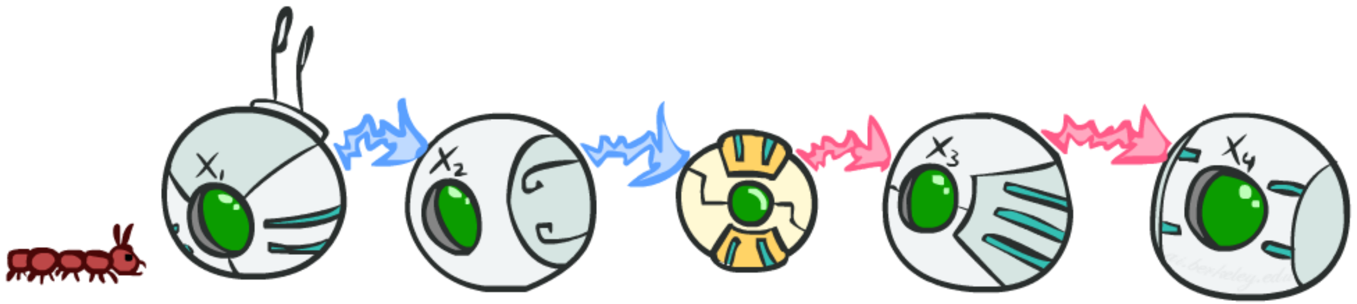
Artificial Intelligence

Markov Models



[These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu.]

Uncertainty and Time

- Often, we want to reason about a **sequence** of observations where the state of the underlying system is **changing**
  - Speech recognition
  - Robot localization
  - User attention
  - Medical monitoring
  - Global climate
- Need to introduce time into our models

Markov Models (aka Markov chain/process)

- Value of X at a given time is called the state

$$X_0 \rightarrow X_1 \rightarrow X_2 \rightarrow X_3 \dashrightarrow$$

$P(X_0)$ $\qquad\qquad$ $P(X_t \mid X_{t-1})$

- The ***transition model*** $P(Xt \mid Xt\text{-}1)$ specifies how the state evolves over time
- ***Stationarity*** assumption: transition probabilities are the same at all times
- ***Markov*** assumption: “future is independent of the past given the present”
  - Xt+1 is independent of $X_0, \ldots, X_{t-1}$ given X_t
  - This is a ***first-order*** Markov model (a *k*th-order model allows dependencies on *k* earlier steps)
- Joint distribution $P(X_0, \ldots, X_T) = P(X_0) \prod_t P(X_t \mid X_{t-1})$

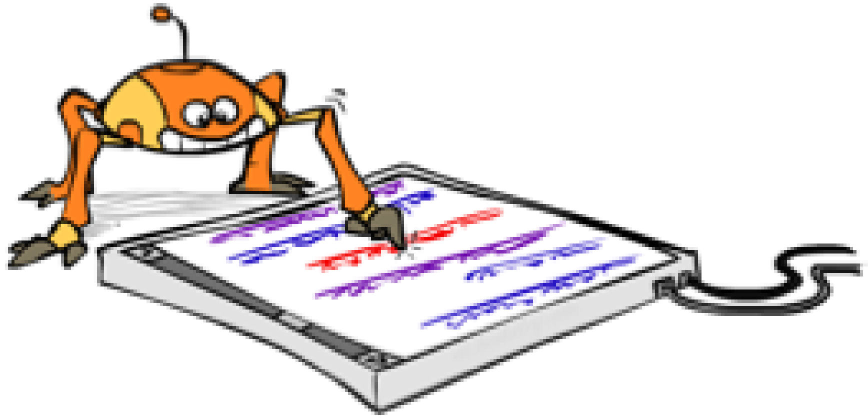
Example: n-gram models

We call ourselves *Homo sapiens*—man the wise—because our **intelligence** is so important to us. For thousands of years, we have tried to understand *how we think*; that is, how a mere handful of matter can perceive, understand, predict, and manipulate a world far larger and more complicated than itself. ….

- State: word at position *t* in text (can also build letter n-grams)
- Transition model (probabilities come from empirical frequencies):
  - Unigram (zero-order): $P(Word_t = i)$
    - "logical are as are confusion a may right tries agent goal the was . . ."
  - Bigram (first-order): $P(Word_t = i \mid Word_{t-1} = j)$
    - "systems are very similar computational approach would be represented . . ."
  - Trigram (second-order): $P(Word_t = i \mid Word_{t-1} = j, Word_{t-2} = k)$
    - "planning and scheduling are integrated the success of naive bayes model is . . ."
- Applications: text classification, spam detection, author identification, language classification, speech recognition

Example: Web browsing

- State: URL visited at step *t*
- Transition model:
  - With probability *p*, choose an outgoing link at random
  - With probability (1-*p*), choose an arbitrary new page
- Question: What is the ***stationary distribution*** over pages?
  - I.e., if the process runs forever, what fraction of time does it spend in any given page?
- Application: Google page rank



Joint Distribution of a Markov Model

$$X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4$$

$$P(X_1) \qquad P(X_t|X_{t-1})$$

- Joint distribution:

$$P(X_1, X_2, X_3, X_4) = P(X_1)P(X_2|X_1)P(X_3|X_2)P(X_4|X_3)$$

- More generally:

$$P(X_1, X_2, \ldots, X_T) = P(X_1)P(X_2|X_1)P(X_3|X_2)\ldots P(X_T|X_{T-1})$$

$$= P(X_1)\prod_{t=2}^{T} P(X_t|X_{t-1})$$

- Questions to be resolved:
  - Does this indeed define a joint distribution?
  - Can every joint distribution be factored this way, or are we making some assumptions about the joint distribution by using this factorization?

Example Markov Chain: Weather

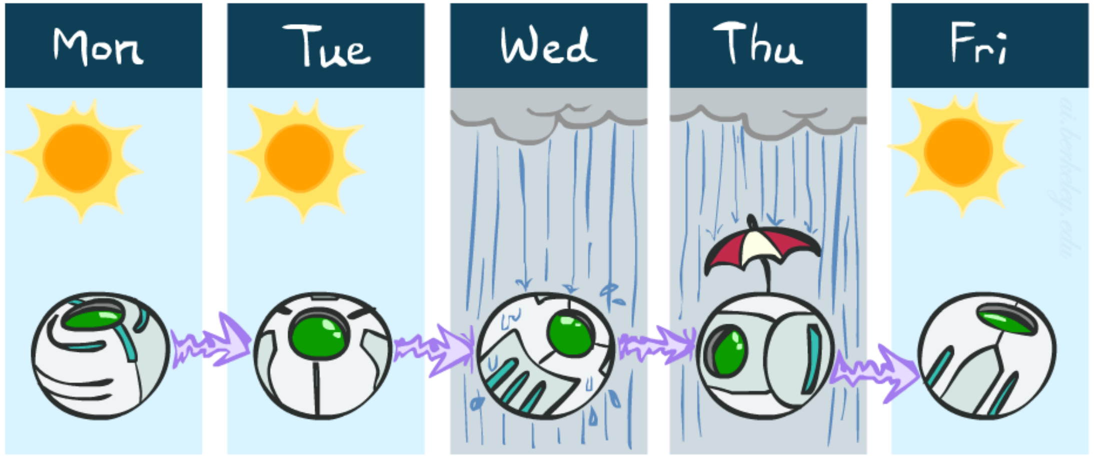
- States {rain, sun}

- Initial distribution $P(X_0)$

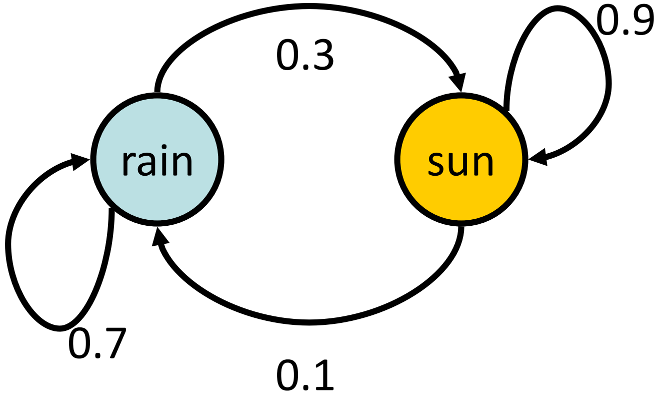
$P(X_0)$	
sun	rain
0.5	0.5

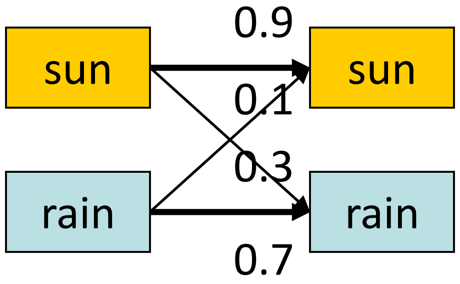
- Transition model $P(X_t \mid X_{t-1})$

X_{t-1}	$P(X_t \mid X_{t-1})$	
	sun	rain
sun	0.9	0.1
rain	0.3	0.7



Two new ways of representing the same CPT

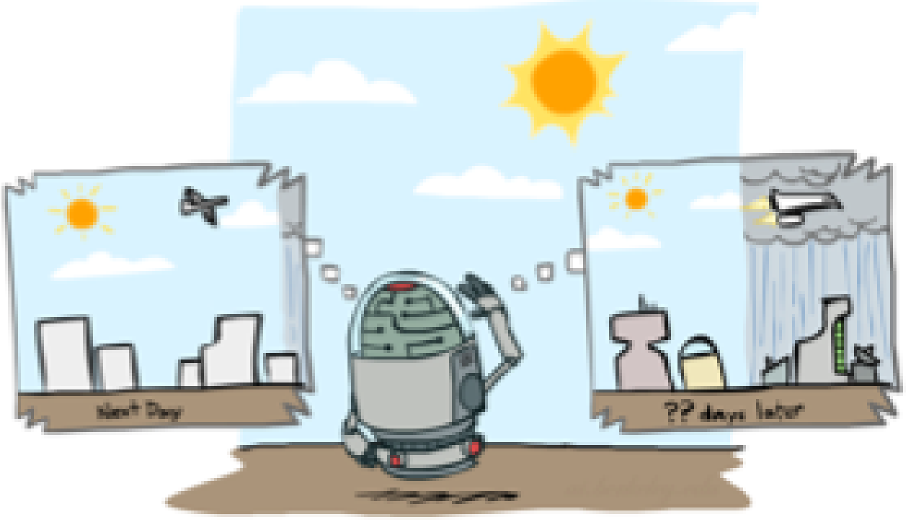




Weather prediction

- Time 0: <0.5,0.5>

X_{t-1}	$P(X_t \mid X_{t-1})$	
	sun	rain
sun	0.9	0.1
rain	0.3	0.7

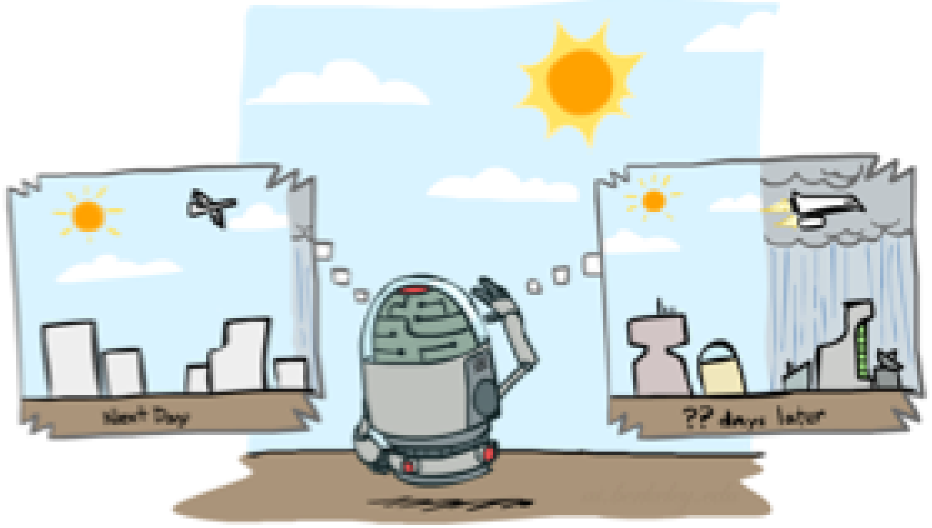


- What is the weather like at time 1?
  - $P(X_1) = \sum_{x_0} P(X_1, X_0 = x_0)$
  - $\quad = \sum_{x_0} P(X_0 = x_0)\, P(X_1 \mid X_0 = x_0)$
  - $\quad = 0.5\langle 0.9, 0.1\rangle + 0.5\langle 0.3, 0.7\rangle = \langle 0.6, 0.4\rangle$

Weather prediction, contd.

- Time 1: <0.6,0.4>

X_{t-1}	$P(X_t \mid X_{t-1})$	
	sun	rain
sun	0.9	0.1
rain	0.3	0.7

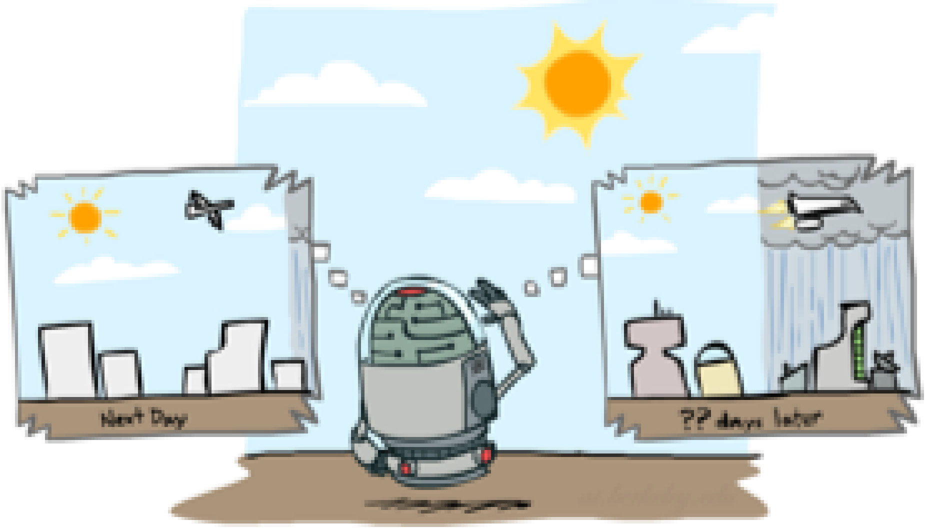


- What is the weather like at time 2?
  - $P(X_2) = \sum_{x_1} P(X_2, X_1{=}x_1)$
  - $\quad = \sum_{x_1} P(X_1{=}x_1)\, P(X_2 \mid X_1{=}x_1)$
  - $\quad = 0.6\langle 0.9, 0.1\rangle + 0.4\langle 0.3, 0.7\rangle = \langle 0.66, 0.34\rangle$

Weather prediction, contd.

- Time 2: <0.66,0.34>

X_{t-1}	$P(X_t \mid X_{t-1})$	
	sun	rain
sun	0.9	0.1
rain	0.3	0.7

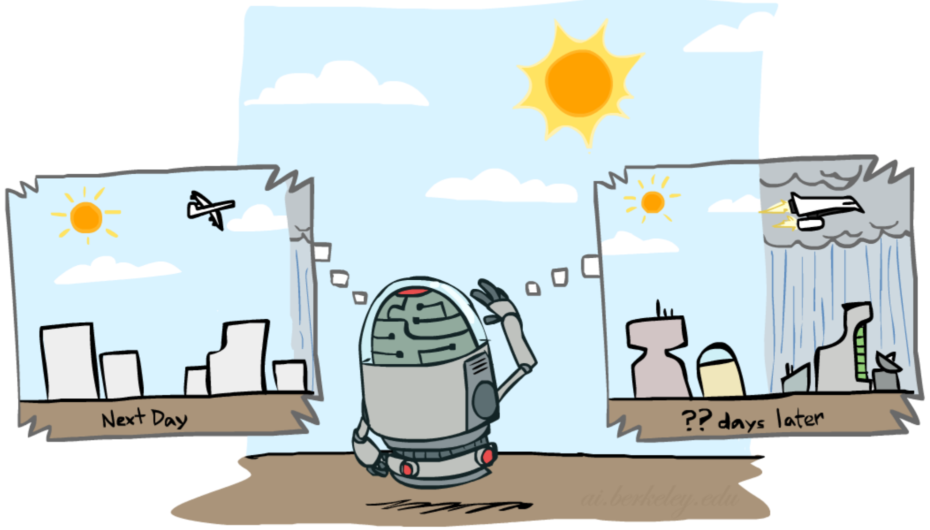


- What is the weather like at time 3?
  - $P(X_3) = \sum_{x_2} P(X_3, X_2{=}x_2)$
  - $ = \sum_{x_2} P(X_2{=}x_2)\, P(X_3 \mid X_2{=}x_2)$
  - $ = 0.66\langle 0.9, 0.1\rangle + 0.34\langle 0.3, 0.7\rangle = \langle 0.696, 0.304\rangle$

Mini-Forward Algorithm

- Question: What's P(X) on some day t?

$$X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4 \dashrightarrow$$

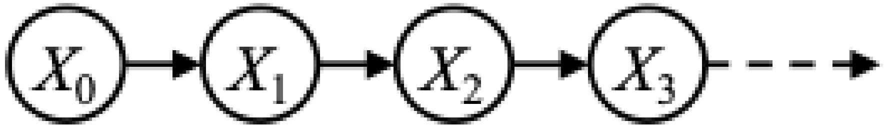


$$\color{red}{P(x_1)} = \text{known}$$

$$P(x_t) = \sum_{x_{t-1}} P(x_{t-1}, x_t)$$
$$= \sum_{x_{t-1}} P(x_t \mid x_{t-1}) P(x_{t-1})$$

Forward simulation

Forward algorithm (simple form)



$P(X_0)$ $P(X_t \mid X_{t-1})$

Probability from previous iteration

Transition model

- What is the state at time t?
  - $P(X_t) = \sum_{x_{t-1}} P(X_t, X_{t-1}=x_{t-1})$
  - $= \sum_{x_{t-1}} P(X_{t-1}=x_{t-1}) \, P(X_t \mid X_{t-1}=x_{t-1})$
- Iterate this update starting at t=0
  - This is called a ***recursive*** update: $P_t = g(P_{t-1}) = g(g(g(g(\ldots P_0))))$

And the same thing in linear algebra

- What is the weather like at time 2?
  - $P(X_2) = 0.6\langle 0.9, 0.1\rangle + 0.4\langle 0.3, 0.7\rangle = \langle 0.66, 0.34\rangle$
- In matrix-vector form:
  - $P(X_2) = \begin{pmatrix} 0.9 & 0.3 \\ 0.1 & 0.7 \end{pmatrix} \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} = \begin{pmatrix} 0.66 \\ 0.34 \end{pmatrix}$

X_{t-1}	$P(X_t \mid X_{t-1})$	
	sun	rain
sun	0.9	0.1
rain	0.3	0.7

- I.e., multiply by T^T, transpose of transition matrix

Stationary Distributions

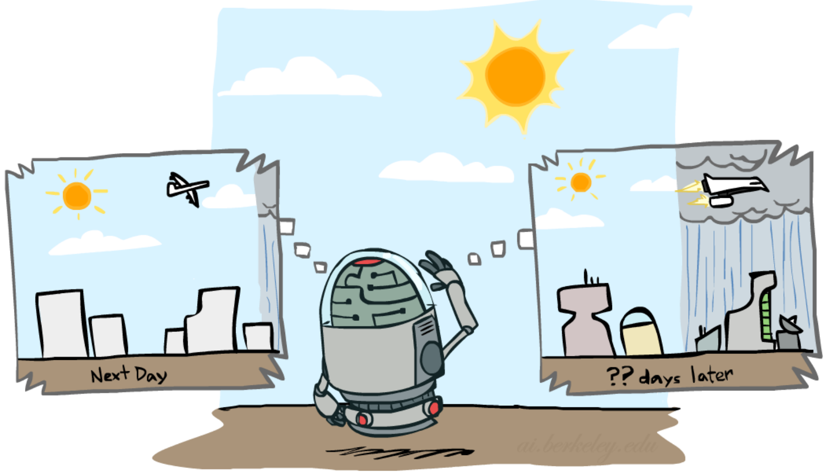
- For most chains:
  - Influence of the initial distribution gets less and less over time.
  - The distribution we end up in is independent of the initial distribution
- Stationary distribution:
  - The distribution we end up with is called the stationary distribution P_∞ of the chain
  - It satisfies

$$P_\infty(X) = P_{\infty+1}(X) = \sum_x P(X|x)P_\infty(x)$$

Example: Stationary Distributions

- Question: What's P(X) at time t = infinity?

$$X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4 \dashrightarrow$$



$$P_\infty(sun) = P(sun|sun)P_\infty(sun) + P(sun|rain)P_\infty(rain)$$
$$P_\infty(rain) = P(rain|sun)P_\infty(sun) + P(rain|rain)P_\infty(rain)$$

$$P_\infty(sun) = 0.9P_\infty(sun) + 0.3P_\infty(rain)$$
$$P_\infty(rain) = 0.1P_\infty(sun) + 0.7P_\infty(rain)$$

X_{t-1}	X_t	$P(X_t \mid X_{t-1})$
sun	sun	0.9
sun	rain	0.1
rain	sun	0.3
rain	rain	0.7

$$\begin{pmatrix} 0.9 & 0.3 \\ 0.1 & 0.7 \end{pmatrix} \begin{pmatrix} p \\ 1-p \end{pmatrix} = \begin{pmatrix} p \\ 1-p \end{pmatrix}$$

$0.9p + 0.3(1-p) = p$

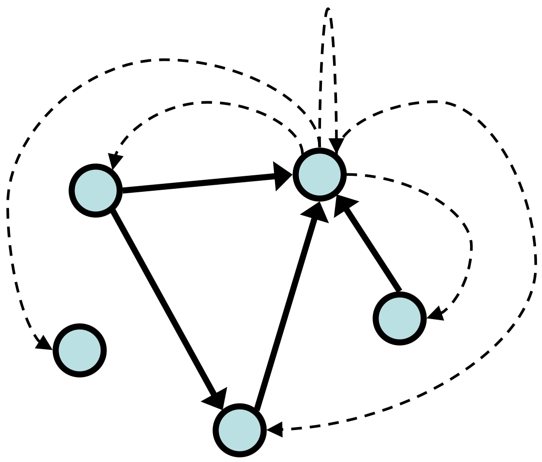
$p = 0.75$

Stationary distribution is <0.75,0.25> ***regardless of starting distribution***

Also: $P_\infty(sun) + P_\infty(rain) = 1$ $\Rightarrow$

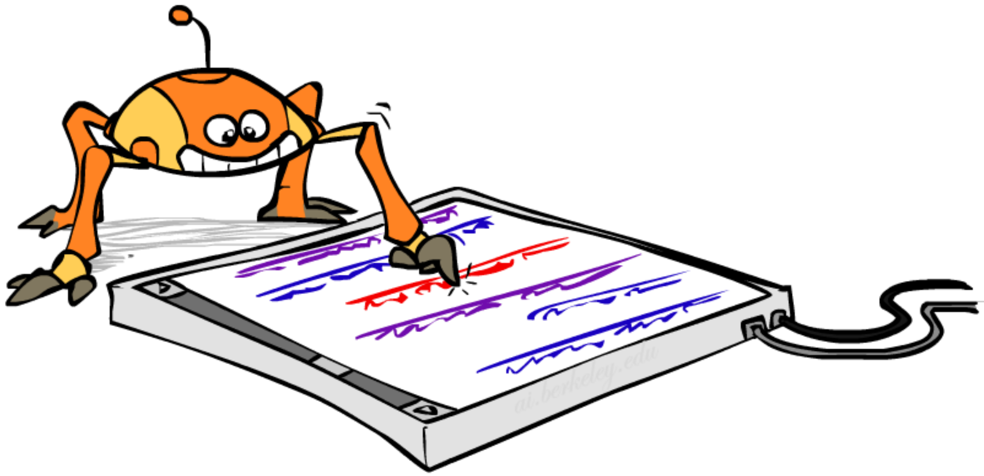
$$P_\infty(sun) = 3/4$$
$$P_\infty(rain) = 1/4$$

Application of Stationary Distribution: Web Link Analysis



- PageRank over a web graph
  - Each web page is a state
  - Initial distribution: uniform over pages
  - Transitions:
    - With prob. c, uniform jump to a random page (dotted lines, not all shown)
    - With prob. 1-c, follow a random outlink (solid lines)

- Stationary distribution
  - Will spend more time on highly reachable pages
  - E.g. many ways to get to the Acrobat Reader download page
  - Somewhat robust to link spam
  - Google 1.0 returned the set of pages containing all your keywords in decreasing rank, now all search engines use link analysis along with many other factors (rank actually getting less important over time)

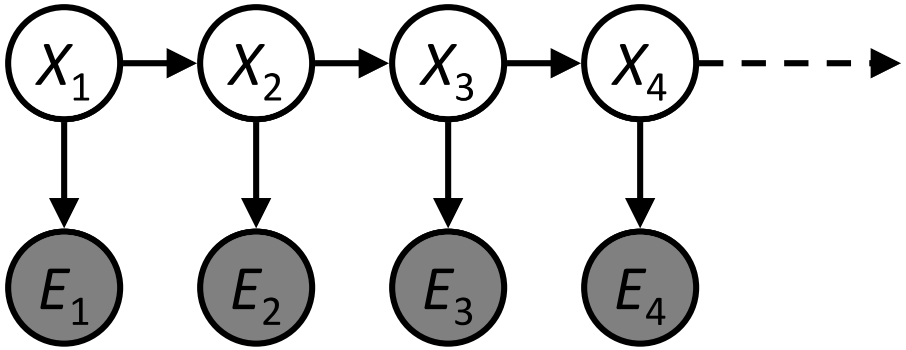


Hidden Markov Models

Hidden Markov Models

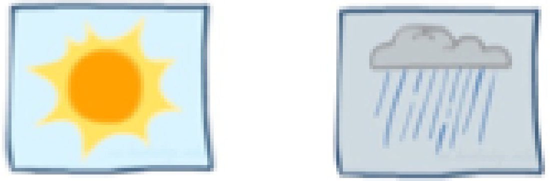
- Usually the true state is not observed directly

- Hidden Markov models (HMMs)
  - Underlying Markov chain over states X
  - You observe evidence E at each time step
  - X_t is a single discrete variable; E_t may be continuous and may consist of several variables

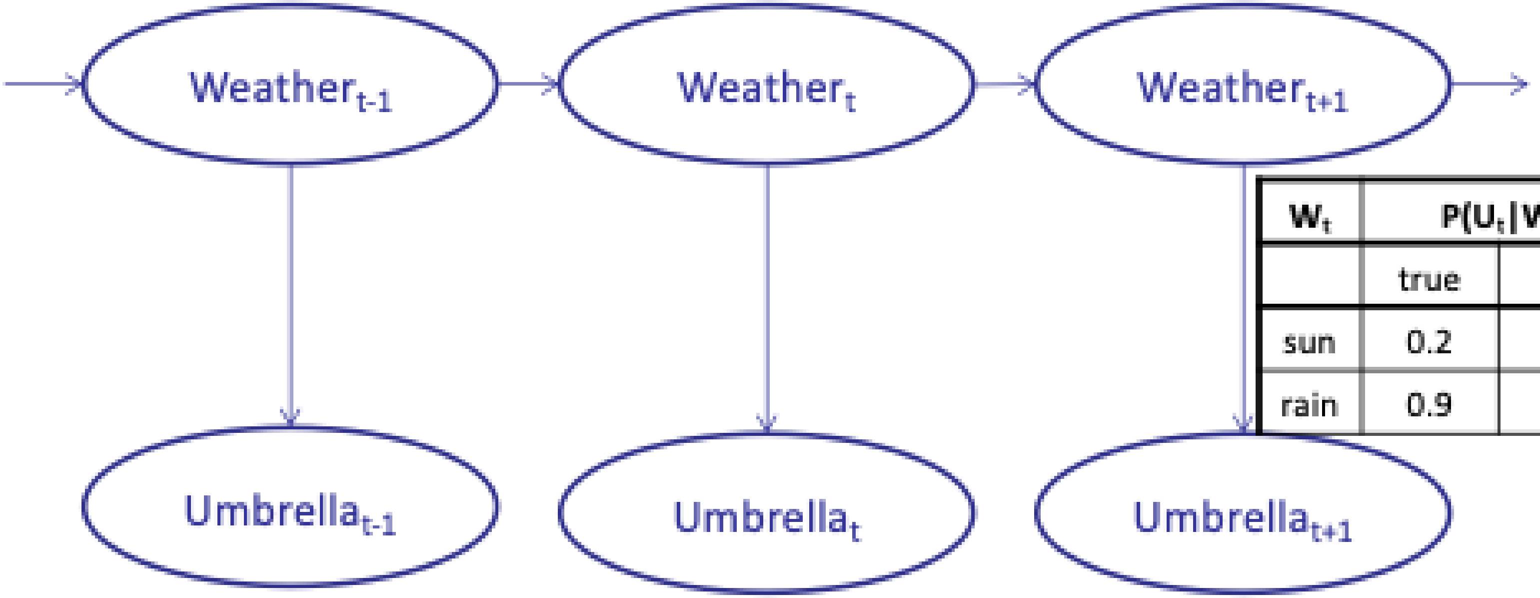


Example: Weather HMM

- An HMM is defined by:
  - Initial distribution: $P(X_0)$
  - Transition model: $P(X_t | X_{t-1})$
  - Sensor model: $P(E_t | X_t)$



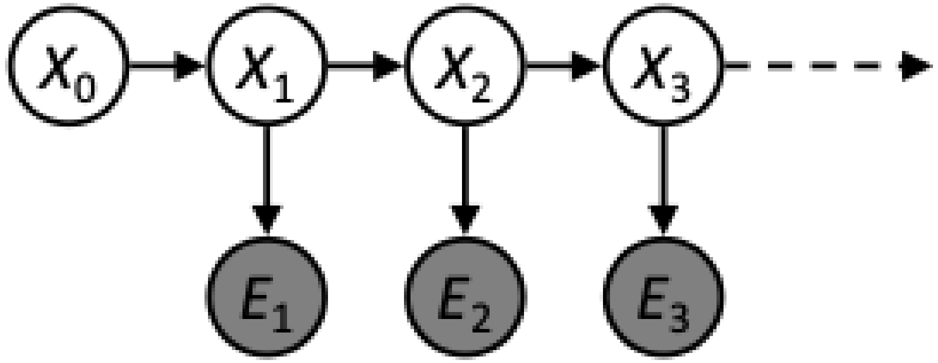
W_{t-1}	$P(W_t \mid W_{t-1})$	
	sun	rain
sun	0.9	0.1
rain	0.3	0.7



W_t	$P(U_t \mid W_t)$	
	true	false
sun	0.2	0.8
rain	0.9	0.1

HMM as probability model

- Joint distribution for Markov model: $P(X_0,\ldots,X_T) = P(X_0) \prod_{t=1:T} P(X_t \mid X_{t-1})$
- Joint distribution for hidden Markov model:
  $P(X_0,E_0,X_1,E_1,\ldots,X_T,E_T) = P(X_0) \prod_{t=1:T} P(X_t \mid X_{t-1})\, P(E_t \mid X_t)$
- Future states are independent of the past given the present
- Current evidence is independent of everything else given the current state
- Are evidence variables independent of each other?



Useful notation:

$X_{a:b} = X_a, X_{a+1}, \ldots, X_b$

Real HMM Examples

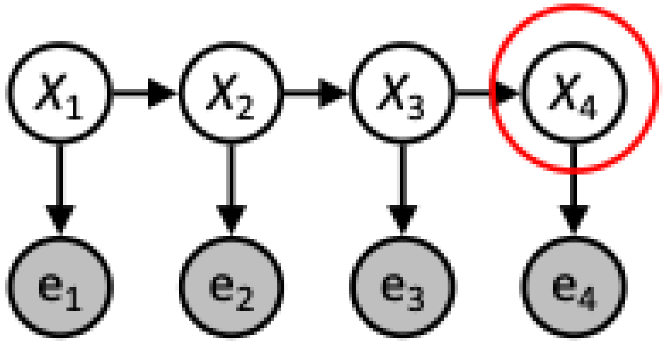
- Speech recognition HMMs:
  - Observations are acoustic signals (continuous valued)
  - States are specific positions in specific words (so, tens of thousands)
- Machine translation HMMs:
  - Observations are words (tens of thousands)
  - States are translation options
- Robot tracking:
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)
- Molecular biology:
  - Observations are nucleotides ACGT
  - States are coding/non-coding/start/stop/splice-site etc.

Inference tasks

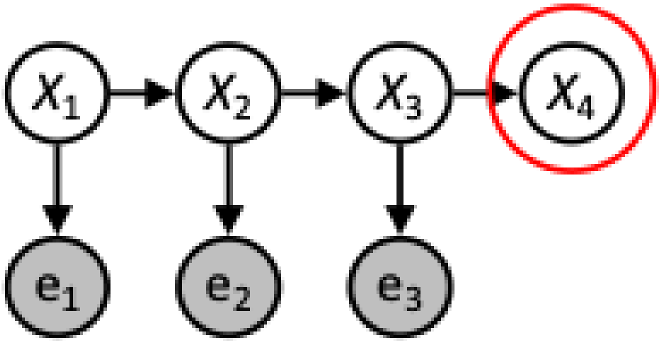
- ***Filtering***: $P(X_t | e_{1:t})$
  - ***belief state***—input to the decision process of a rational agent
- ***Prediction***: $P(X_{t+k} | e_{1:t})$ for $k > 0$
  - evaluation of possible action sequences; like filtering without the evidence
- ***Smoothing***: $P(X_k | e_{1:t})$ for $0 \leq k < t$
  - better estimate of past states, essential for learning
- ***Most likely explanation***: $\arg\max_{x_{1:t}} P(x_{1:t} \mid e_{1:t})$
  - speech recognition, decoding with a noisy channel

Inference tasks

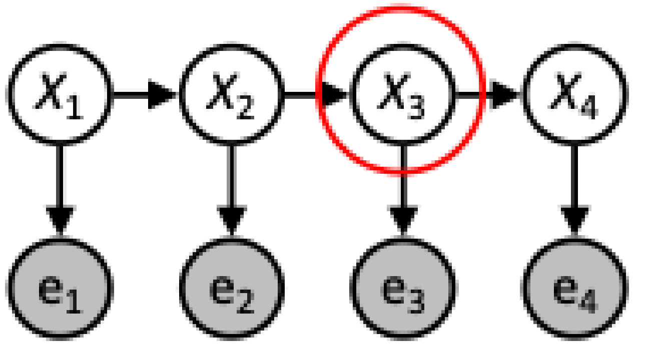
Filtering: $P(X_t | e_{1:t})$



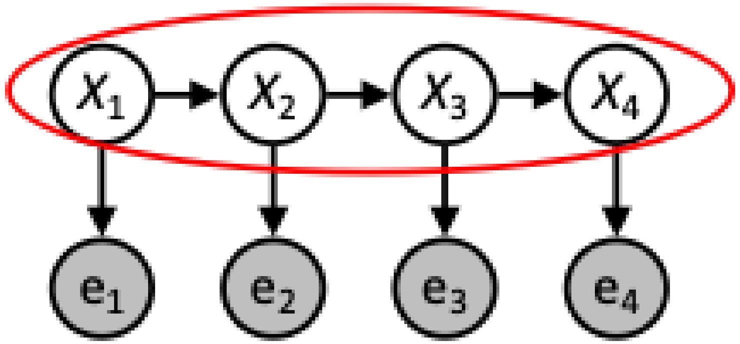
Prediction: $P(X_{t+k} | e_{1:t})$



Smoothing: $P(X_k | e_{1:t})$, $k<t$



Explanation: $P(X_{1:t} | e_{1:t})$



Filtering / Monitoring

- Filtering, or monitoring, or state estimation, is the task of maintaining the distribution $f_{1:t} = P(X_t | e_{1:t})$ over time

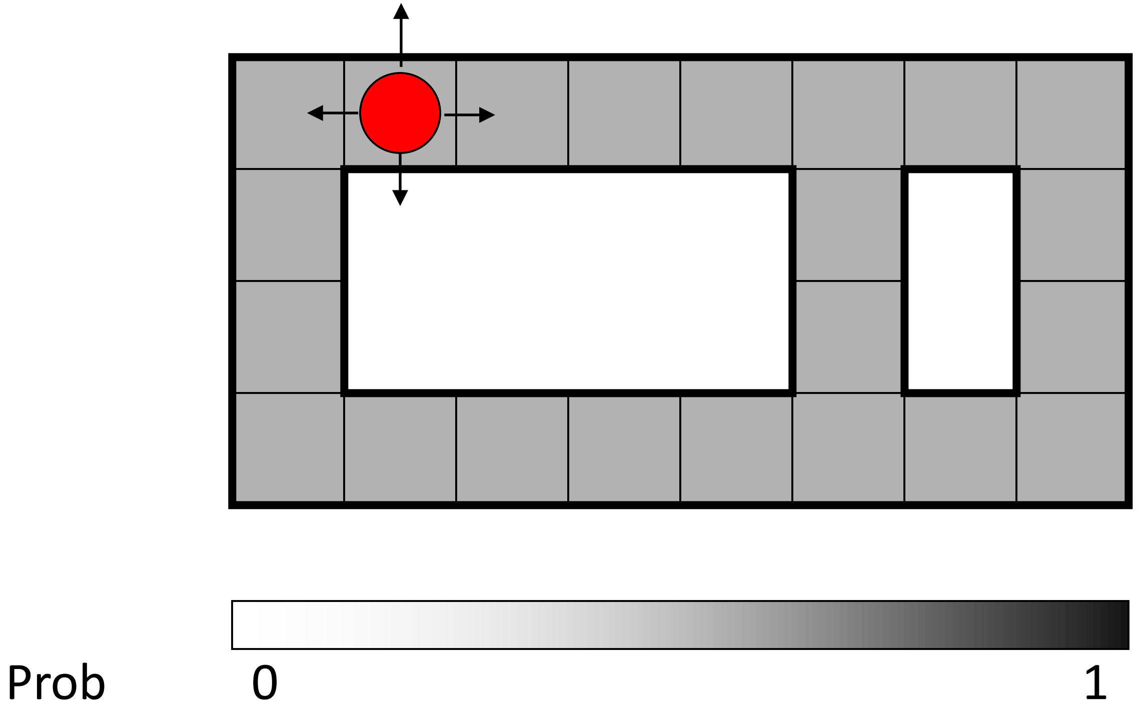
- We start with f_0 in an initial setting, usually uniform

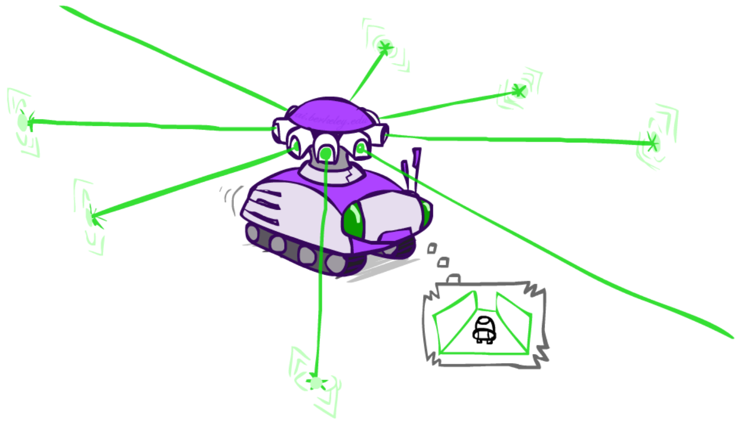
- Filtering is a fundamental task in engineering and science

- The Kalman filter (continuous variables, linear dynamics, Gaussian noise) was invented in 1960 and used for trajectory estimation in the Apollo program; core ideas used by Gauss for planetary observations; **>1,000,000** papers on Google Scholar

Example: Robot Localization

Example from Michael Pfeiffer



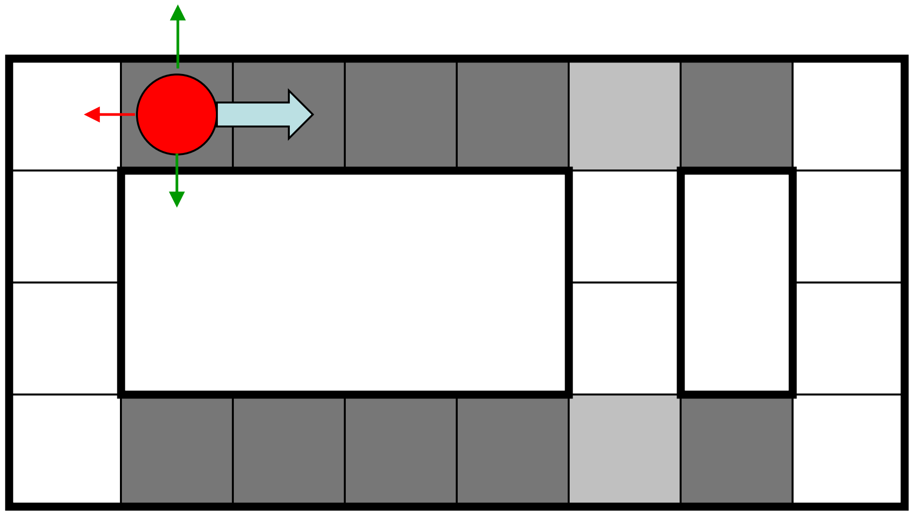


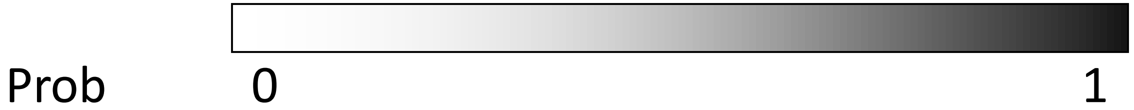
t=0

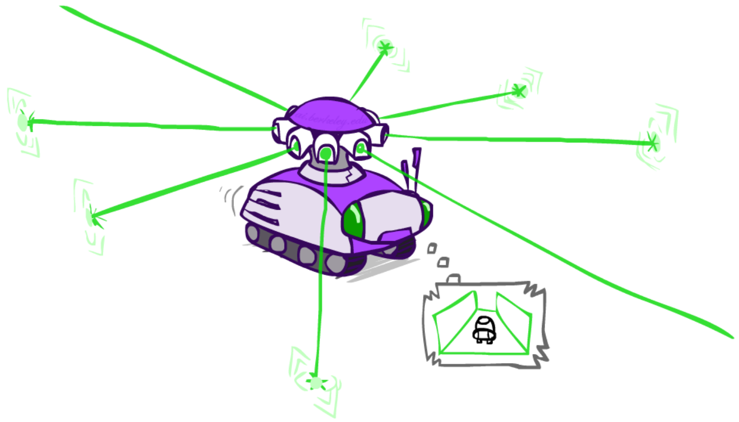
Sensor model: can read in which directions there is a wall, never more than 1 mistake

Motion model: may not execute action with small prob.

Example: Robot Localization



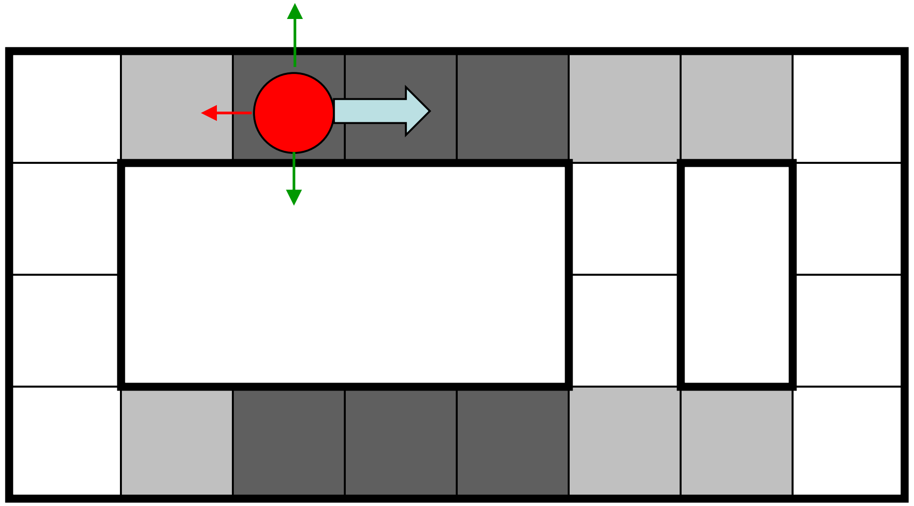




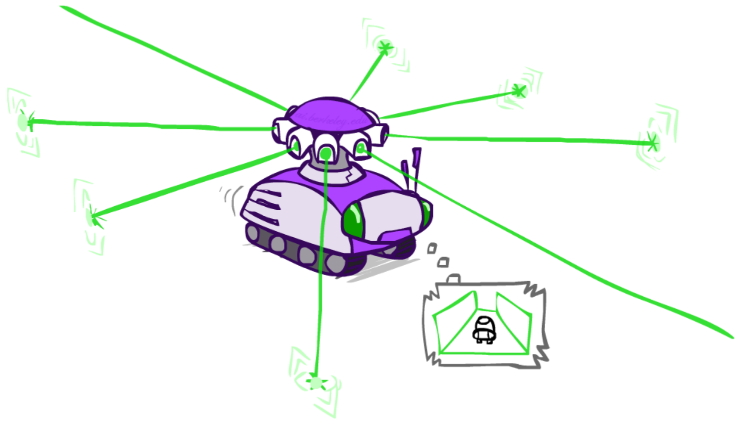
t=1

Lighter grey: was possible to get the reading, but less likely b/c required 1 mistake

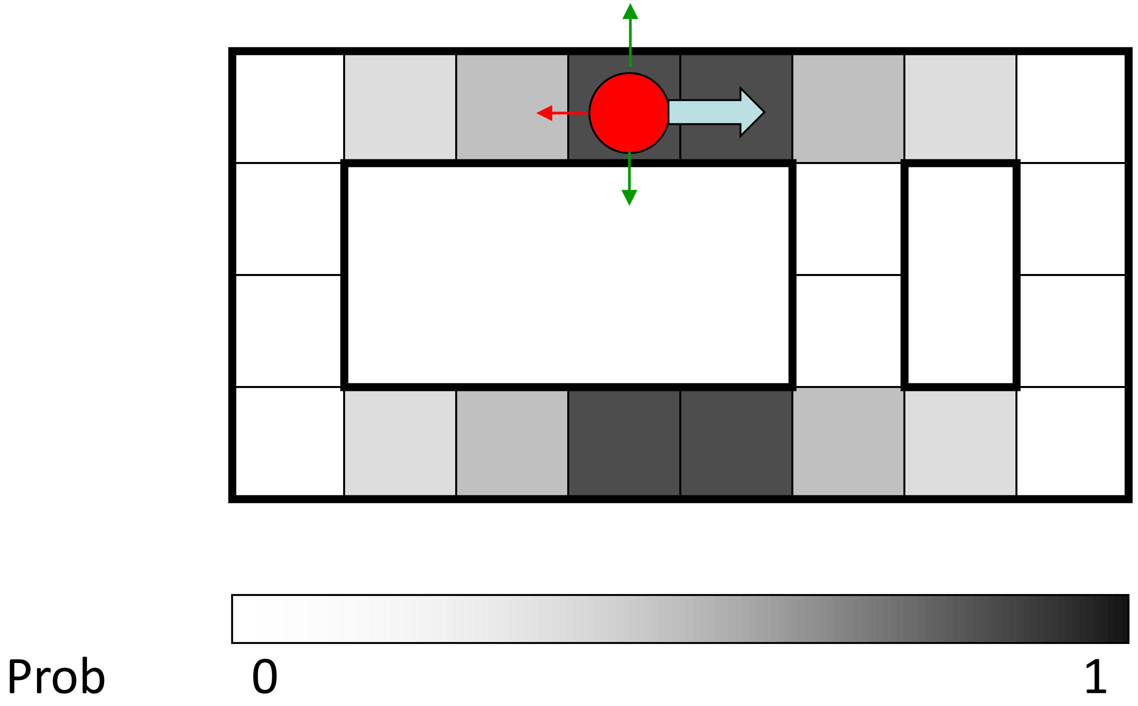
Example: Robot Localization

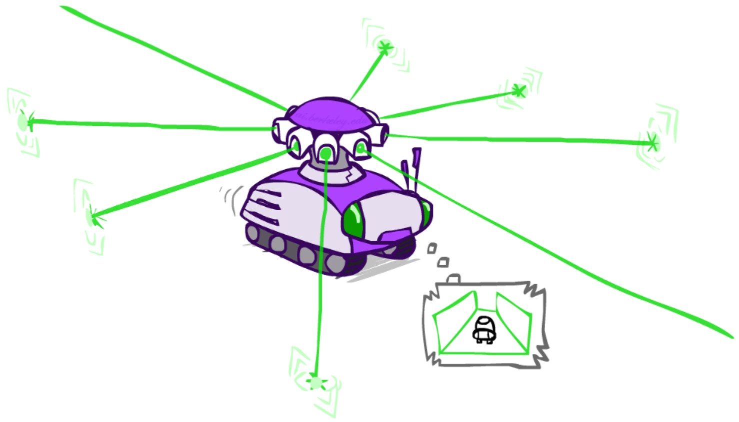


t=2



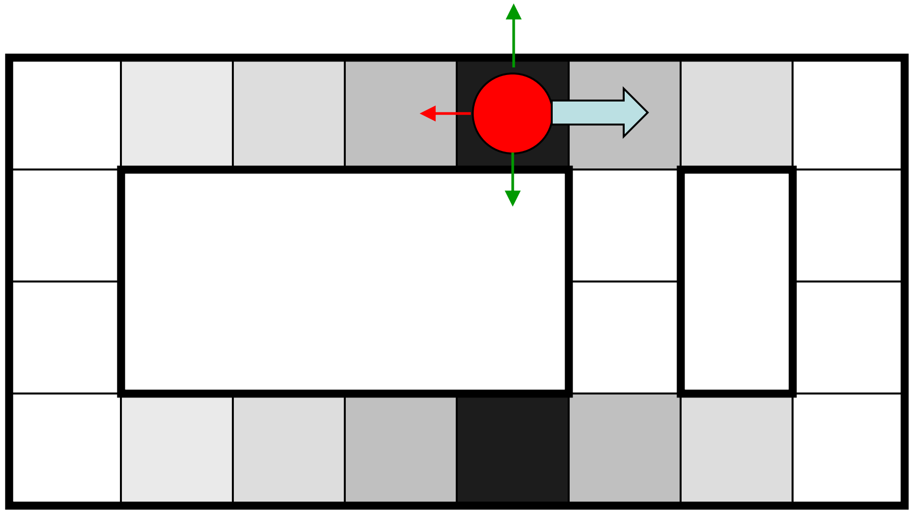
Example: Robot Localization

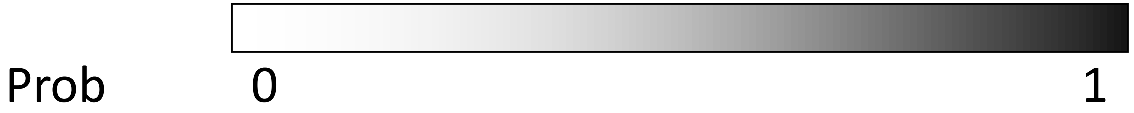


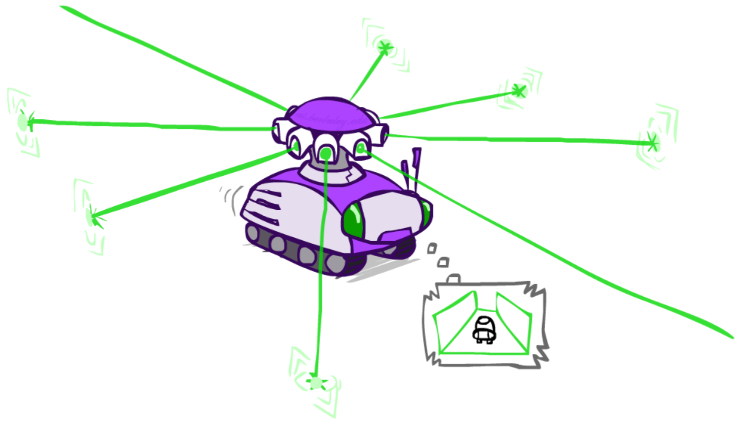


t=3

Example: Robot Localization

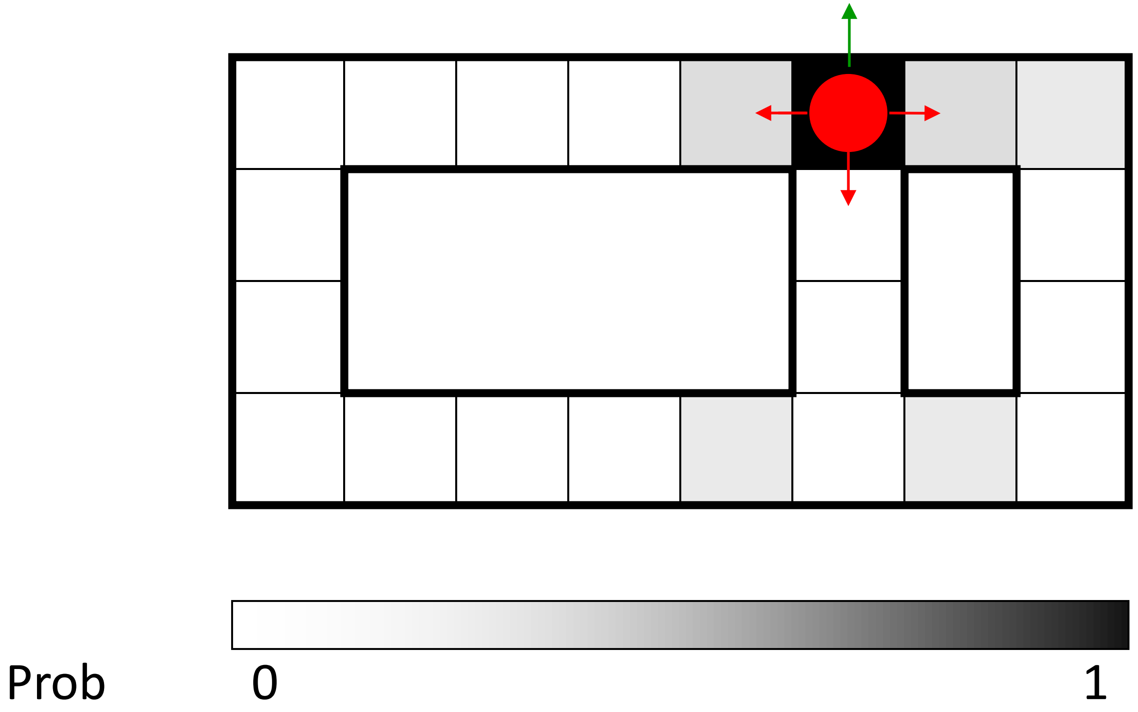


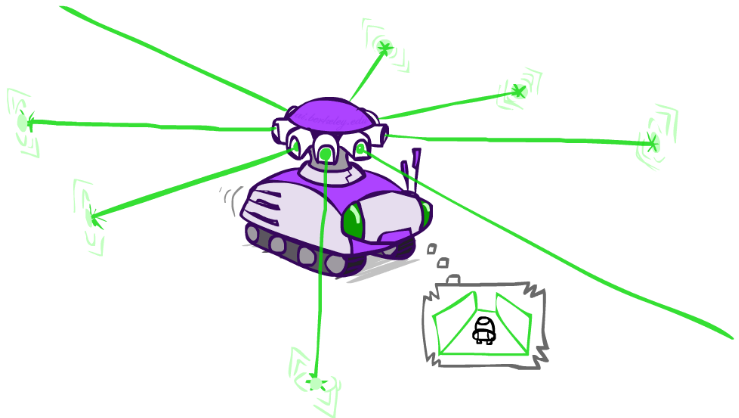




t=4

Example: Robot Localization

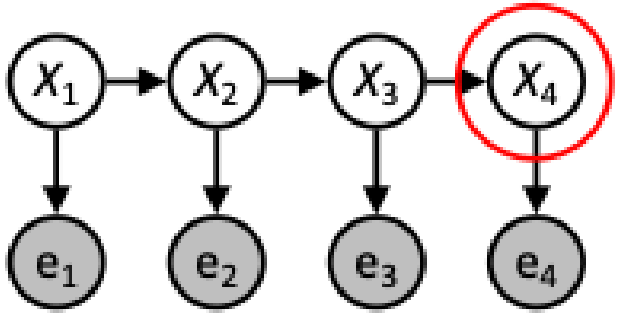




t=5

Filtering algorithm

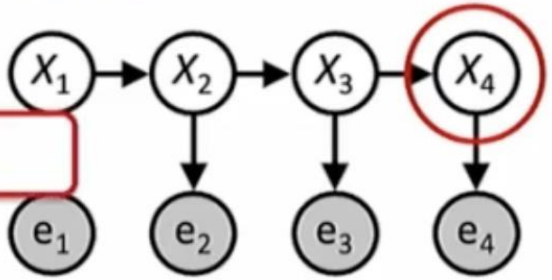
- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$

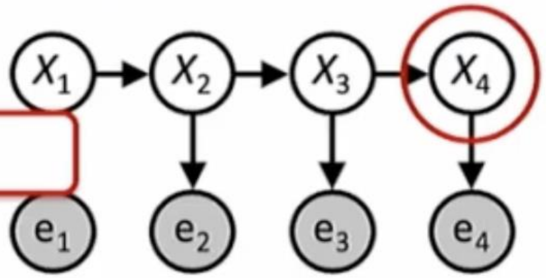


Apply Bayes' rule

- $P(X_{t+1} | e_{1:t+1}) = P(\underline{X_{t+1}} | e_{1:t}, \underline{e_{t+1}})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$

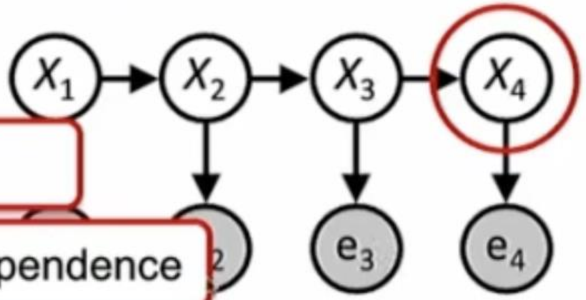


Apply Bayes' rule

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $\qquad = \alpha\, P(e_{t+1} | X_{t+1}, e_{1:t})\, P(X_{t+1} | e_{1:t})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



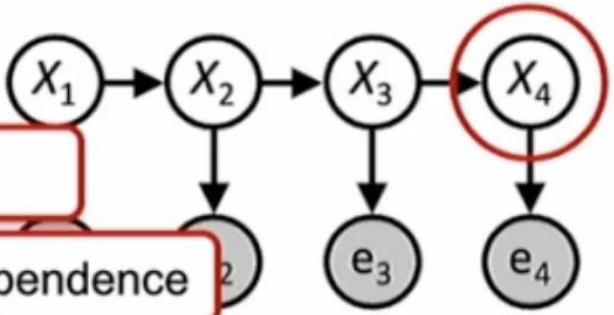
Apply Bayes' rule

Apply conditional independence

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1}, e_{1:t})\, P(X_{t+1} | e_{1:t})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1}|e_{1:t+1}) = g(e_{t+1}, P(X_t|e_{1:t}))$



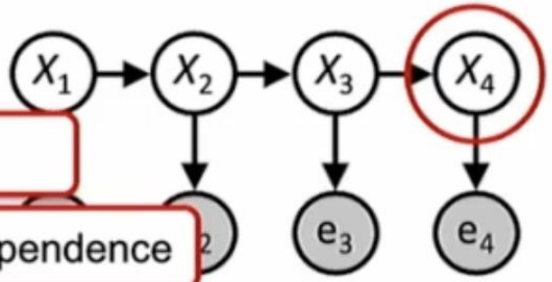
Apply Bayes' rule

Apply conditional independence

- $P(X_{t+1}|e_{1:t+1}) = P(X_{t+1}|e_{1:t}, e_{t+1})$
- $= \alpha\, P(e_{t+1}|X_{t+1}, e_{1:t})\, P(X_{t+1}|\, e_{1:t})$
- $= \alpha\, P(e_{t+1}|X_{t+1})\, P(X_{t+1}|\, e_{1:t})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1}|e_{1:t+1}) = g(e_{t+1}, P(X_t|e_{1:t}))$



Apply Bayes' rule

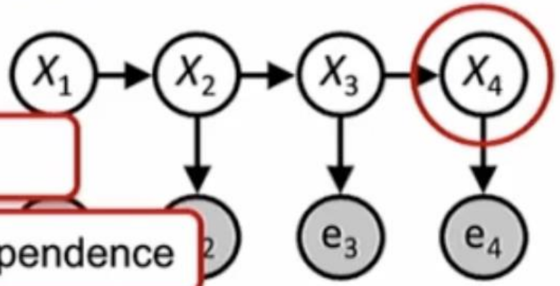
Apply conditional independence

- $P(X_{t+1}|e_{1:t+1}) = P(\underline{X_{t+1}}|e_{1:t}, \underline{e_{t+1}})$
- $= \alpha\, P(e_{t+1}|\underline{X_{t+1}, e_{1:t}})\, P(X_{t+1}| e_{1:t})$
- $= \alpha\, P(e_{t+1}|X_{t+1})\, \underline{P(X_{t+1}| e_{1:t})}$

Predict

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



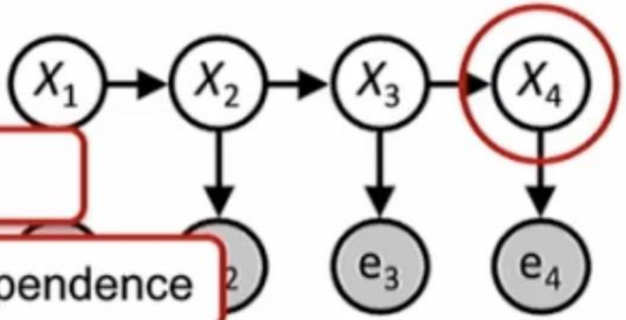
Apply Bayes' rule

Apply conditional independence

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $\quad = \alpha \, P(e_{t+1} | X_{t+1}, e_{1:t}) \, P(X_{t+1} | e_{1:t})$
- $\quad = \alpha \, \underbrace{P(e_{t+1} | X_{t+1})}_{\text{Update}} \, \underbrace{P(X_{t+1} | e_{1:t})}_{\text{Predict}}$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



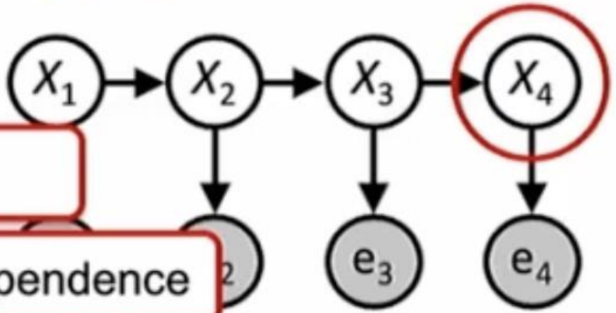
Apply Bayes' rule

Apply conditional independence

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $= \alpha\, P(e_{t+1} | X_{t+1}, e_{1:t})\, P(X_{t+1} | e_{1:t})$
- $= \underbrace{\alpha}_{\text{Normalize}}\, \underbrace{P(e_{t+1} | X_{t+1})}_{\text{Update}}\, \underbrace{P(X_{t+1} | e_{1:t})}_{\text{Predict}}$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



Apply Bayes' rule

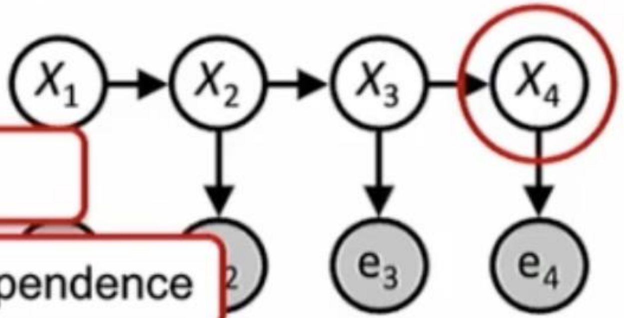
Apply conditional independence

Condition on X_t

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $= \alpha \, P(e_{t+1} | X_{t+1}, e_{1:t}) \, P(X_{t+1} | e_{1:t})$
- $= \alpha \, P(e_{t+1} | X_{t+1}) \, P(X_{t+1} | e_{1:t})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



Apply Bayes' rule

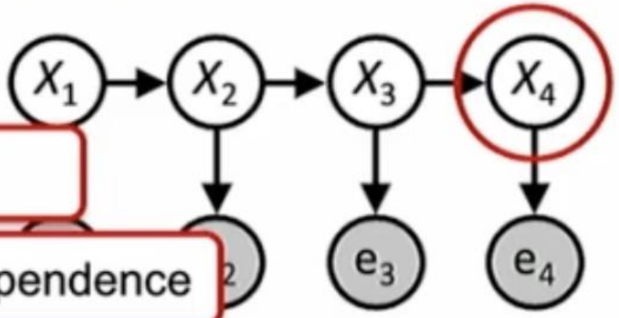
Apply conditional independence

Condition on X_t

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1}, e_{1:t})\, P(X_{t+1} | e_{1:t})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1})\, P(X_{t+1} | e_{1:t})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1}) \sum_{x_t} P(x_t | e_{1:t})\, P(X_{t+1} | x_t, e_{1:t})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



Apply Bayes' rule

Apply conditional independence

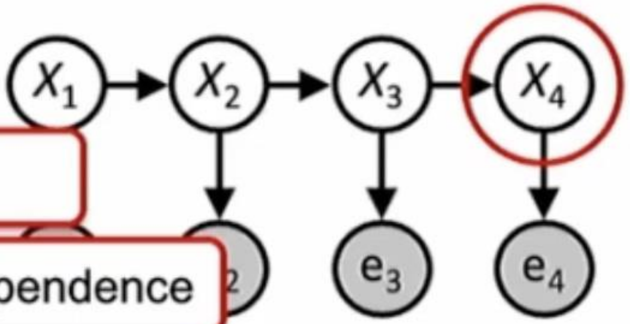
Condition on X_t

Apply conditional independence

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1}, e_{1:t})\, P(X_{t+1} | e_{1:t})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1})\, P(X_{t+1} | e_{1:t})$
- $\quad = \alpha\, P(e_{t+1} | X_{t+1}) \sum_{x_t} P(x_t | e_{1:t})\, P(X_{t+1} | x_t, e_{1:t})$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



Apply Bayes' rule

Apply conditional independence

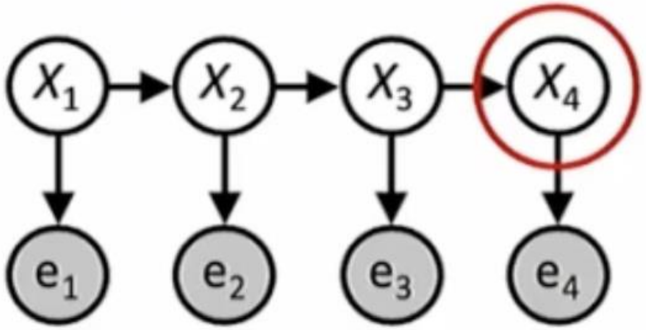
Condition on X_t

Apply conditional independence

- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $= \alpha P(e_{t+1} | X_{t+1}, e_{1:t}) P(X_{t+1} | e_{1:t})$
- $= \alpha P(e_{t+1} | X_{t+1}) P(X_{t+1} | e_{1:t})$
- $= \alpha P(e_{t+1} | X_{t+1}) \sum_{x_t} P(x_t | e_{1:t}) P(X_{t+1} | x_t, e_{1:t})$
- $= \alpha P(e_{t+1} | X_{t+1}) \sum_{x_t} P(x_t | e_{1:t}) P(X_{t+1} | x_t)$

Filtering algorithm

- Aim: devise a ***recursive filtering*** algorithm of the form
  - $P(X_{t+1} | e_{1:t+1}) = g(e_{t+1}, P(X_t | e_{1:t}))$



- $P(X_{t+1} | e_{1:t+1}) = P(X_{t+1} | e_{1:t}, e_{t+1})$
- $= \alpha\, P(e_{t+1} | X_{t+1}, e_{1:t})\, P(X_{t+1} | e_{1:t})$
- $= \alpha\, P(e_{t+1} | X_{t+1})\, P(X_{t+1} | e_{1:t})$
- $= \alpha\, P(e_{t+1} | X_{t+1}) \sum_{x_t} P(x_t | e_{1:t})\, P(X_{t+1} | x_t, e_{1:t})$
- $= \alpha\, \underbrace{P(e_{t+1} | X_{t+1})}_{\text{Given by HMM}} \sum_{x_t} \underbrace{P(x_t | e_{1:t})}_{\text{Pre-computed}}\, \underbrace{P(X_{t+1} | x_t)}_{\text{Given by HMM}}$