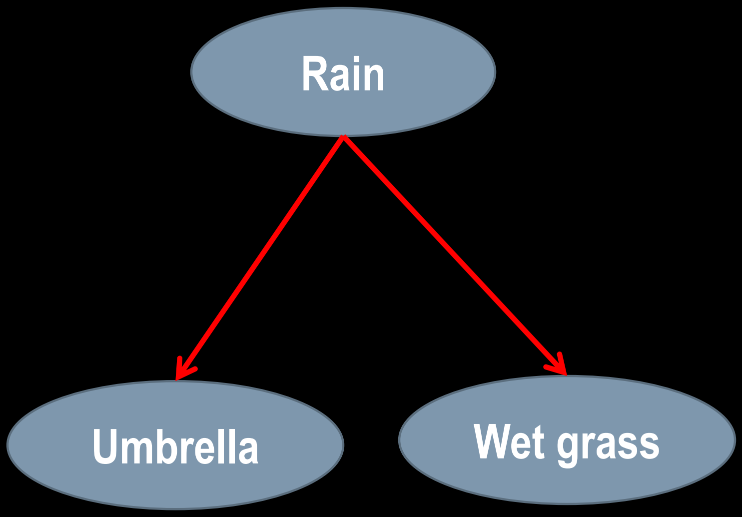
Probability with time and HMMs

Some examples from Sebastian Thrun

Previously…

- We talked about Bayes nets as ways to represent probability distributions



- But what if the events we want to represent change over time?

Outline

- Time and uncertainty
- Inference: filtering, prediction, smoothing
- Hidden Markov Models

Time and uncertainty

The world changes; we need to track and predict it

Basic idea: copy state and evidence variables for each time step

$\mathbf{X}_t$ = set of unobservable state variables at time t
e.g., $BloodSugar_t$, $StomachContents_t$, etc.

$\mathbf{E}_t$ = set of observable evidence variables at time t
e.g., $MeasuredBloodSugar_t$, $PulseRate_t$, $FoodEaten_t$

This assumes **discrete time**; step size depends on problem

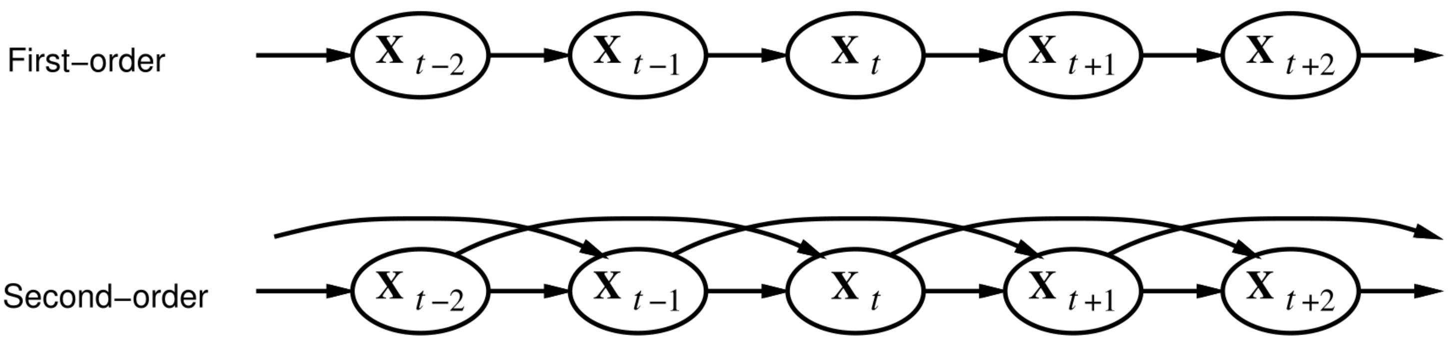
Notation: $\mathbf{X}_{a:b} = \mathbf{X}_a, \mathbf{X}_{a+1}, \ldots, \mathbf{X}_{b-1}, \mathbf{X}_b$

Markov processes (Markov chains)

Construct a Bayes net from these variables: parents?

Markov assumption: $\mathbf{X}_t$ depends on **bounded** subset of $\mathbf{X}_{0:t-1}$

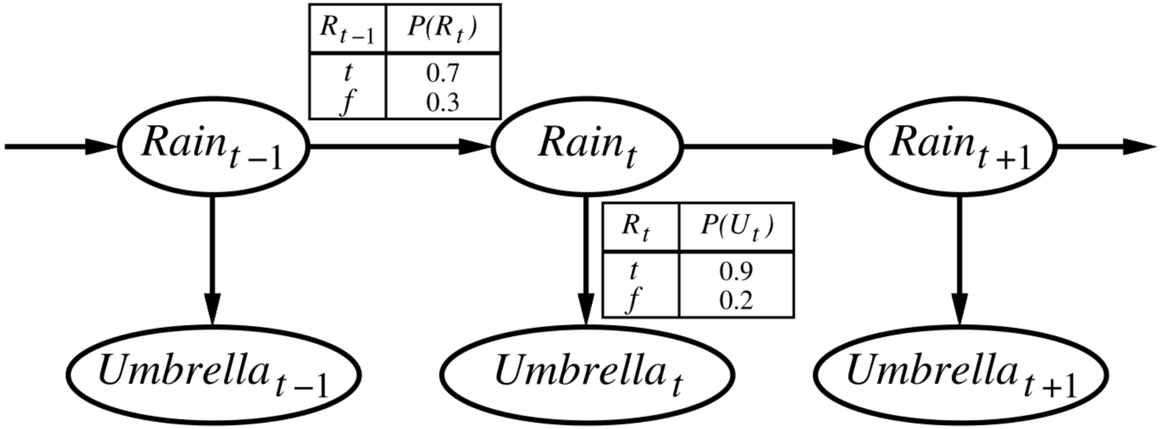
First-order Markov process: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$
Second-order Markov process: $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-2}, \mathbf{X}_{t-1})$



Sensor Markov assumption: $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_{0:t}, \mathbf{E}_{0:t-1}) = \mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$

Stationary process: transition model $\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1})$ and
sensor model $\mathbf{P}(\mathbf{E}_t|\mathbf{X}_t)$ fixed for all t

Example



First-order Markov assumption not exactly true in real world!

Possible fixes:

1. **Increase order** of Markov process
2. **Augment state**, e.g., add $Temp_t$, $Pressure_t$

Example: robot motion.

Augment position and velocity with $Battery_t$

Inference tasks

Filtering: $\mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$
belief state—input to the decision process of a rational agent

Prediction: $\mathbf{P}(\mathbf{X}_{t+k}|\mathbf{e}_{1:t})$ for $k > 0$
evaluation of possible action sequences;
like filtering without the evidence

Smoothing: $\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t})$ for $0 \leq k < t$
better estimate of past states

Most likely explanation: $\arg\max_{\mathbf{x}_{1:t}} P(\mathbf{x}_{1:t}|\mathbf{e}_{1:t})$
speech recognition, decoding with a noisy channel

Filtering

Aim: devise a **recursive** state estimation algorithm:

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = f(\mathbf{e}_{t+1}, \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t}))$$

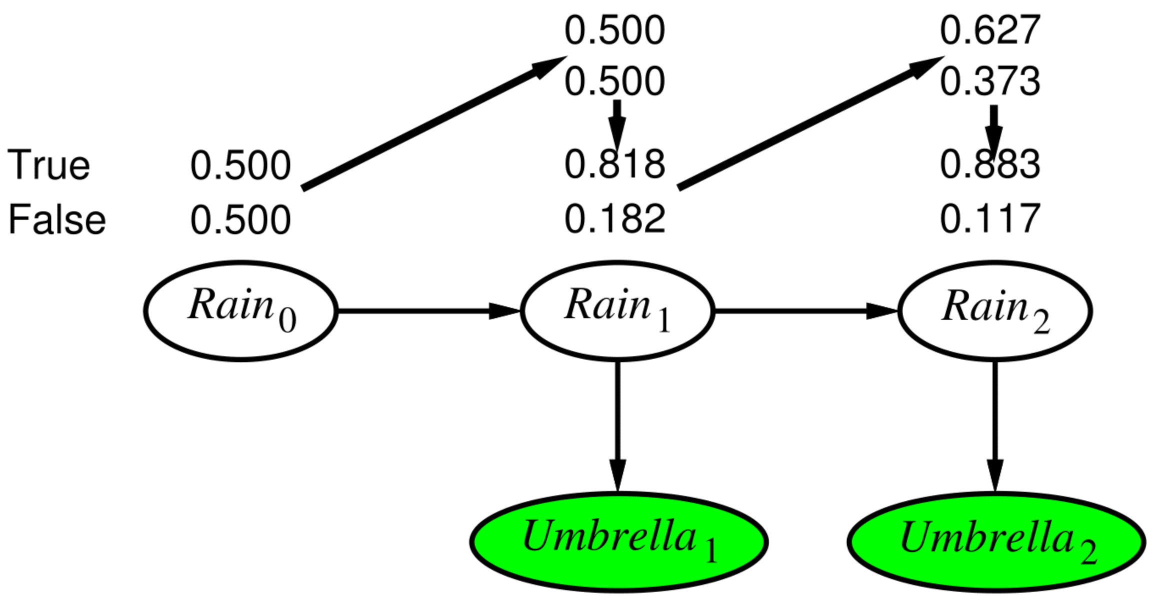
$$\begin{aligned}\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) &= \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}, \mathbf{e}_{t+1}) \\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}, \mathbf{e}_{1:t})\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) \\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})\end{aligned}$$

I.e., prediction + estimation. Prediction by summing out $\mathbf{X}_t$:

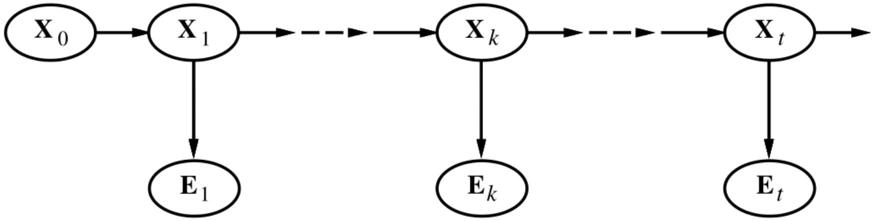
$$\begin{aligned}\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\Sigma_{\mathbf{x}_t}\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t, \mathbf{e}_{1:t})P(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})\Sigma_{\mathbf{x}_t}\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{e}_{1:t})\end{aligned}$$

$\mathbf{f}_{1:t+1} = \text{FORWARD}(\mathbf{f}_{1:t}, \mathbf{e}_{t+1})$ where $\mathbf{f}_{1:t} = \mathbf{P}(\mathbf{X}_t|\mathbf{e}_{1:t})$
Time and space **constant** (independent of t)

Filtering example



Smoothing



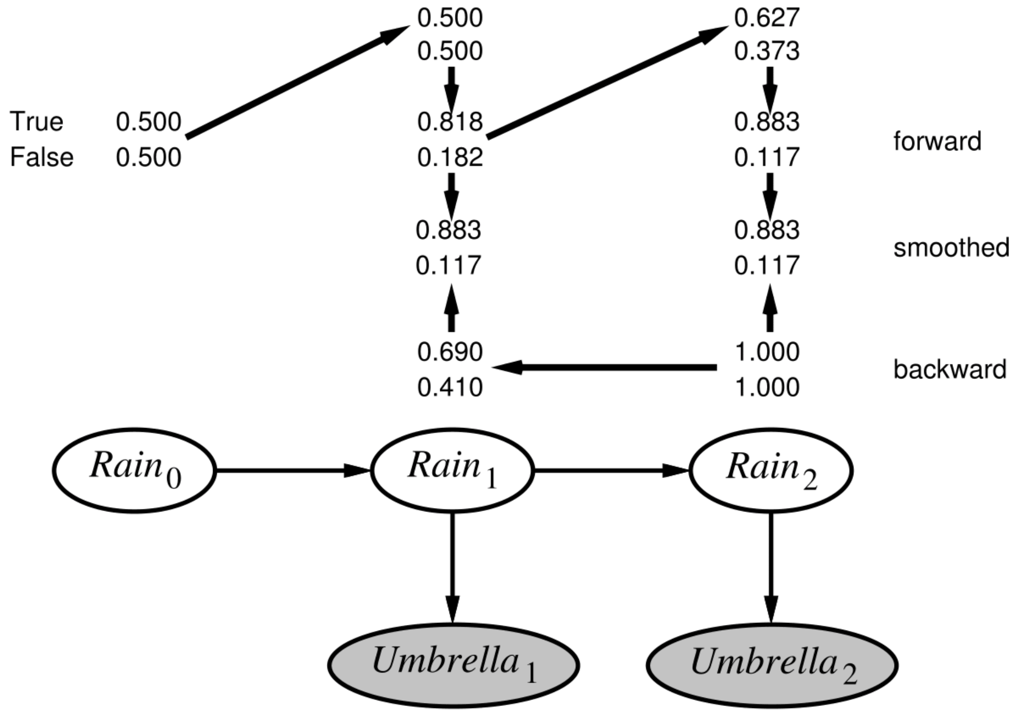
Divide evidence $\mathbf{e}_{1:t}$ into $\mathbf{e}_{1:k}$, $\mathbf{e}_{k+1:t}$:

$$\begin{aligned}
\mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:t}) &= \mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k}, \mathbf{e}_{k+1:t}) \\
&= \alpha \mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k, \mathbf{e}_{1:k}) \\
&= \alpha \mathbf{P}(\mathbf{X}_k|\mathbf{e}_{1:k})\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) \\
&= \alpha \mathbf{f}_{1:k}\mathbf{b}_{k+1:t}
\end{aligned}$$

Backward message computed by a backwards recursion:

$$\begin{aligned}
\mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k) &= \Sigma_{\mathbf{x}_{k+1}} \mathbf{P}(\mathbf{e}_{k+1:t}|\mathbf{X}_k, \mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \\
&= \Sigma_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k) \\
&= \Sigma_{\mathbf{x}_{k+1}} P(\mathbf{e}_{k+1}|\mathbf{x}_{k+1})P(\mathbf{e}_{k+2:t}|\mathbf{x}_{k+1})\mathbf{P}(\mathbf{x}_{k+1}|\mathbf{X}_k)
\end{aligned}$$

Smoothing example



Forward–backward algorithm: cache forward messages along the way
Time linear in t (polytree inference), space $O(t|\mathbf{f}|)$

Most likely explanation

Most likely sequence $\neq$ sequence of most likely states!!!!

Most likely path to each $\mathbf{x}_{t+1}$
= most likely path to **some** $\mathbf{x}_t$ plus one more step

$$\max_{\mathbf{x}_1 \ldots \mathbf{x}_t} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{X}_{t+1} | \mathbf{e}_{1:t+1})$$
$$= \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} P(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t | \mathbf{e}_{1:t}) \right)$$

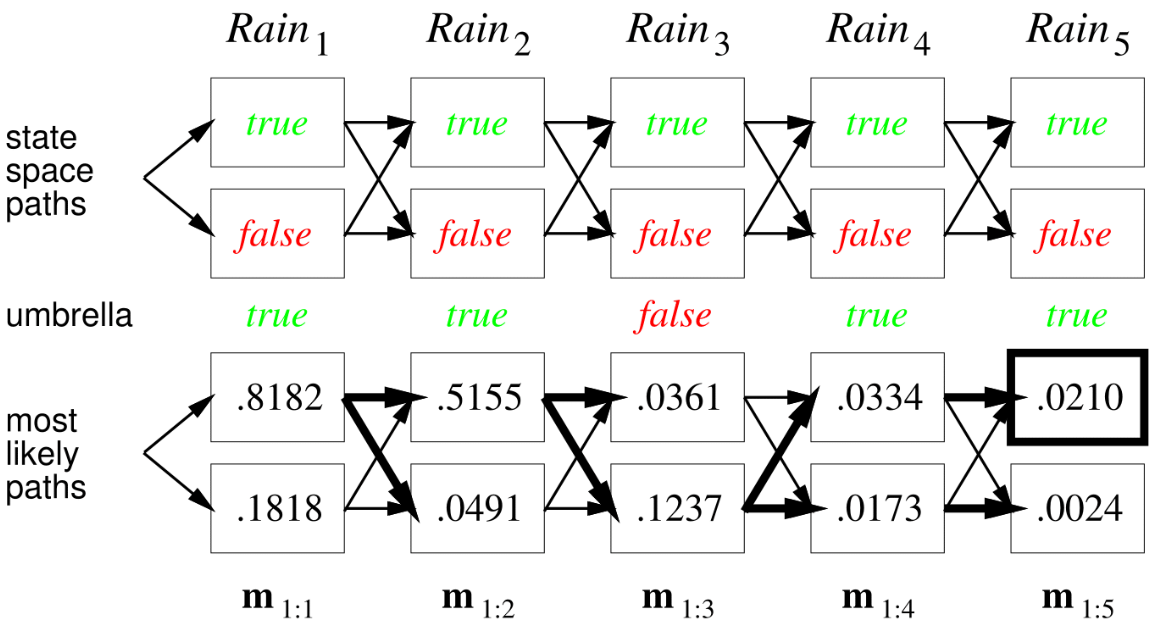
Identical to filtering, except $\mathbf{f}_{1:t}$ replaced by

$$\mathbf{m}_{1:t} = \max_{\mathbf{x}_1 \ldots \mathbf{x}_{t-1}} \mathbf{P}(\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}, \mathbf{X}_t | \mathbf{e}_{1:t}),$$

I.e., $\mathbf{m}_{1:t}(i)$ gives the probability of the most likely path to state i.
Update has sum replaced by max, giving the Viterbi algorithm:

$$\mathbf{m}_{1:t+1} = \mathbf{P}(\mathbf{e}_{t+1} | \mathbf{X}_{t+1}) \max_{\mathbf{x}_t} \left(\mathbf{P}(\mathbf{X}_{t+1} | \mathbf{x}_t) \mathbf{m}_{1:t} \right)$$

Viterbi example



Hidden Markov Models HMMs

Hidden Markov Models (HMMs)

- An HMM is a temporal probabilistic model in which the state is described by a **single discrete random variable**.

$\mathbf{X}_t$ is a single, discrete variable (usually $\mathbf{E}_t$ is too)
Domain of X_t is $\{1, \ldots, S\}$

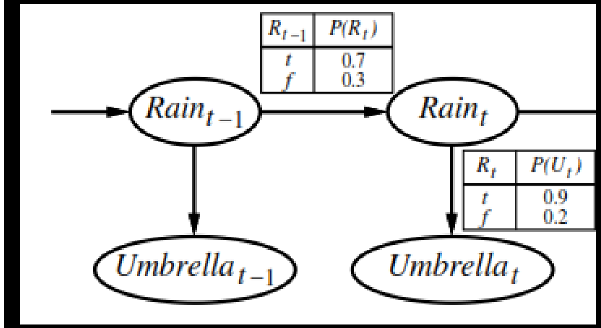
Transition matrix $\mathbf{T}_{ij} = P(X_t = j | X_{t-1} = i)$, e.g., $\begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$

Sensor matrix $\mathbf{O}_t$ for each time step, diagonal elements $P(e_t | X_t = i)$
e.g., with $U_1 = true$, $\mathbf{O}_1 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix}$

Forward and backward messages as column vectors:

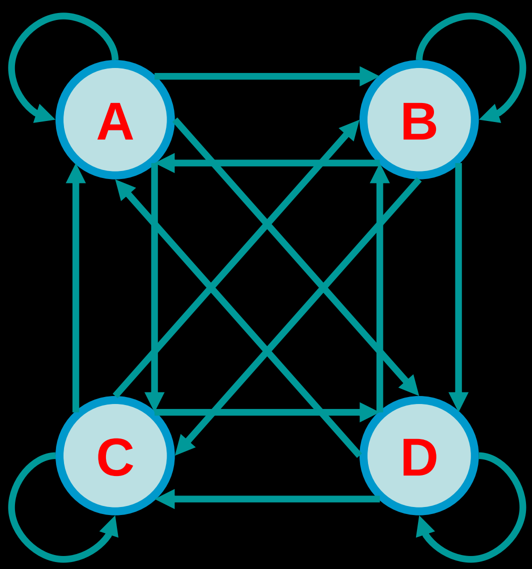
$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{b}_{k+1:t} &= \mathbf{T} \mathbf{O}_{k+1} \mathbf{b}_{k+2:t} \end{aligned}$$

Forward-backward algorithm needs time $O(S^2 t)$ and space $O(St)$



HMMs

- Since we only have one variable (the system state), HMMs are often depicted like state machines:



Don't confuse this with a Bayes net diagram!

HMM with four states: A, B, C, D

- In robotics: HMMs useful for anomaly detection, localization, etc.

Example: The Dishonest Casino

A casino has two dice:

- Fair die

  P(1) = P(2) = P(3) = P(5) = P(6) = 1/6

- Loaded die

  P(1) = P(2) = P(3) = P(5) = 1/10

  P(6) = 1/2

Casino player switches back-&-forth between fair and loaded die once every 20 turns

Game:

1. You bet $1
2. You roll (always with a fair die)
3. Casino player rolls (maybe with fair die, maybe with loaded die)
4. Highest number wins $2

Question # 1 – Evaluation

GIVEN

A sequence of rolls by the casino player:

1245526462146146136136661664661636616366163616515615115146123562344

Prob = 1.3×10^{-35}

QUESTION

How likely is this sequence, given our model of how the casino works?

This is the **EVALUATION** problem in HMMs

Question # 2 – Decoding

GIVEN

A sequence of rolls by the casino player

124552646214614613613	6661664661636616366163616	515615115146123562344
FAIR	LOADED	FAIR

QUESTION

What portion of the sequence was generated with the fair die, and what portion with the loaded die?

This is the **DECODING** question in HMMs

Question # 3 – Learning

GIVEN

A sequence of rolls by the casino player

124552646214614613613666166466163661636616361651561511514612356234

Prob(6) = 64%

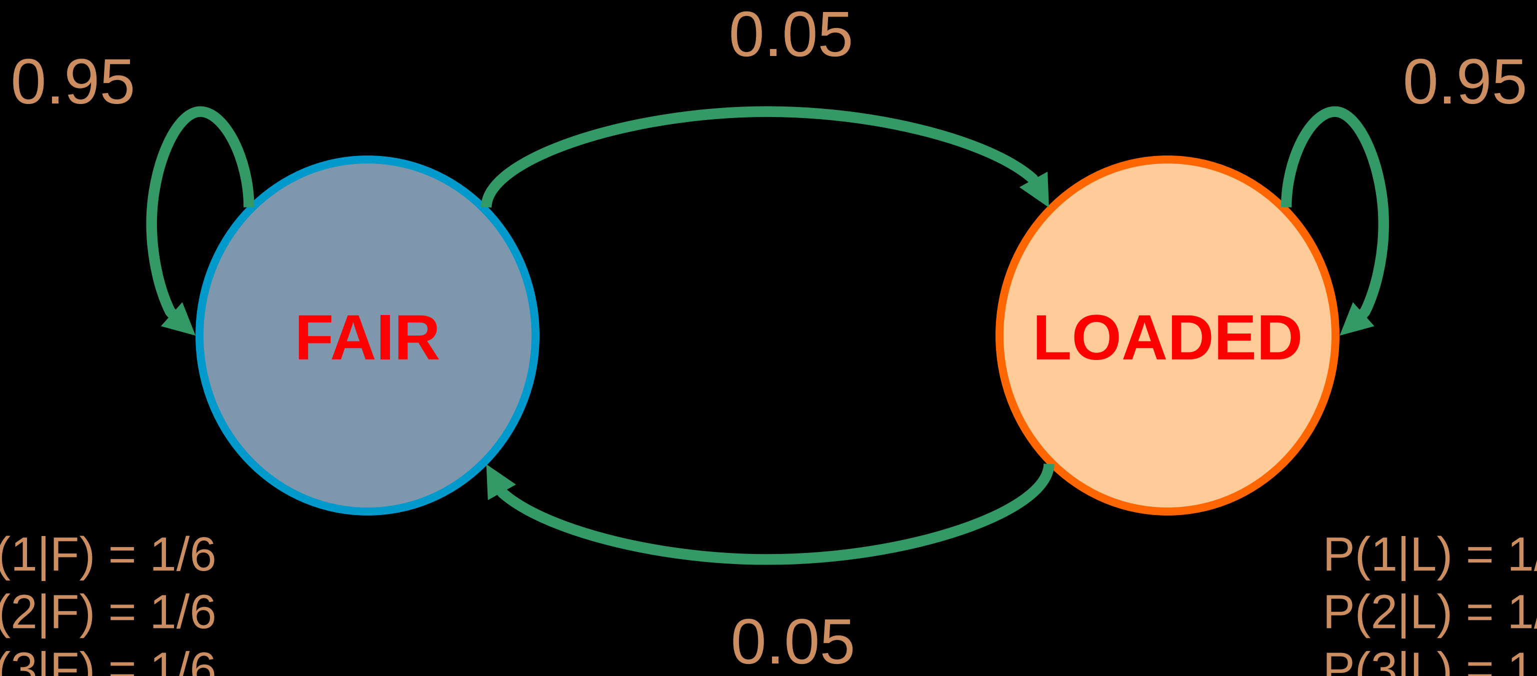
QUESTION

How “loaded” is the loaded die? How “fair” is the fair die? How often does the casino player change from fair to loaded, and back?

This is the **LEARNING** question in HMMs

- We won’t cover this, but if you’re interested look-up the Baum–Welch Algorithm

The dishonest casino HMM model

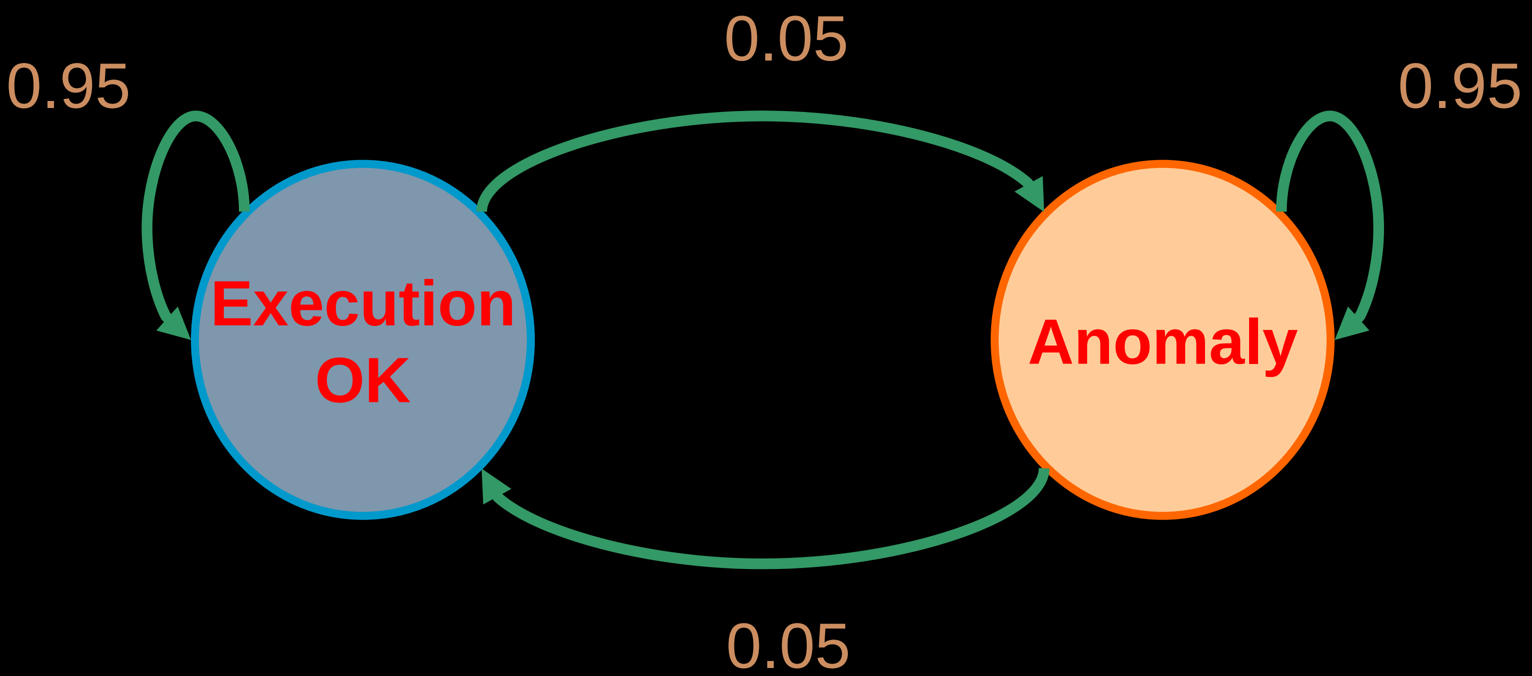


P(1|F) = 1/6
P(2|F) = 1/6
P(3|F) = 1/6
P(4|F) = 1/6
P(5|F) = 1/6
P(6|F) = 1/6

P(1|L) = 1/10
P(2|L) = 1/10
P(3|L) = 1/10
P(4|L) = 1/10
P(5|L) = 1/10
P(6|L) = 1/2

Not just for casino games!

- We need methods like this for anomaly detection in robotics!



- We'll see an example later

More precise definition of a hidden Markov model

Definition: A hidden Markov model (HMM)

- Alphabet $\Sigma = \{ b_1, b_2, \ldots, b_M \}$
- Set of states $Q = \{ 1, \ldots, K \}$
- Transition probabilities between any two states

  a_{ij} = transition prob from state i to state j

  $a_{i1} + \ldots + a_{iK} = 1$, for all states i = 1…K

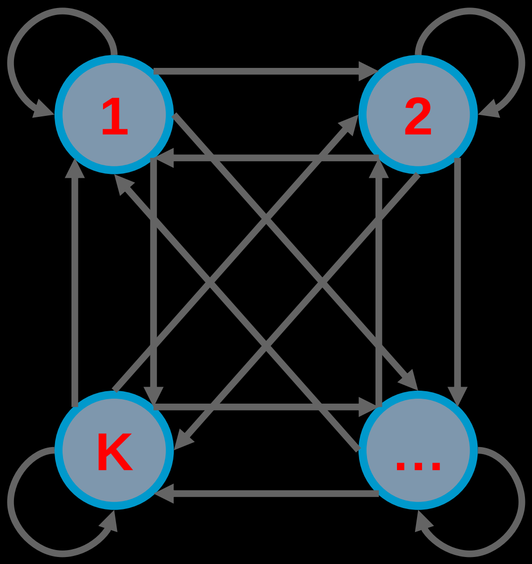
- Start probabilities a_{0i}

  $a_{01} + \ldots + a_{0K} = 1$

- Emission probabilities within each state

  $e_i(b) = P(E_i = b \mid X_i = k)$

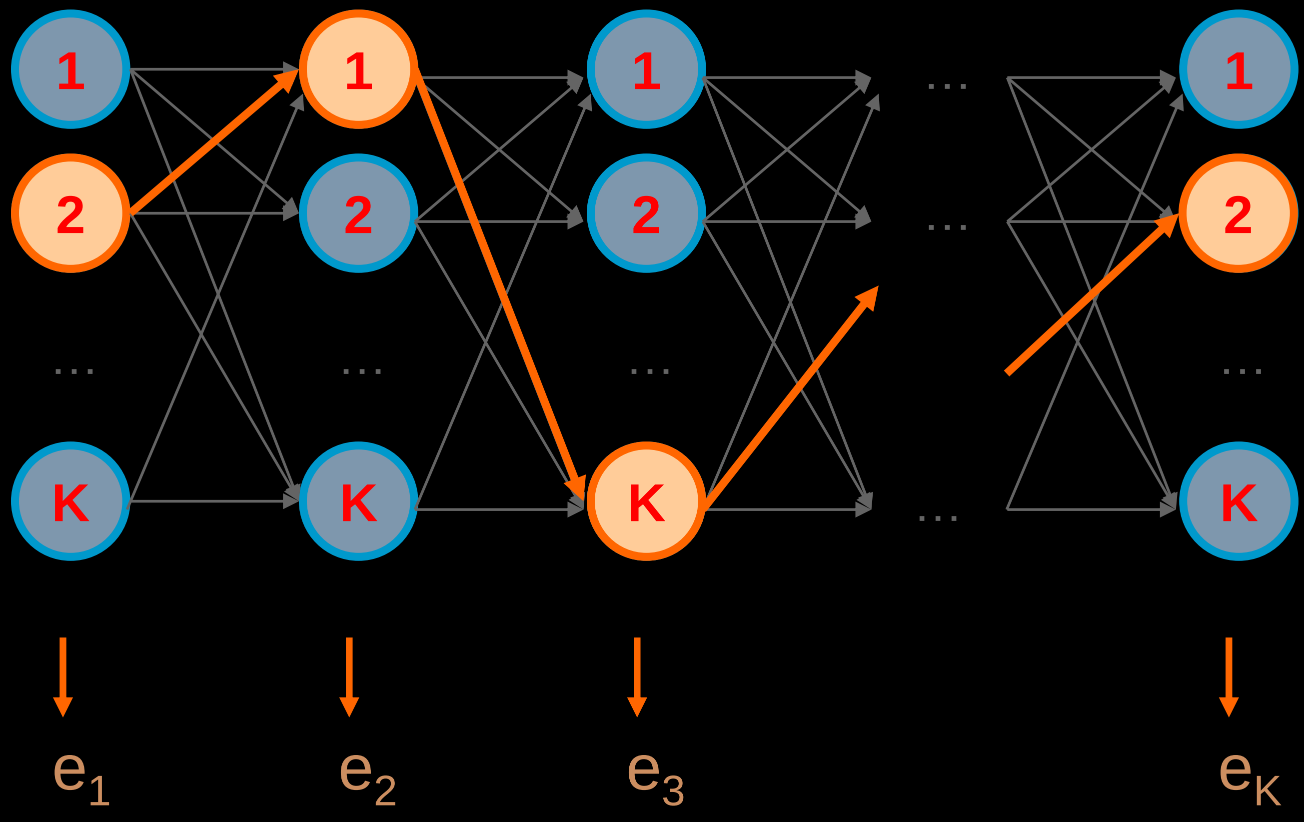
  $e_i(b_1) + \ldots + e_i(b_M) = 1$, for all states i = 1…K



A parse of a sequence

Given a sequence $\mathbf{e} = e_1......e_N$,

A parse of $\mathbf{e}$ is a sequence of states $\mathbf{X} = x_1, ..., x_N$



Evaluation: Likelihood of a parse

- Given a sequence $\mathbf{e} = e_1 \ldots\ldots e_N$ and a parse $\mathbf{x} = x_1, \ldots\ldots, x_N$,

- To find how likely this scenario is (given our HMM):

$$P(\mathbf{e}, \mathbf{x}) = P(e_1, \ldots, e_N, x_1, \ldots\ldots, x_N) =$$

$$P(e_N \mid x_N)\, P(x_N \mid x_{N-1}) \ldots\ldots P(e_2 \mid x_2)\, P(x_2 \mid x_1)\, P(e_1 \mid x_1)\, P(x_1) =$$

$$a_{0x1}\, a_{x1x2} \ldots\ldots a_{xN-1xN}\, e_{x1}(e_1) \ldots\ldots e_{xN}(e_N)$$

Evaluation problem for the dishonest casino

- Let the sequence of rolls be:

  e = 1, 2, 1, 5, 6, 2, 1, 5, 2, 4

- Then, what is the likelihood of

  x = Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair, Fair?

  (say initial probs $a_{0Fair} = ½$, $a_{0Loaded} = ½$)

  $½ \times P(1 \mid Fair)\, P(Fair \mid Fair)\, P(2 \mid Fair)\, P(Fair \mid Fair) \ldots P(4 \mid Fair) =$

  $½ \times (1/6)^{10} \times (0.95)^{9} = .00000000521158647211 \sim= 0.5 \times 10^{-9}$

Evaluation problem for the dishonest casino

So, the likelihood the die is fair in this run is just 0.521×10^{-9}

What is the likelihood of

x = Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded, Loaded?

$\frac{1}{2} \times P(1 \mid \text{Loaded})\, P(\text{Loaded}, \text{Loaded}) \ldots P(4 \mid \text{Loaded}) =$

$\frac{1}{2} \times (1/10)^9 \times (1/2)^1 (0.95)^9 = .00000000015756235243 \sim= 0.16 \times 10^{-9}$

Therefore, it is more likely that all the rolls are done with the fair die, than that they are all done with the loaded die

Evaluation problem for the dishonest casino

- Let the sequence of rolls be:

  e = 1, 6, 6, 5, 6, 2, 6, 6, 3, 6

- What is the likelihood **x** = F, F, …, F?

  $\frac{1}{2} \times (1/6)^{10} \times (0.95)^{9} \sim= 0.5 \times 10^{-9}$, same as before

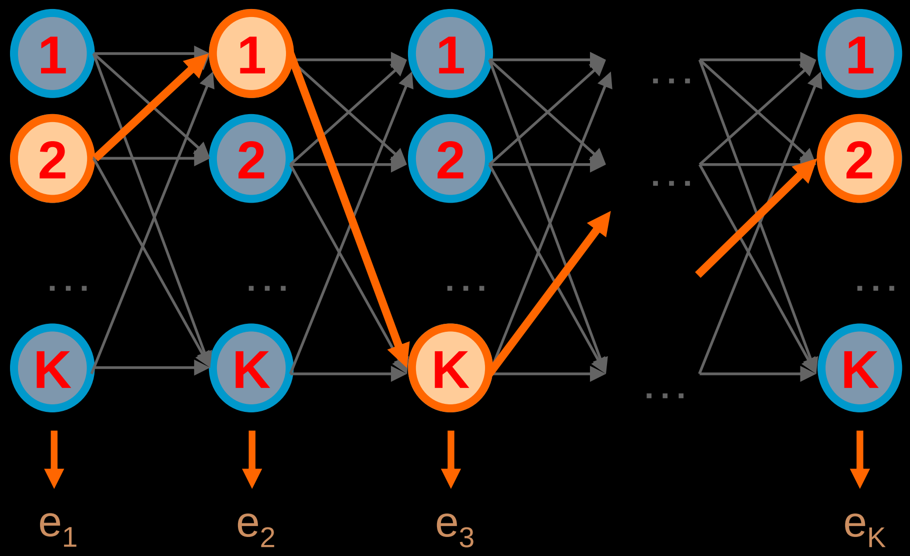
- What is the likelihood **x** = L, L, …, L?

  $\frac{1}{2} \times (1/10)^{4} \times (1/2)^{6} (0.95)^{9} = .00000049238235134735 \sim= 0.5 \times 10^{-7}$

So, it is 100 times more likely the die is loaded

Decoding Problem

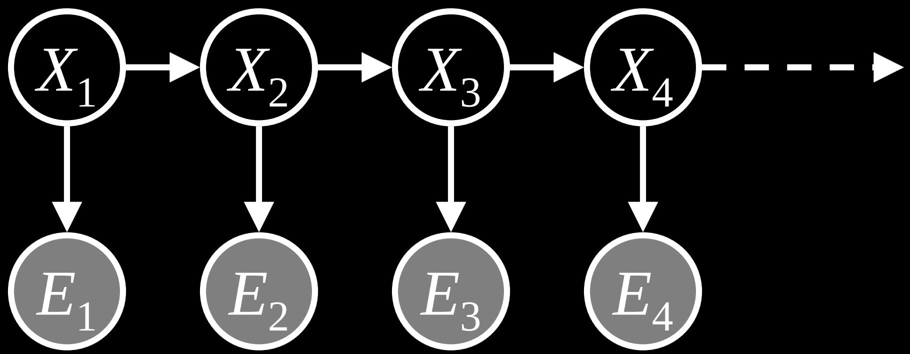
- GIVEN $\mathbf{e} = e_{1,}e_2\ldots\ldots e_N$
- Find $\mathbf{x} = x_1, \ldots\ldots, x_N$, that maximizes $P(\mathbf{e}, \mathbf{x})$
- $x^* = \text{argmax}_x\, P(\mathbf{e}, \mathbf{x})$
- This is just like finding the max-likelihood sequence of a Bayes net
- Use the Viterbi algorithm



BREAK

HMMs for Robot Localization

- What we need to know



(1) State domains: e.g. (x,y) position

(2) Evidence domains: Sensor model

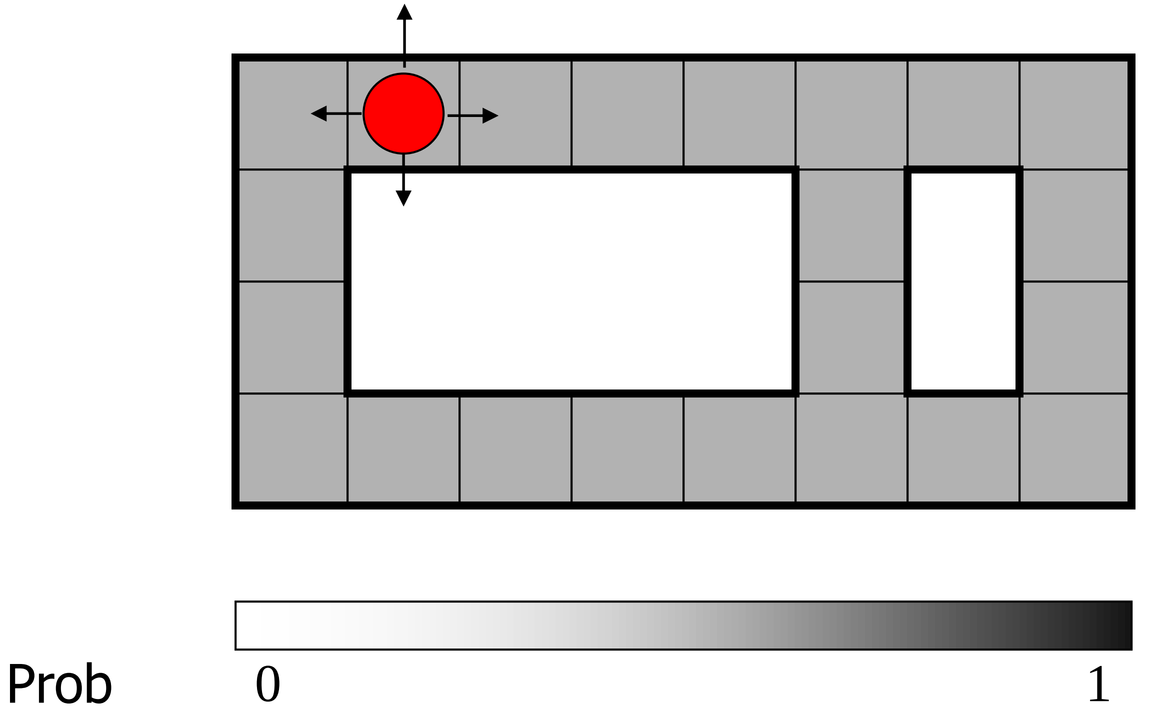
(3) Probability of states at time 0: Usually known or “kidnapped robot”

(4) Transition probability: Defined by actions

(5) Emission probability: Defined by sensor model

Localization Example

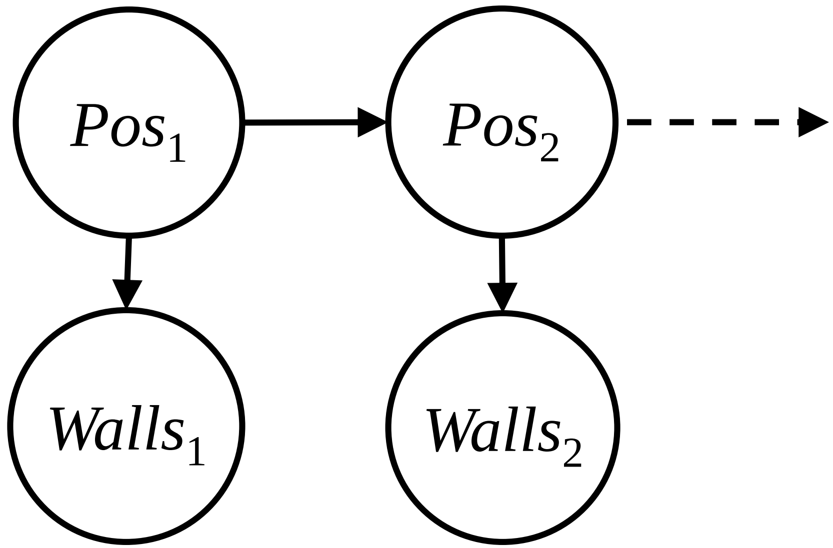
Example from Michael Pfeiffer

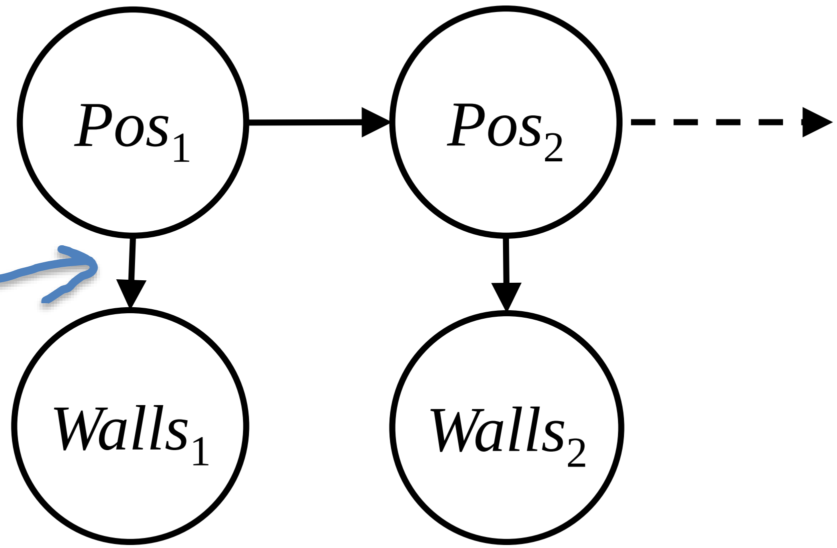


t=0

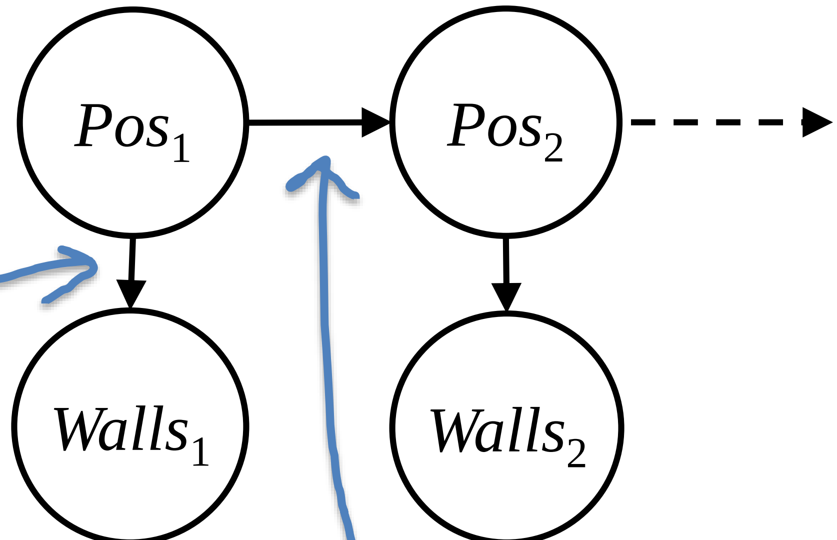
Sensor model: never more than 1 mistake

Motion model: may not execute action with small prob.


Pos_1
Pos_2
$Walls_1$
$Walls_2$



Sometimes sensors are wrong

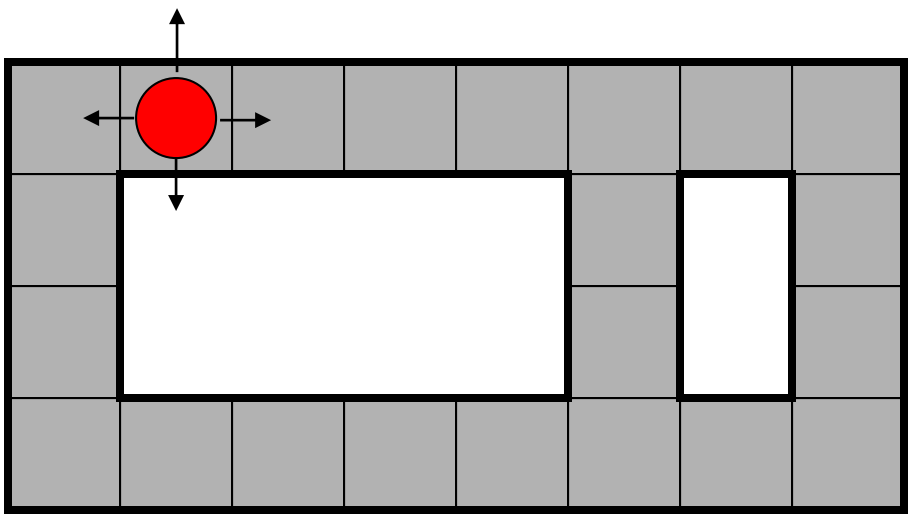


Sometimes sensors are wrong

Sometimes we have actuation error

Wall-sensing Localization Example

Example from Michael Pfeiffer

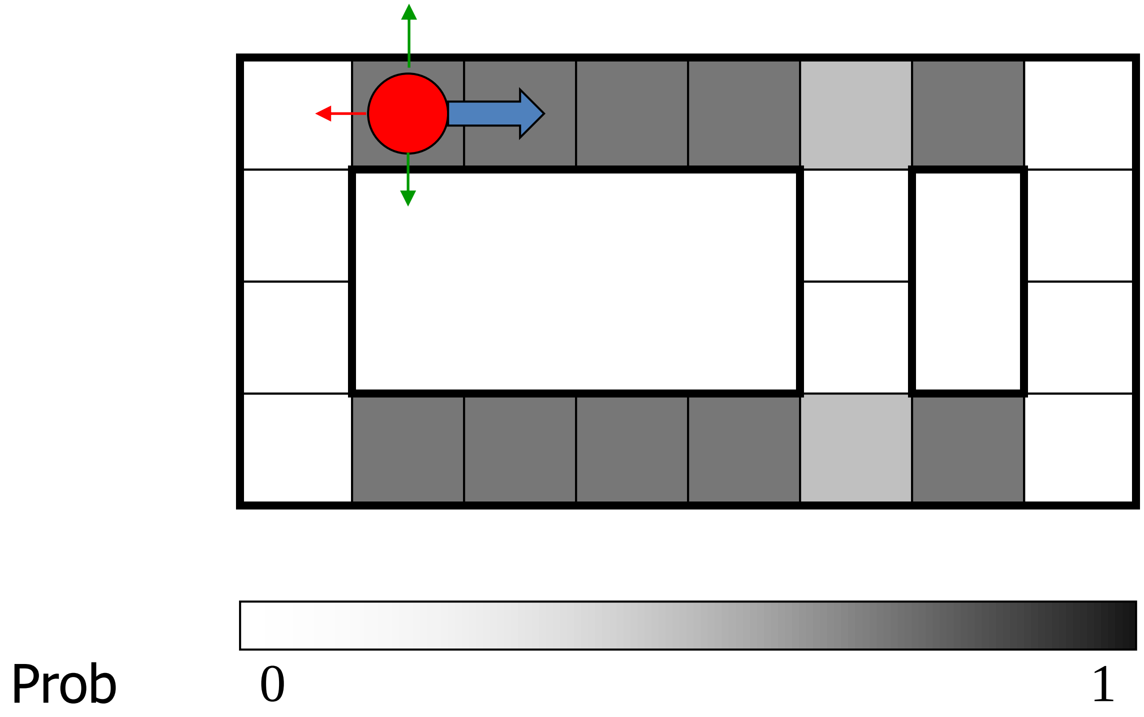


t=0

Sensor model: never more than 1 mistake

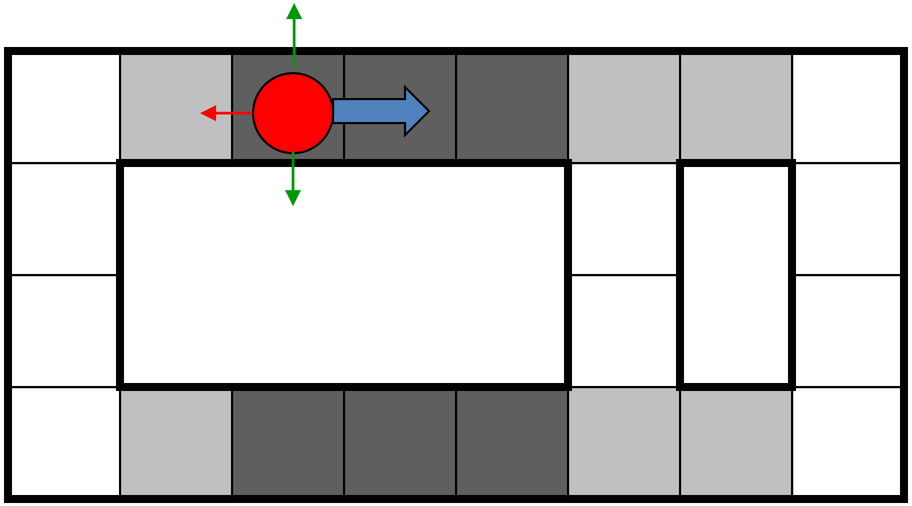
Motion model: may not execute action with small prob.

Wall-sensing Localization Example



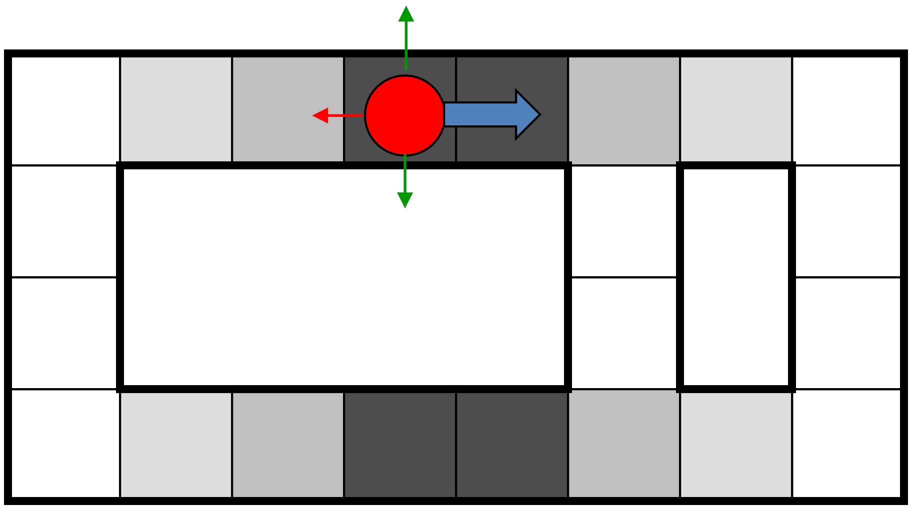
t=1

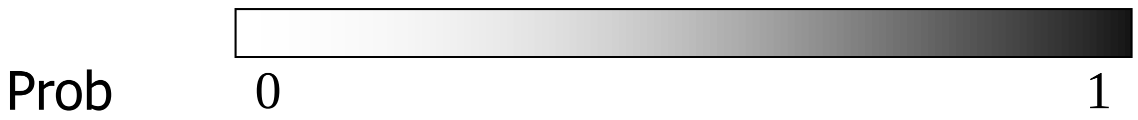
Wall-sensing Localization Example



t=2

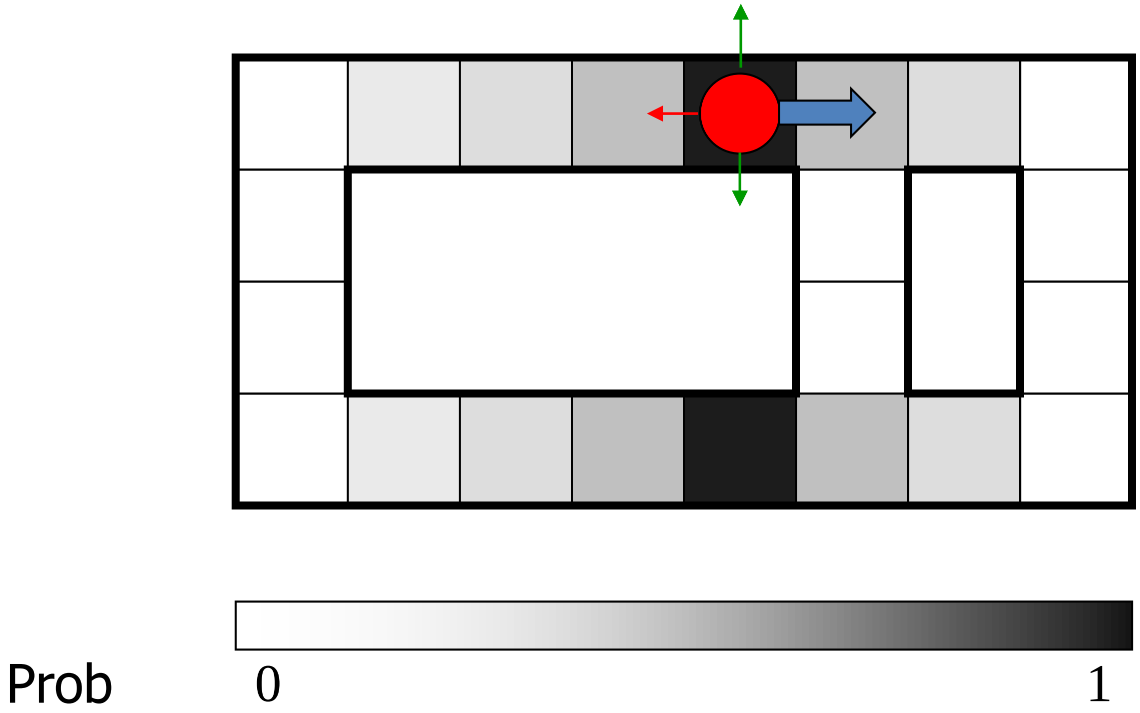
Wall-sensing Localization Example





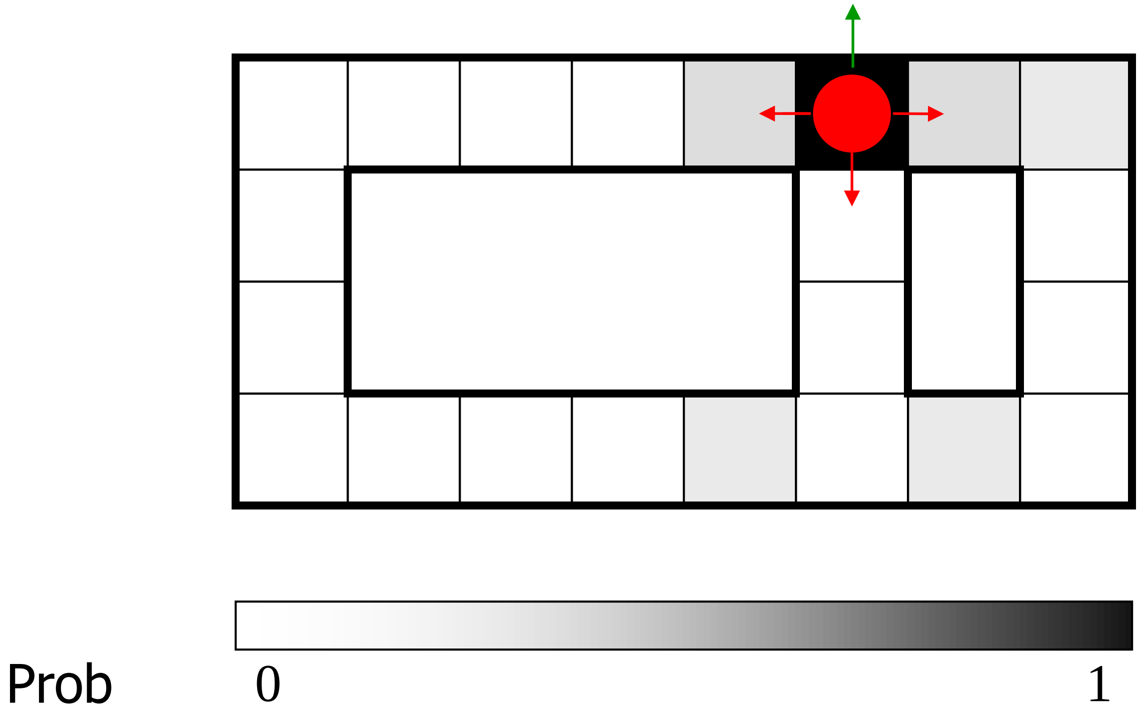
t=3

Wall-sensing Localization Example



t=4

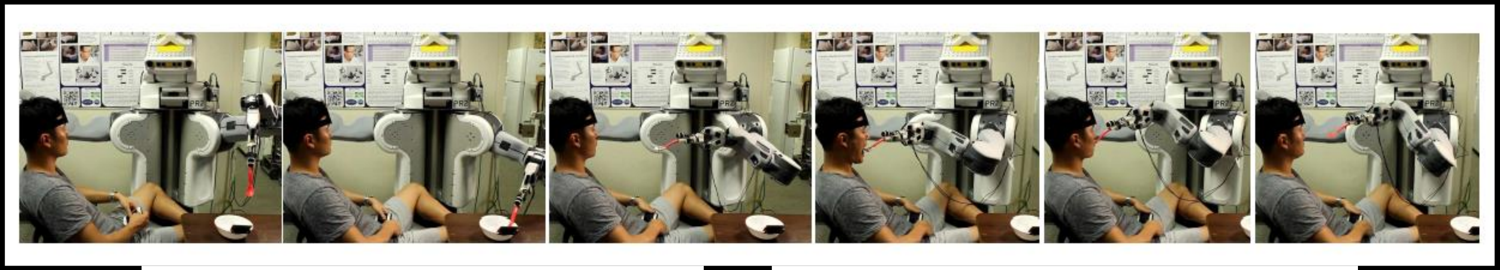
Wall-sensing Localization Example

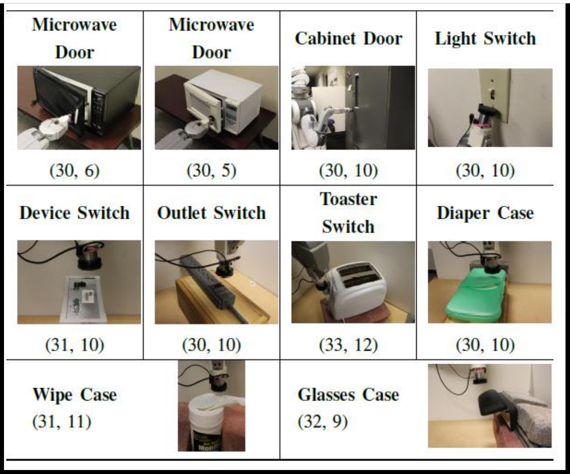


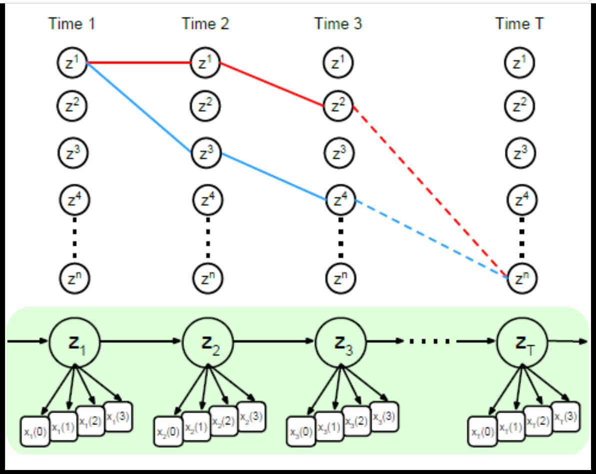
t=5

HMM Evaluation Problem Application: Anomaly Detection for Robot Manipulation

- Monitor anomalies in manipulation task execution with an HMM
  - Uses force, visual, auditory, and kinematic sensing







- Using General Hidden Markov Model library (http://www.ghmm.org)

"Multimodal Execution Monitoring for Anomaly Detection During Robot Manipulation," D. Park, Z. Erickson, T. Bhattacharjee, and C. Kemp, ICRA 2016

Summary

- Can construct Bayes nets that account for time-varying uncertainty
- These are useful for filtering, prediction, smoothing and most likely path questions
- HMMs are Bayes nets that make Markov assumption and have a single discrete random variable
  - Evaluation: How likely is this sequence of evidence given a model
  - Decoding: Find sequence of states that best explains the evidence
- HMMs have been used for localization and anomaly detection in robotics

Homework

- AI book Ch. 15.4-15.5
- Kalman Filter Derivation here
- HW 4 is due Wednesday!