

$$P(\neg R_1, \neg U_1) = 0.28 + 0.12 = 0.4$$
$$P(R_1, \neg U_1) = 0.015 + 0.035 = 0.05$$

Transition model



Transition probabilities

	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

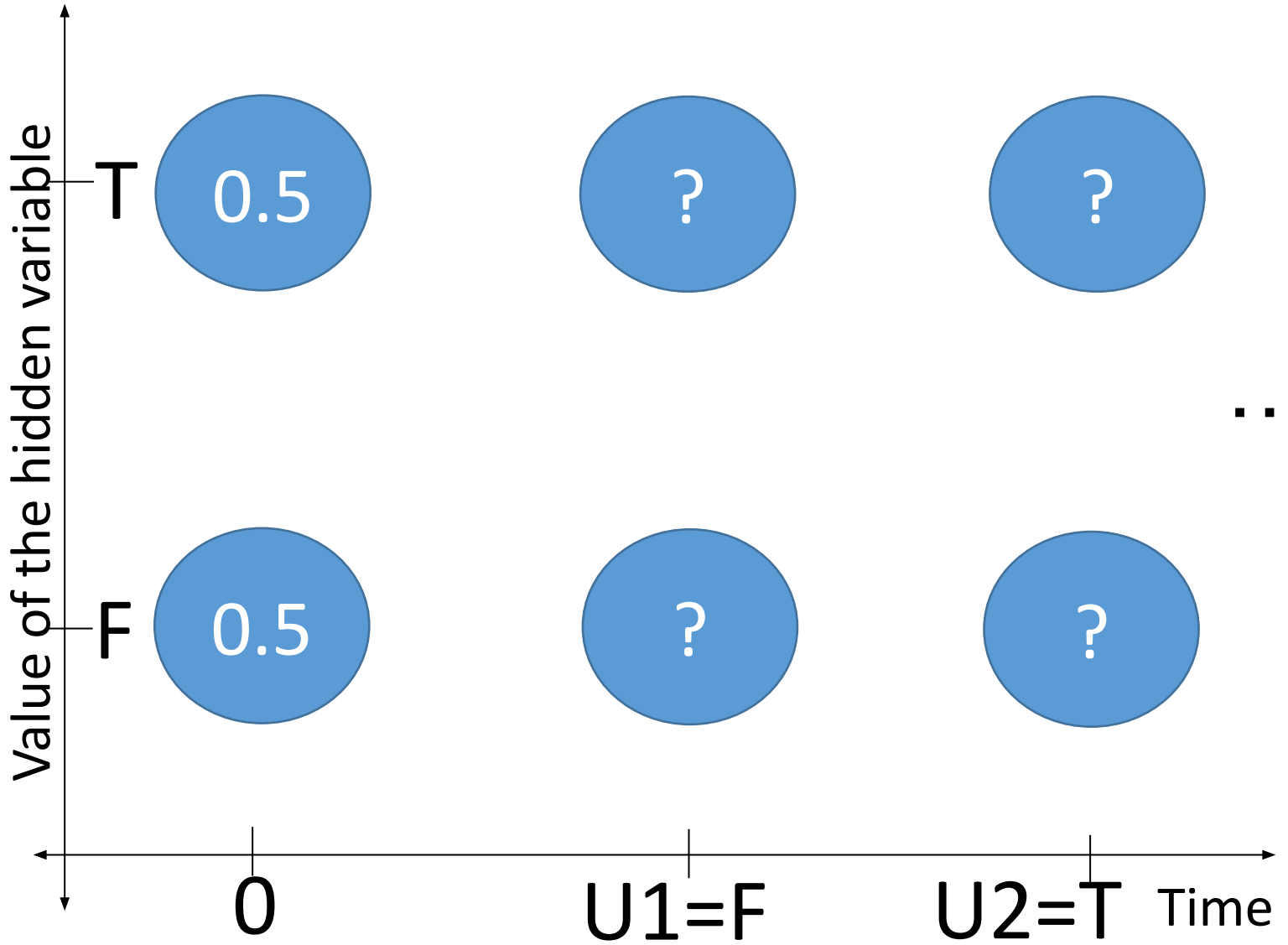
Observation probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

Forward Algorithm: The Trellis

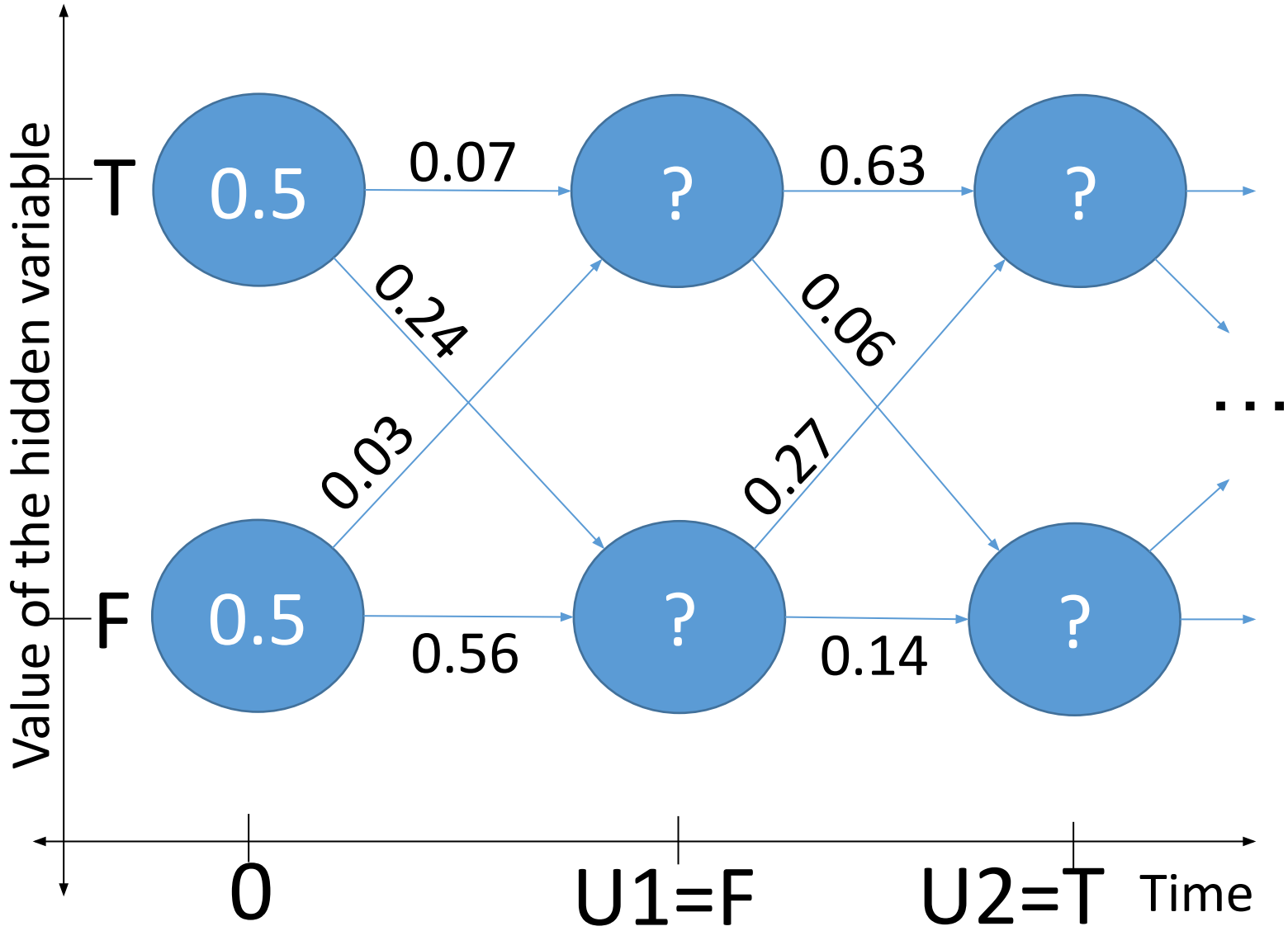
We can visualize the forward algorithm using a TRELLIS:

- Node = a value of the hidden variable at a given time
- Numerical value of the node = probability that the hidden variable takes that value



Forward Algorithm: The Trellis

- Edge = a possible transition from R_{t-1} to R_t
- Numerical value of the edge = $\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$

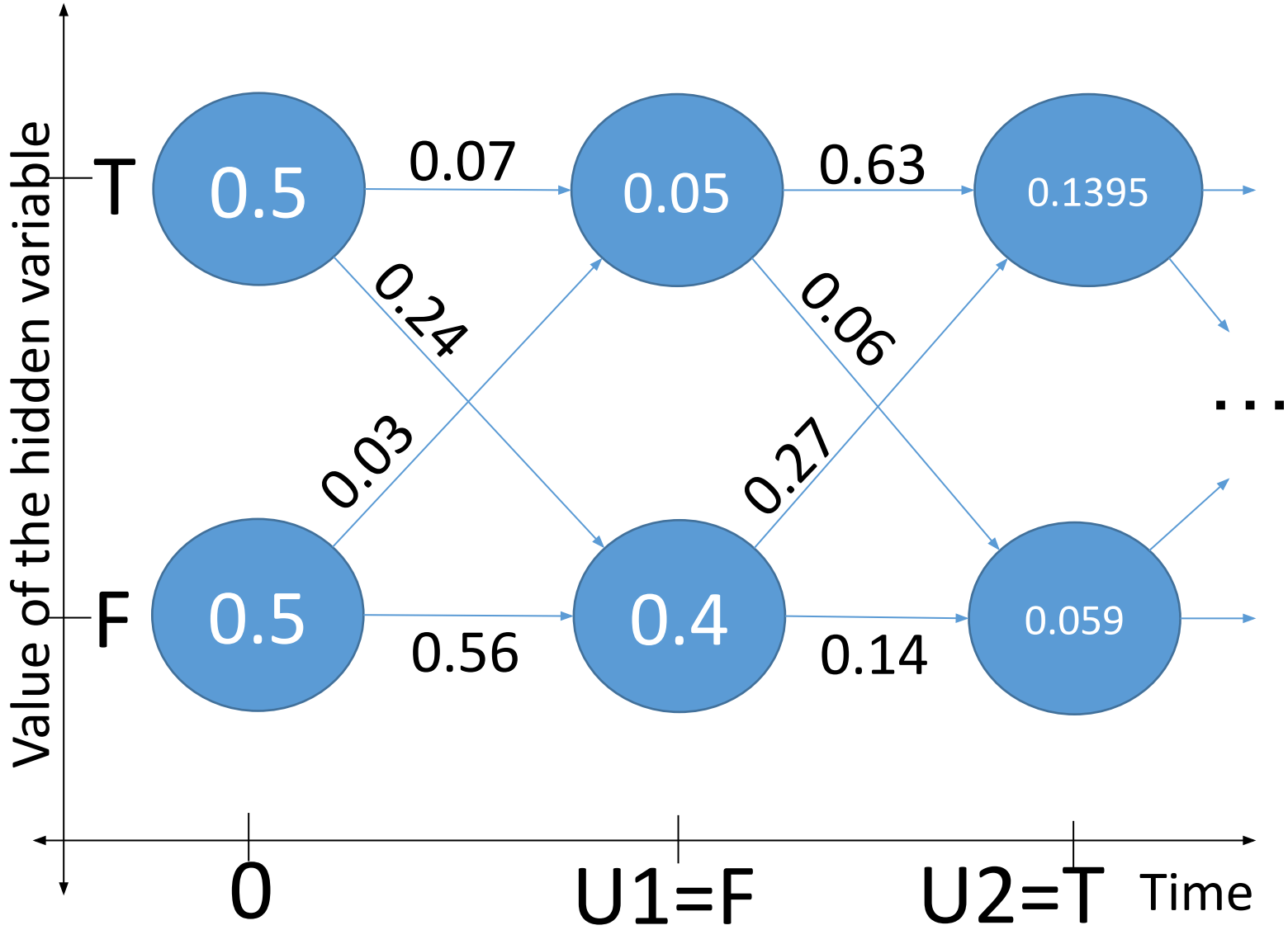


Forward Algorithm: The Trellis

- v_{it} = value of i^{th} node at time t
- e_{ijt} = edge connecting node $v_{i,t-1}$ to v_{jt}

Forward algorithm is just:

$$v_{jt} = \sum_i v_{i,t-1} e_{ijt}$$



Example: Filtering in UmbrellaWorld

Iterate $t = 2$:

Multiply:

$$P(\neg R_1, \neg R_2, \neg U_1, U_2) = (0.4)(0.7)(0.2) = 0.056$$
$$P(\neg R_1, R_2, \neg U_1, U_2) = (0.4)(0.3)(0.9) = 0.108$$
$$P(R_1, \neg R_2, \neg U_1, U_2) = (0.05)(0.3)(0.2) = 0.003$$
$$P(R_1, R_2, \neg U_1, U_2) = (0.05)(0.7)(0.9) = 0.0315$$

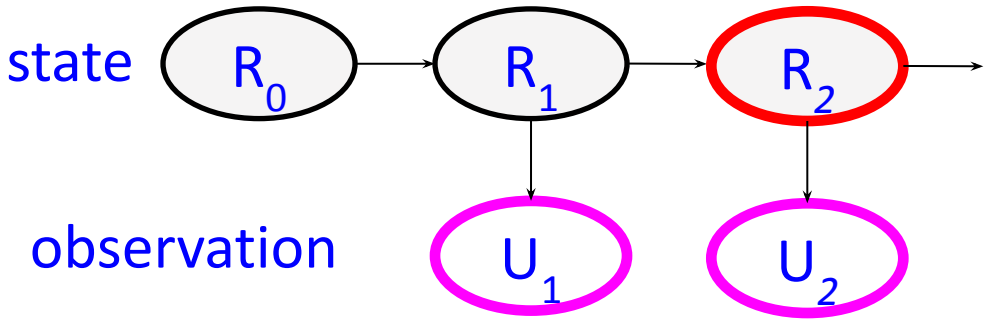
Add:

$$P(\neg R_2, \neg U_1, U_2) = 0.056 + 0.003 = 0.059$$
$$P(R_2, \neg U_1, U_2) = 0.108 + 0.0315 = 0.1395$$

Terminate:

$$P(R_2 | \neg U_1, U_2) = \frac{0.1395}{0.1395 + 0.059} = 0.7$$

Transition model



Transition probabilities

	R_t = T	R_t = F
R_{t-1} = T	0.7	0.3
R_{t-1} = F	0.3	0.7

Observation probabilities

	U_t = T	U_t = F
R_t = T	0.9	0.1
R_t = F	0.2	0.8

Forward Algorithm: The Trellis

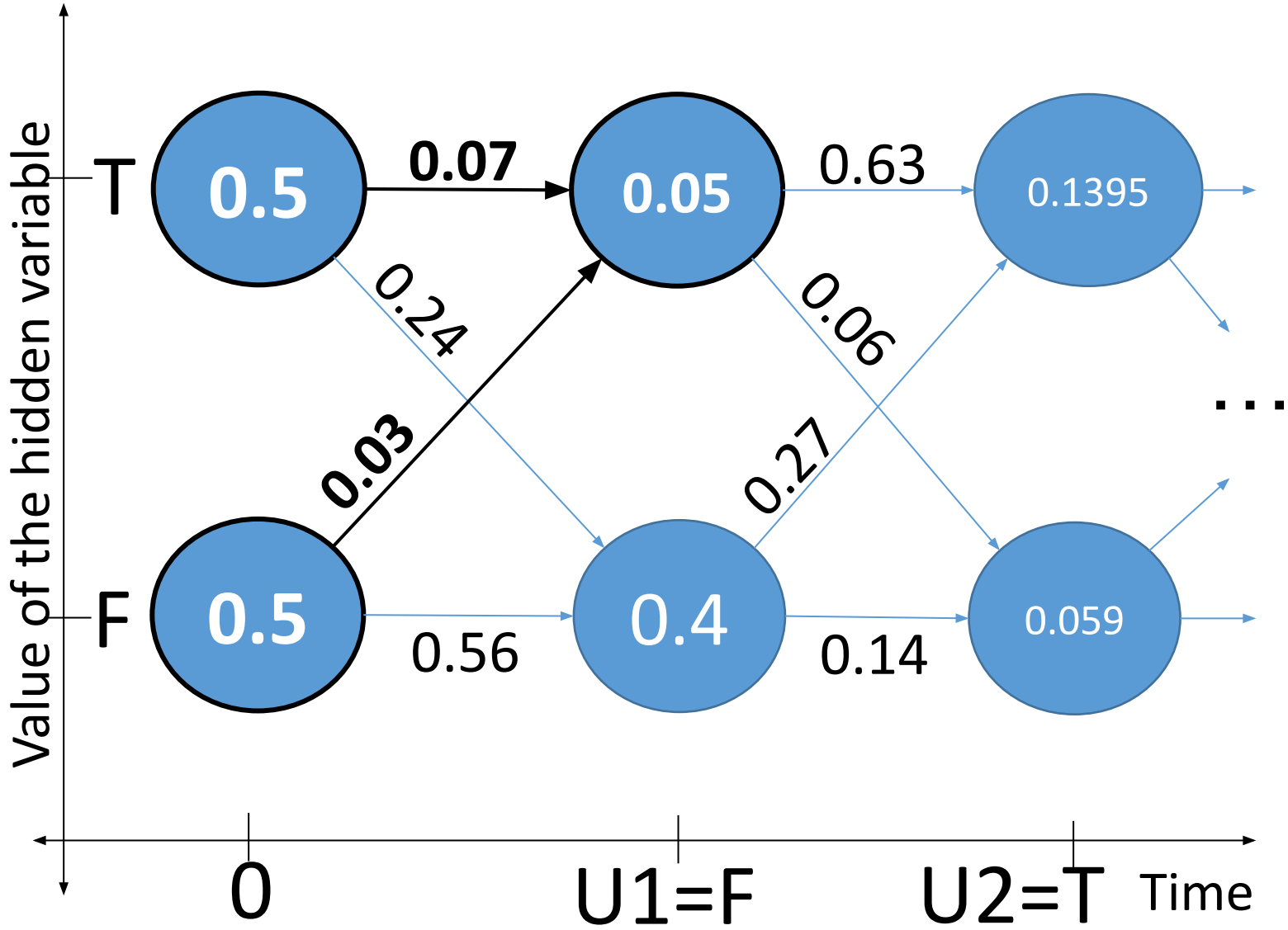
- v_{it} = value of i[th] node at time t
- e_{ijt} = edge connecting node $v_{i,t-1}$ to v_{jt}

Forward algorithm is just:

$$v_{jt} = \sum_i v_{i,t-1} e_{ijt}$$

Terminate:

$$P(R_2 | \neg U_1, U_2) = \frac{0.1395}{0.1395 + 0.059} = 0.7$$



Forward Algorithm vs. Viterbi Algorithm

- Forward Algorithm
  - Goal: efficiently compute $\mathrm{P}(R_T|U_1, \ldots, U_T)$
  - Complexity $\mathcal{O}\{N^2T\}$
  - Key equation: $v_{jt} = \sum_i v_{i,t-1} e_{ijt}$
- Viterbi Algorithm
  - Goal: efficiently find the values of
    $$R_0^*, \ldots, R_T^* = \operatorname{argmax} \mathrm{P}(R_0, \ldots, R_T|U_1, \ldots, U_T)$$
  - Complexity $\mathcal{O}\{N^2T\}$
  - Key equation: $v_{jt} = \max_i v_{i,t-1} e_{ijt}$
  - Back-pointer: $i^*(j,t) = \operatorname{argmax}_i v_{i,t-1} e_{ijt}$

Viterbi Algorithm: Key concepts

- Goal: efficiently find the values of

$$R_0^*, \ldots, R_T^* = \text{argmax}\, \mathrm{P}(R_0, \ldots, R_T | U_1, \ldots, U_T)$$

- To do that efficiently, we need to keep track of TWO pieces of information at each node v_{jt}:
  - **Path Cost**: Probability of the best path until node j at time t

$$v_{jt} = \max_i \max_{R_0,\ldots,R_{t-2}} P(R_0, U_1, R_1, \ldots, U_{t-1}, R_{t-1} = i, U_t)$$

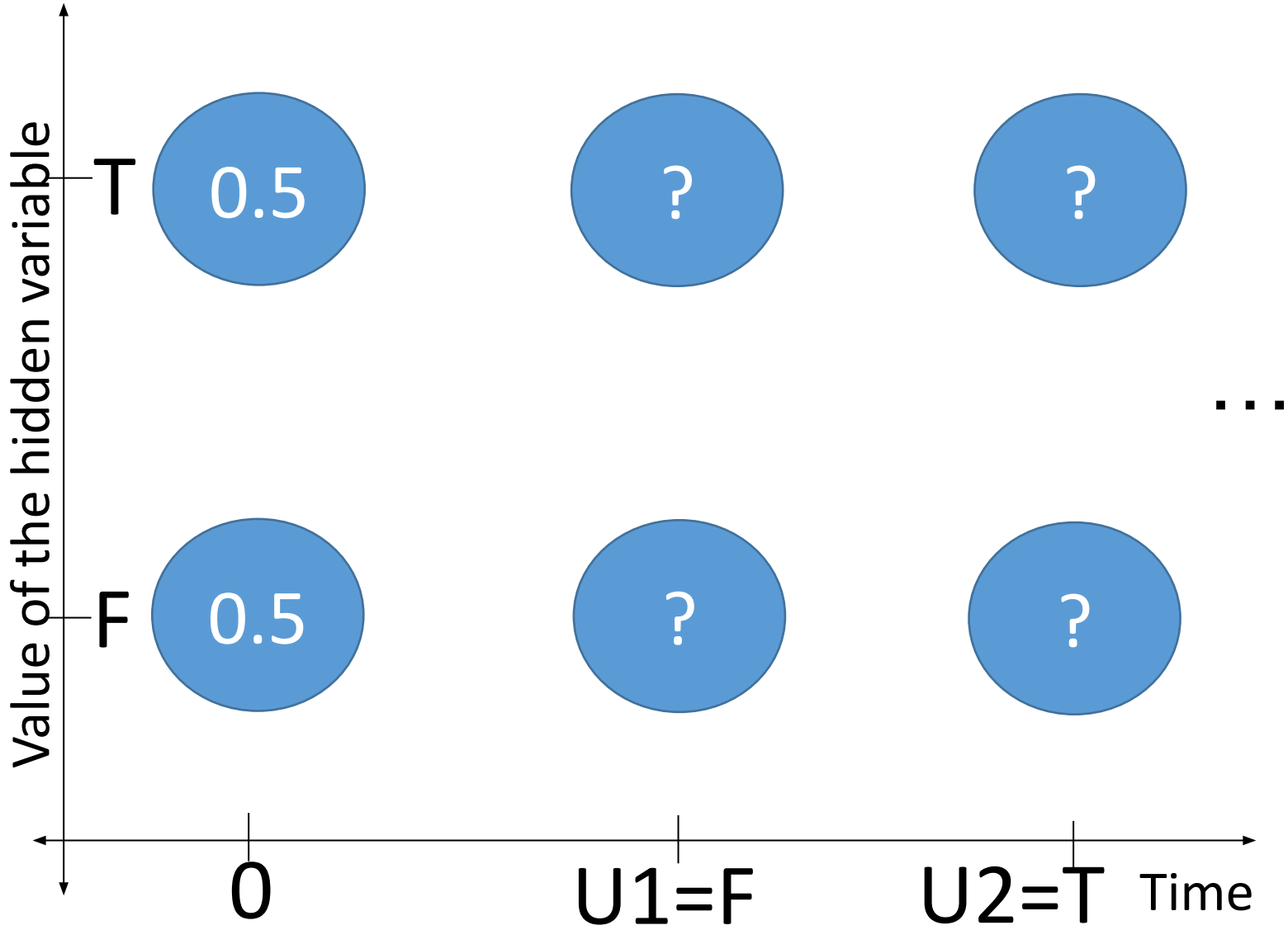
  - **Backpointer**: which node, i, precedes node j on the best path?

$$i^*(j,t) = \text{argmax}_i \max_{R_0,\ldots,R_{t-2}} P(R_0, U_1, R_1, \ldots, U_{t-1}, R_{t-1} = i, U_t)$$

Viterbi Algorithm: The Trellis

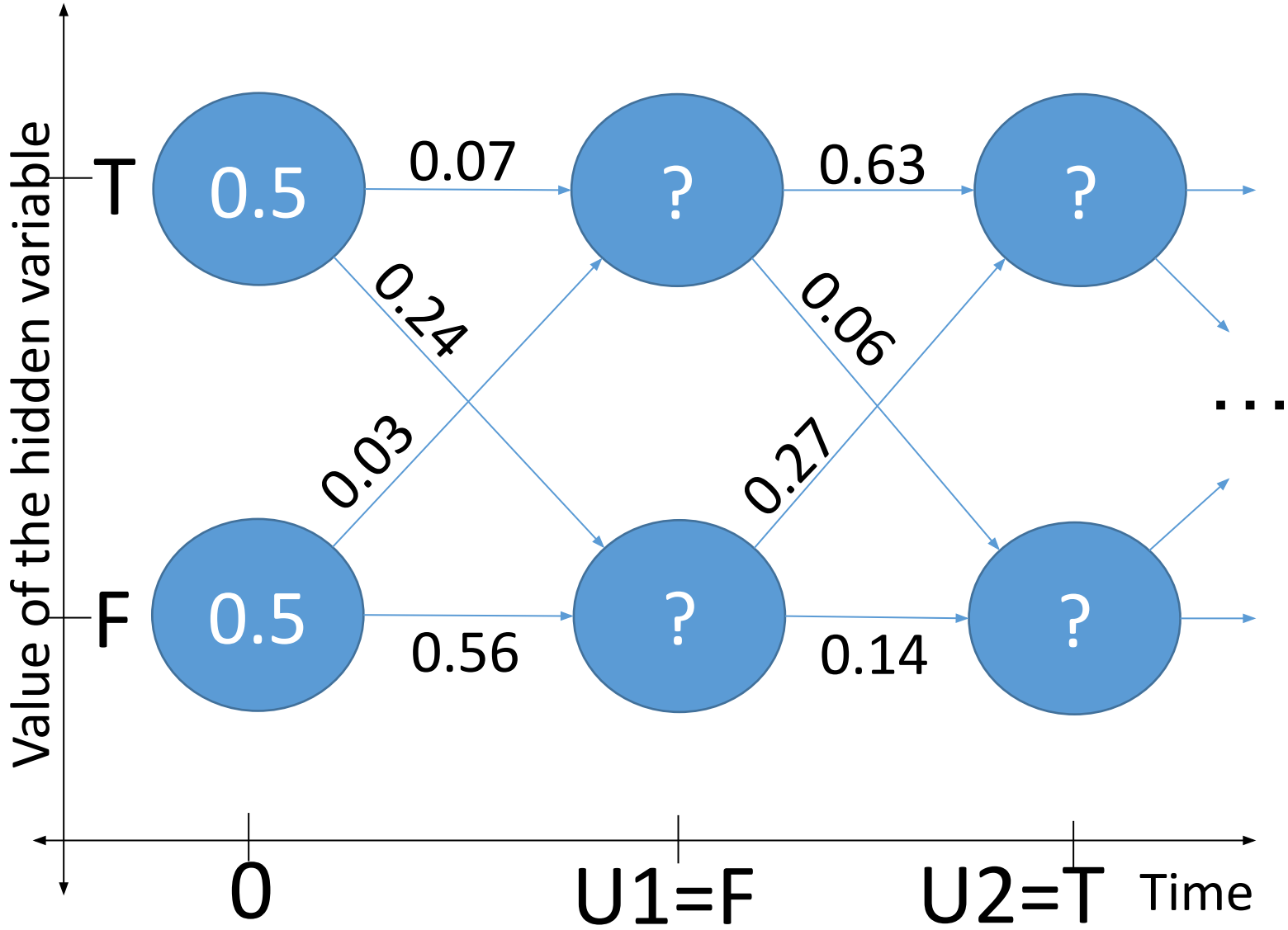
We can visualize the Viterbi algorithm using a TRELLIS:

- Node = a value of the hidden variable at a given time
- Numerical value of the node = probability that the hidden variable takes that value



Viterbi Algorithm: The Trellis

- Edge = a possible transition from R_{t-1} to R_t
- Numerical value of the edge = $\mathrm{P}(R_t|R_{t-1})\mathrm{P}(U_t|R_t)$



Viterbi Algorithm: The Trellis

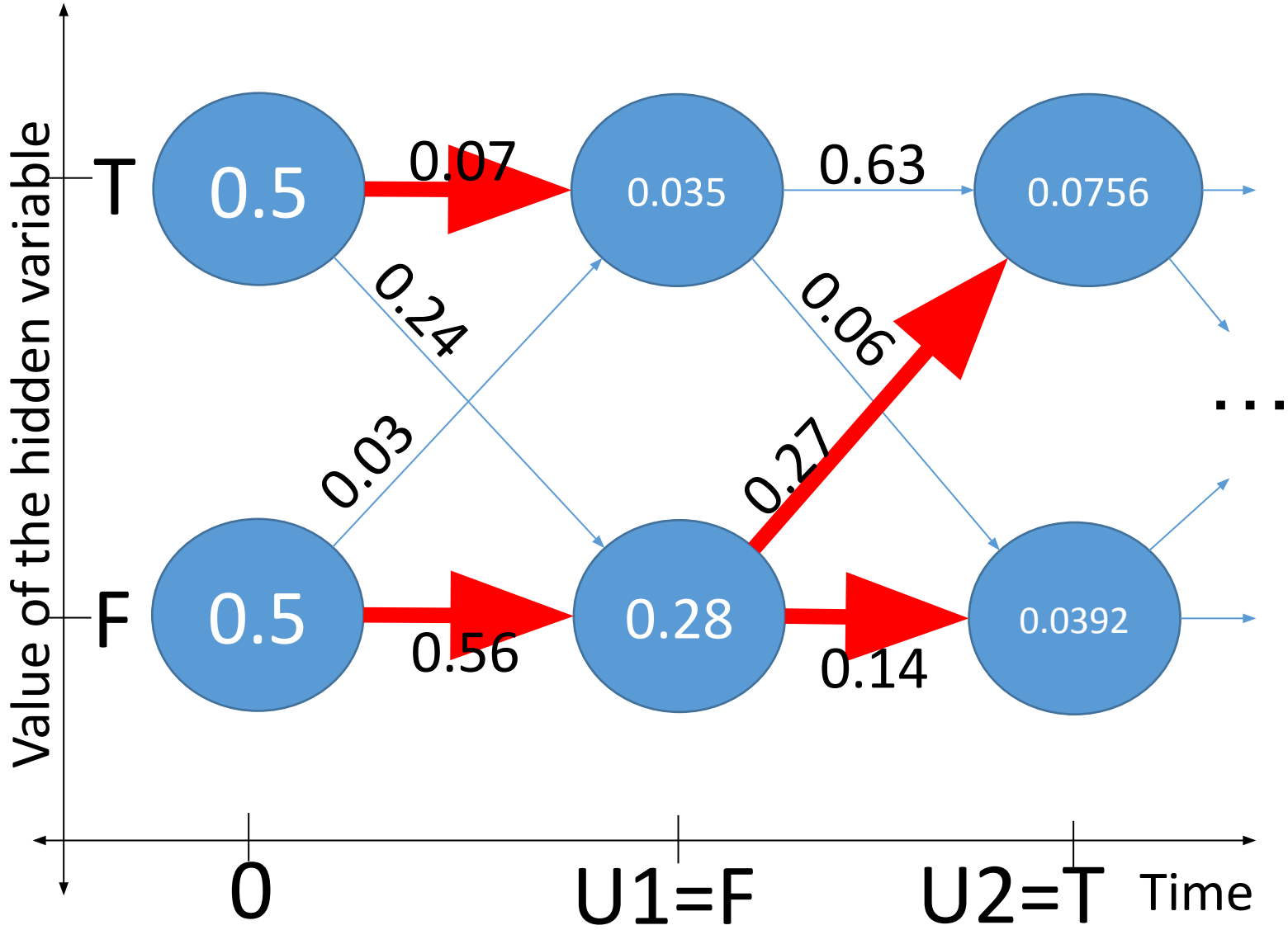
- v_{it} = value of ith node at time t
- e_{ijt} = edge connecting node $v_{i,t-1}$ to v_{jt}

Viterbi algorithm is:

$$v_{jt} = \max_i v_{i,t-1} e_{ijt}$$

Backpointer is:

$$i^*(j,t) = \operatorname{argmax}_i v_{i,t-1} e_{ijt}$$



Viterbi Algorithm: Termination

- Choose the node with the largest final value

$$\max_j v_{jT} = \max_{R_0,\ldots,R_T} P(R_0, U_1, \ldots, U_T, R_T)$$

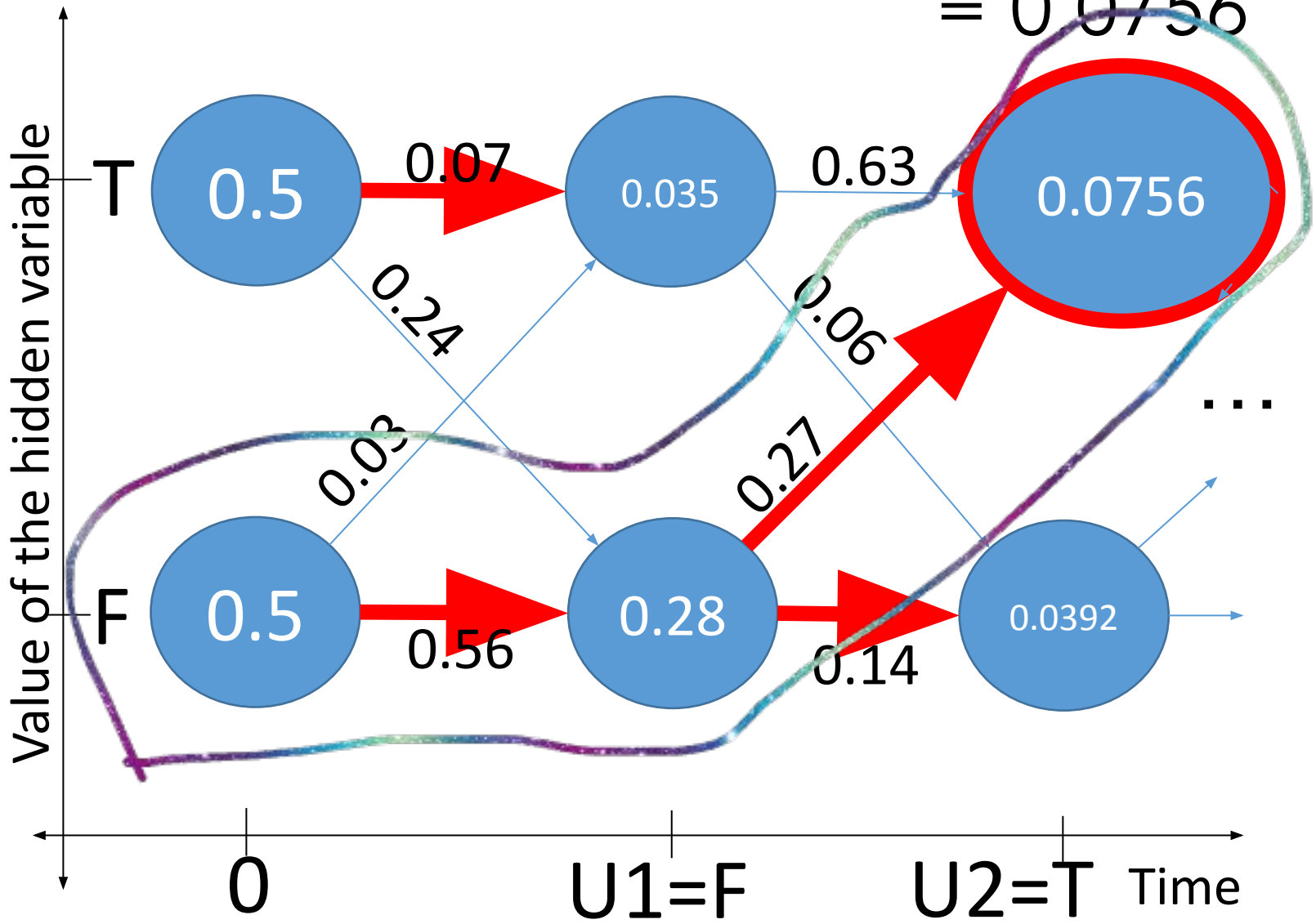
- Trace its backpointers to find

$$R_0^*, \ldots, R_T^*$$

Viterbi Algorithm: Termination

Best path probability = 0.0756

Best path: $\neg R_0 \neg R_1 R_2$

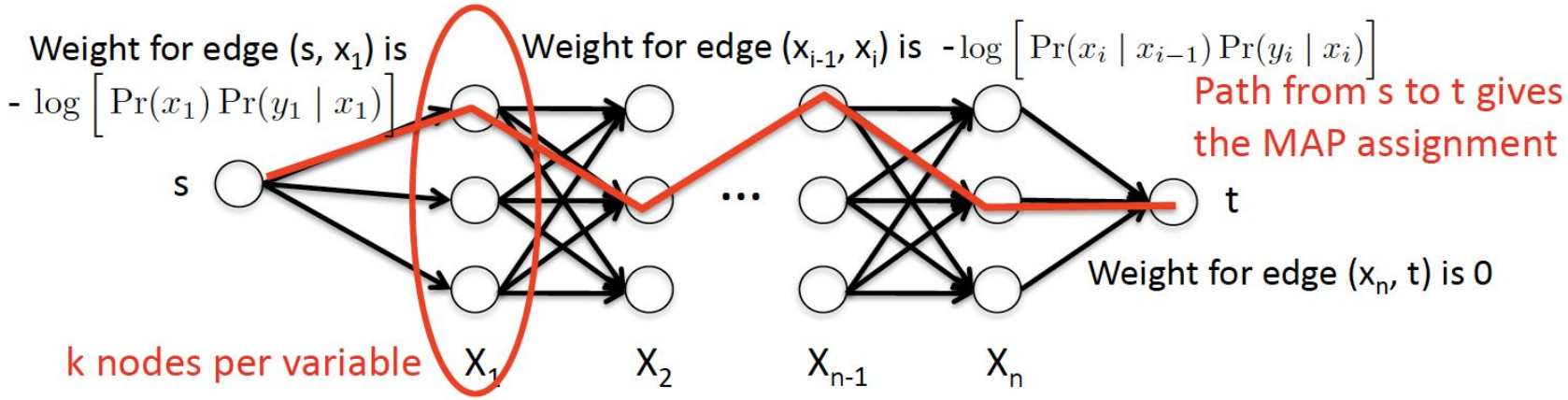


MAP Inference n HMMs

- MAP inference in HMMs can *also* be solved in linear time!

$$\begin{aligned}
&\arg\max_{\mathbf{x}} \Pr(x_1, \ldots x_n \mid y_1, \ldots, y_n) = \arg\max_{\mathbf{x}} \Pr(x_1, \ldots x_n, y_1, \ldots, y_n) \\
&= \arg\max_{\mathbf{x}} \log \Pr(x_1, \ldots x_n, y_1, \ldots, y_n) \\
&= \arg\max_{\mathbf{x}} \log \Big[\Pr(x_1) \Pr(y_1 \mid x_1) \Big] + \sum_{i=2}^{n} \log \Big[\Pr(x_i \mid x_{i-1}) \Pr(y_i \mid x_i) \Big]
\end{aligned}$$

- Formulate as a shortest paths problem



Called the Viterbi algorithm

Given the observations, what is the most probable sequence of hidden states that may have generated the observations?

- POS Tagging

Applications of HMMs

- Speech recognition
  - Predict phonemes from the sounds forming words (i.e., the actual signals)
- Natural language processing
  - Predict parts of speech (verb, noun, determiner, etc.) from the words in a sentence
- Computational biology
  - Predict intron/exon regions from DNA
  - Predict protein structure from DNA (locally)
- And many many more!

Causality?

- When Bayes' nets reflect the true causal patterns:
  - Often simpler (nodes have fewer parents)
  - Often easier to think about and to elicit from experts
- BNs need not actually be causal
  - Sometimes no causal net exists over the domain (especially if variables are missing)
  - End up with arrows that reflect correlation, not causation
- What do the arrows really mean?
  - Topology may happen to encode causal structure
  - Topology really encodes conditional independence

$$P(x_i|x_1, \ldots x_{i-1}) = P(x_i|parents(X_i))$$