(Hidden) Markov Models
Wrap-up

With slides from Dan Klein and Pieter Abbiel

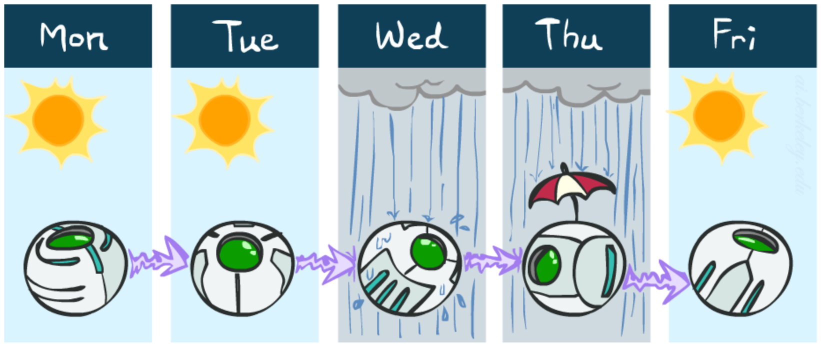
Today

- Review Project 2 solution (optional)
- BN Inference with time
  - Markov Chains
  - Hidden Markov Models
- Project 3: Ghostbusters
- Wrap-up: Search and Inference for ”intelligent” agents
- **[optional] Midterm review today 5:30pm in W301**

Example Markov Chain: Weather

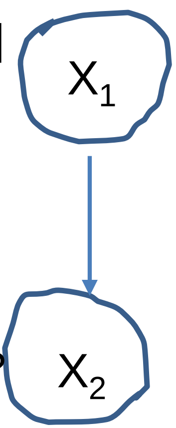
- States: X = {rain, sun}

- Initial distribution: 1.0 sun



Two new ways of representing the same BN

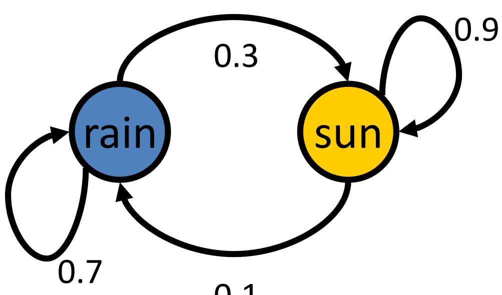
[rain=0]
[sun=1]

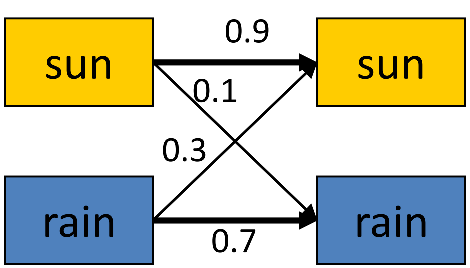


$P(X_2)$=?

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

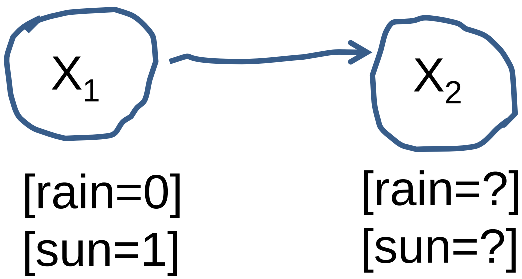
X_{t-1}	X_t	$P(X_t \mid X_{t-1})$
sun	sun	0.9
sun	rain	0.1
rain	sun	0.3
rain	rain	0.7

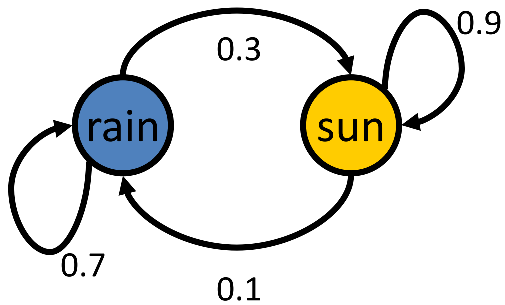




Weather Markov Chain: Prediction

- Initial distribution: 1.0 sun





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

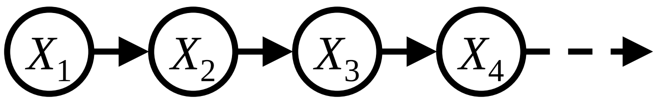
- What is the probability distribution after one step?

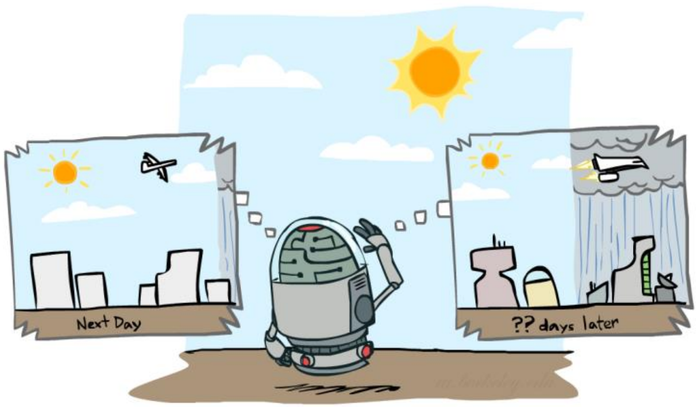
$$P(X_2 = \text{sun}) = P(X_2 = \text{sun} | X_1 = \text{sun}) P(X_1 = \text{sun}) + P(X_2 = \text{sun} | X_1 = \text{rain}) P(X_1 = \text{rain})$$

$$0.9 \cdot 1.0 + 0.3 \cdot 0.0 = 0.9$$

Prediction: Mini-Forward Algorithm

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





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

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

Forward simulation

Example Run of Mini-Forward Algorithm

- From initial observation of sun

$$\begin{array}{ccccccccc}
\left\langle \begin{matrix} 1.0 \\ 0.0 \end{matrix} \right\rangle & & \left\langle \begin{matrix} 0.9 \\ 0.1 \end{matrix} \right\rangle & & \left\langle \begin{matrix} 0.84 \\ 0.16 \end{matrix} \right\rangle & & \left\langle \begin{matrix} 0.804 \\ 0.196 \end{matrix} \right\rangle & \Rightarrow & \left\langle \begin{matrix} 0.75 \\ 0.25 \end{matrix} \right\rangle \\
\mathrm{P}(X_1) & & \mathrm{P}(X_2) & & \mathrm{P}(X_3) & & \mathrm{P}(X_4) & & \mathrm{P}(X_\infty)
\end{array}$$

- From initial observation of rain

$$\begin{array}{ccccccccc}
\left\langle \begin{matrix} 0.0 \\ 1.0 \end{matrix} \right\rangle & & \left\langle \begin{matrix} 0.3 \\ 0.7 \end{matrix} \right\rangle & & \left\langle \begin{matrix} 0.48 \\ 0.52 \end{matrix} \right\rangle & & \left\langle \begin{matrix} 0.588 \\ 0.412 \end{matrix} \right\rangle & \Rightarrow & \left\langle \begin{matrix} 0.75 \\ 0.25 \end{matrix} \right\rangle \\
\mathrm{P}(X_1) & & \mathrm{P}(X_2) & & \mathrm{P}(X_3) & & \mathrm{P}(X_4) & & \mathrm{P}(X_\infty)
\end{array}$$

- From yet another initial distribution $\mathrm{P}(X_1)$:

$$\begin{array}{ccccc}
\left\langle \begin{matrix} p \\ 1-p \end{matrix} \right\rangle & & \ldots & \Rightarrow & \left\langle \begin{matrix} 0.75 \\ 0.25 \end{matrix} \right\rangle \\
\mathrm{P}(X_1) & & & & \mathrm{P}(X_\infty)
\end{array}$$

Markov Chains: (Simplified)? Matrix View

- A probability vector **x** = $(x_1, \ldots x_n)$ tells us our *belief distribution which state we are at a given time.*
- For convenience we represent X as a *row* vector
  X_1: [1, 0] (sun=1, rain=0)
  X_2: [0.9, 0.1] (sun=0.9, rain=0.1)
- Transition Probability *Matrix* **P** = (re-formated) CPT

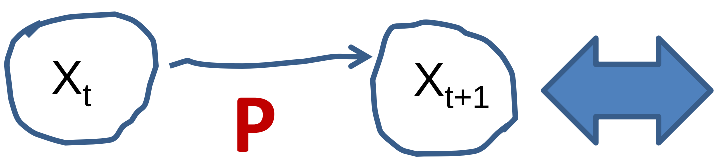
CPT $P(X_t \mid X_{t-1})$:

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

$X_t \backslash X_{t+1}$	sun	rain
sun	0.9	0.1
rain	0.3	0.7

Forward Algorithm in vector form

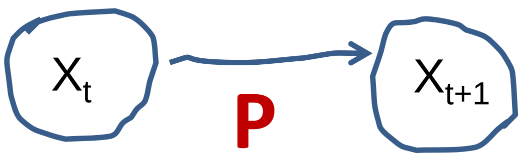
- If the probability vector is **x** $= (x_1, \ldots x_n)$ at this step, what is **x'** at the next step?
- Recall that row *i* of **the transition probability Matrix P** tells us where we go next from state *i*.
- So from **x**, our next state is computed as **xP**.



$$\underbrace{\boxed{x_1, x_2, x_3 \ldots, x_n}}_{\mathbf{X_t}} \times \underbrace{\begin{matrix} p(x_1|x_1)\ p(x_2|x_1) \ldots\ p(x_n|x_1) \\ p(x_1|x_2)\ p(x_2|x_2) \ldots\ p(x_n|x_2) \\ \ldots \\ \ldots \\ p(x_1|x_n)\ p(x_2|x_n) \ldots\ p(x_n|x_n) \end{matrix}}_{\mathbf{P}}$$

Forward Algorithm in vector form

- So from **x**, our next belief/state Prob is **xP**.



$$\underbrace{\begin{bmatrix} x_1, & x_2, & x_3 \ldots, & x_n \end{bmatrix}}_{\mathbf{X_t}} \; \mathrm{X} \; \underbrace{\begin{bmatrix} p(x_1|x_1) & p(x_2|x_1) & \ldots & p(x_n|x_1) \\ p(x_1|x_2) & p(x_2|x_2) & \ldots & p(x_n|x_2) \\ \ldots & & & \\ \ldots & & & \\ p(x_1|x_n) & p(x_2|x_n) & \ldots & p(x_n|x_n) \end{bmatrix}}_{\mathbf{P}}$$

$$\Downarrow$$

$$\underbrace{\begin{bmatrix} x'_1, & x'_2, & x'_3 \ldots, & x'_n \end{bmatrix}}_{\mathbf{X_{t+1}}}$$

$$x'_1 = x_1 * p(x_1|x_1) + x_2 * p(x_1|x_2) + x_3 * p(x_1|x_3) + \ldots + x_n * p(x_1|x_n)$$

$$x'_2 = x_1 * p(x_2|x_1) + x_2 * p(x_2|x_2) + x_3 * p(x_2|x_3) + \ldots + x_n * p(x_2|x_n)$$

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

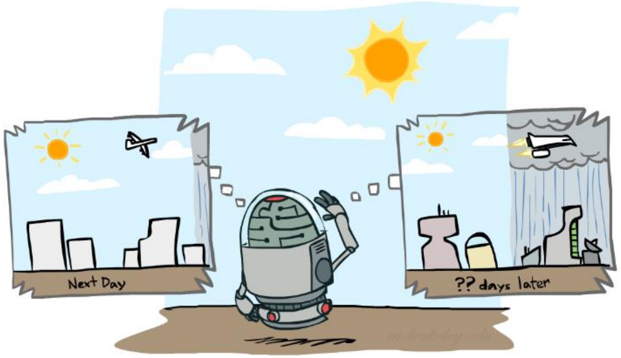
- Stationary distribution:
  - The distribution we end up with is called the stationary distribution P_∞ of the Markov chain
  - It satisfies

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

Example: Stationary Distributions

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

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



1
$$P_\infty(sun) = P(sun|sun)P_\infty(sun) + P(sun|rain)P_\infty(rain)$$
$$P_\infty(rain) = P(rain|sun)P_\infty(sun) + P(rain|rain)P_\infty(rain)$$

2
$$P_\infty(sun) = 0.9P_\infty(sun) + 0.3P_\infty(rain)$$
$$P_\infty(rain) = 0.1P_\infty(sun) + 0.7P_\infty(rain)$$

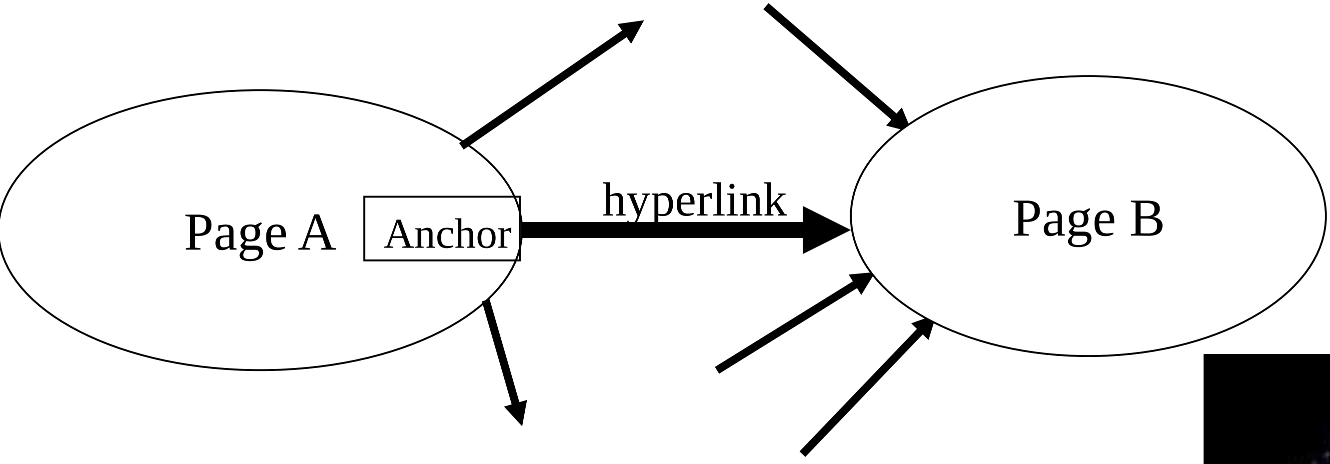
3
$$P_\infty(sun) = 3P_\infty(rain)$$
$$P_\infty(rain) = 1/3P_\infty(sun)$$

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

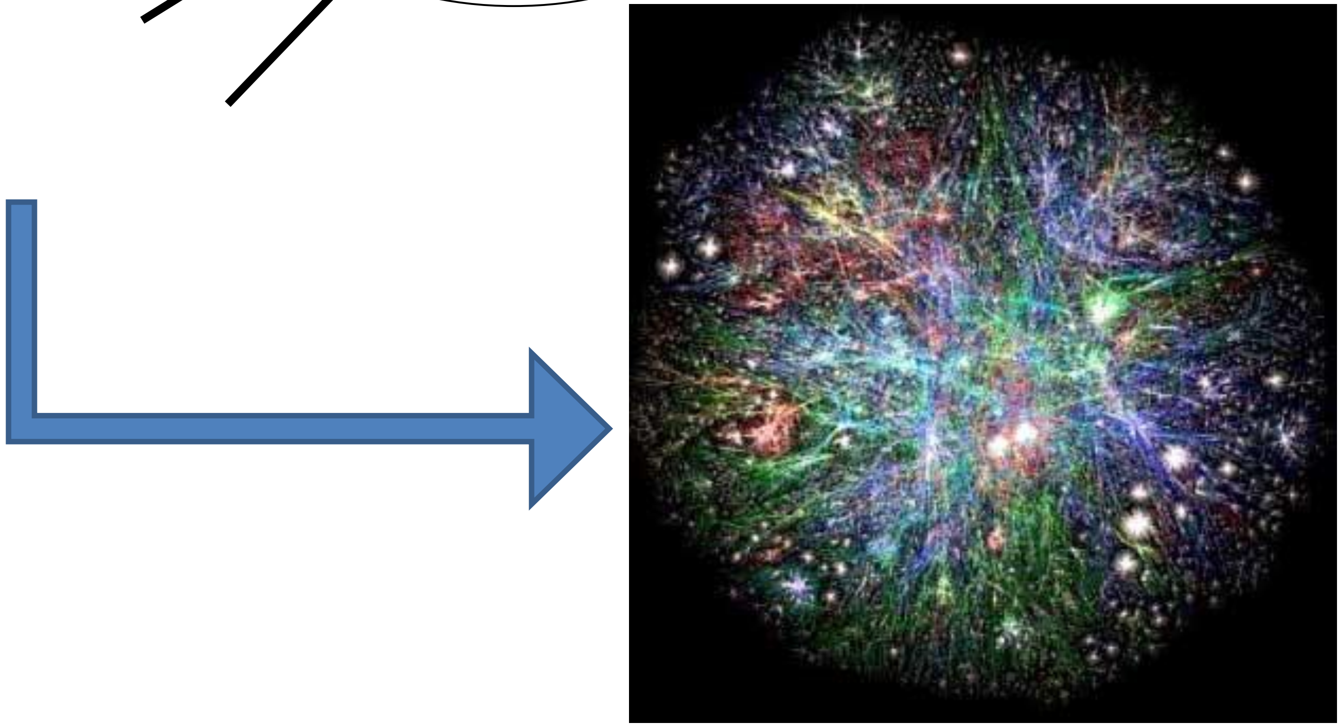
$$\Rightarrow \quad P_\infty(sun) = 3/4$$
$$P_\infty(rain) = 1/4$$

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

Application: PageRank

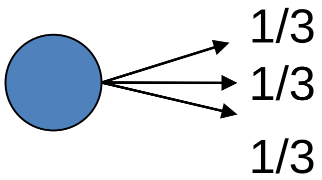


Web: Directed graph



Random (Markov) Surfer Model

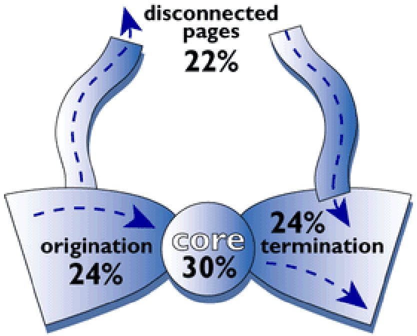
- Imagine a browser doing a random walk on web pages:
  - Start at a random page



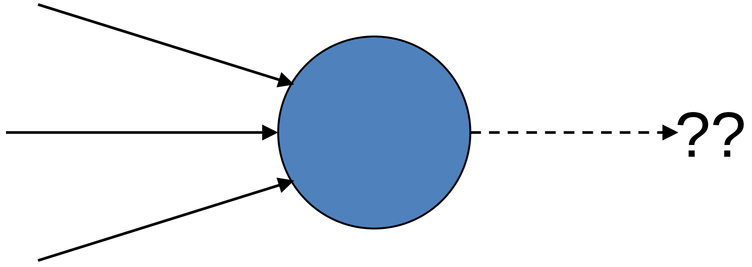
  - At each step, go out of the current page along one of the links on that page, with uniform prob.
- "In the steady state" each page has a long-term visit rate - use this as the page's PageRank score.
  - Only works for "ergodic" chains (no dead-ends)

Not quite enough

- The web is full of dead-ends.

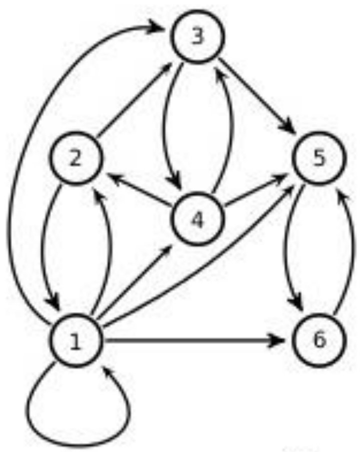


  - Random walk can get stuck in dead-ends.
  - Makes no sense to talk about long-term visit rates.



Add Teleportation

- Introduce a "random teleportation" probability, $E(p)$, that sends a surfer to any page with small (random) probability.



The random surfer model

At a node …

1. *follow edges* with prob α
2. *do something else* with prob $(1-\alpha)$

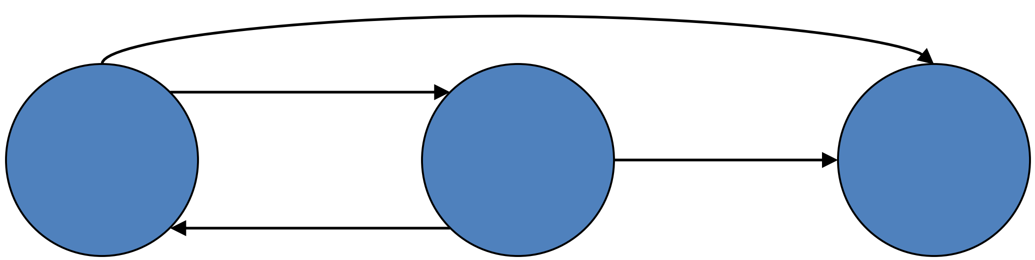
The important pages are the places we are most likely to find the random surfer

$$R(p) = c\left(\sum_{q:q\to p} \frac{R(q)}{N_q} + E(p) \right)$$

R(p) = Steady State Markov Prob!

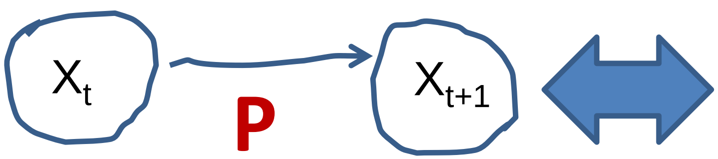
- For all pages *i*, $\sum_{j=1}^{n} P_{ij} = 1.$ (normalization)

- Represent the teleporting random walk as a Markov chain, for this case:



Iterative (Forward) Way of computing **a**

- Recall, regardless of where we start, we eventually reach the steady state **a**.
- Start with any distribution (say **x**=(*10…0*)).
- After one step, we're at **xP**;
- after two steps at $\mathbf{xP}^2$, then $\mathbf{xP}^3$ and so on.
- "Eventually" means for "large" k, $\mathbf{xP}^k = \mathbf{a}$.
- Algorithm: multiply **x** by increasing powers of **P** until the product converges.



$$\underbrace{x_1, x_2, x_3 \ldots, x_n}_{\mathbf{X_t}} \quad \mathbf{x} \quad \underbrace{\begin{matrix} p(x_1|x_1)\ p(x_2|x_1)\ldots\ p(x_n|x_1) \\ p(x_1|x_2)\ p(x_2|x_2)\ldots\ p(x_n|x_2) \\ \ldots \\ \ldots \\ p(x_1|x_n)\ p(x_2|x_n)\ldots\ p(x_n|x_n) \end{matrix}}_{\mathbf{P}}$$

*PageRank Algorithm

*Page = *Larry Page*, not web page!

Let S be the total set of pages.

Let $\forall p \in S$: $E(p) = \alpha/|S|$ (for some $0<\alpha<1$, e.g. 0.15)

Initialize $\forall p \in S$: $R(p) = 1/|S|$

Until ranks do not change (much) (*convergence*)

For each $p \in S$:

$$R'(p) = \left[(1-\alpha) \sum_{q:q \to p} \frac{R(q)}{N_q} \right] + E(p)$$

$$c = 1/\sum_{p \in S} R'(p)$$

For each $p \in S$: $R(p) = cR'(p)$ (*normalize*)

Speed of Convergence

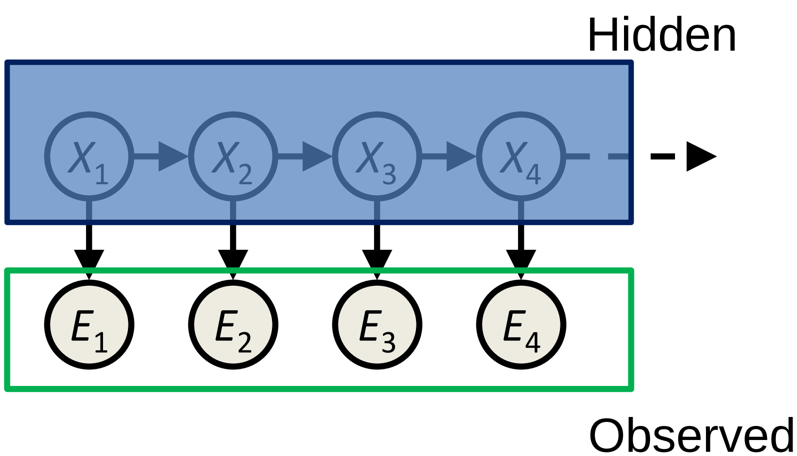
- Early experiments on Google used 322 million links.
- PageRank algorithm converged (within small tolerance) in about 52 iterations.
- Number of iterations required for convergence is empirically O(log *n*) (where *n* is the number of links).
- Speed of convergence shown related to sizes of eigenvectors of the matrix P

Which page has highest PageRank?

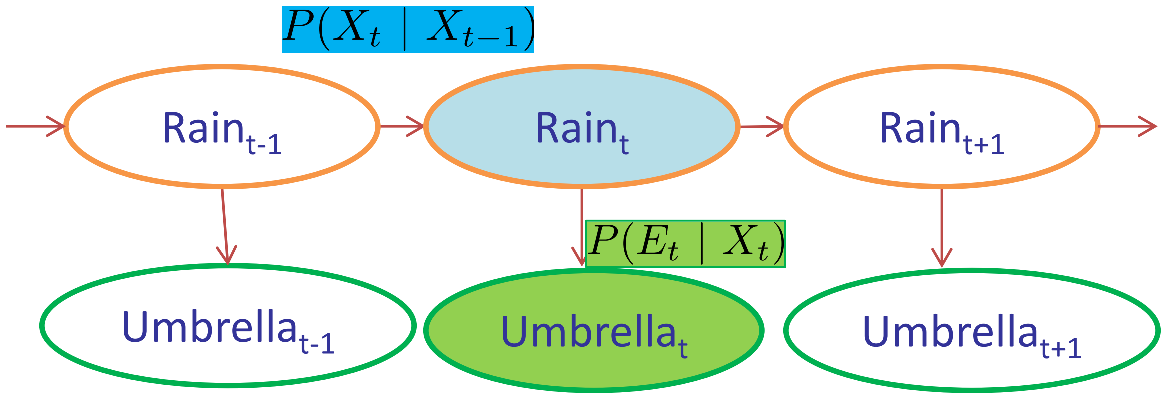
- c.f. 1997: Netscape!
- c.f. 2005: Wikipedia!
  - Maybe not (hard to measure externally)
- Some other sites with high PageRank
  - Google
  - MS Internet Explorer, Firefox homepages

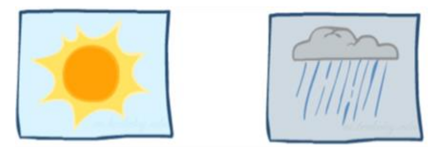
Hidden Markov Models

- Markov chains not so useful for most agents
  - Need observations to update your beliefs
- Hidden Markov models (HMMs)
  - Underlying Markov chain over states X
  - **You observe outputs (effects) at each time step**



Example: Weather HMM



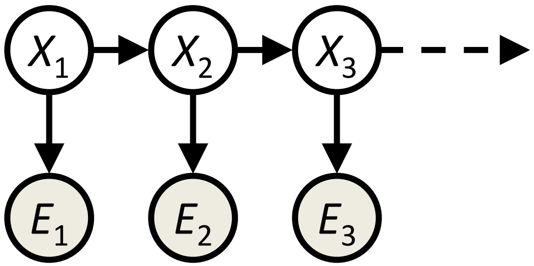


- An HMM is defined by:
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X_t \mid X_{t-1})$
  - **Emissions/observations:** $\boxed{P(E_t \mid X_t)}$

R_t	R_{t+1}	$P(R_{t+1} \mid R_t)$
+r	+r	0.7
+r	-r	0.3
-r	+r	0.3
-r	-r	0.7

R_t	U_t	$P(U_t \mid R_t)$
+r	+u	0.9
+r	-u	0.1
-r	+u	0.2
-r	-u	0.8

Joint Distribution of an HMM



- Joint distribution:

$$P(X_1, E_1, X_2, E_2, X_3, E_3) = P(X_1)P(E_1|X_1)P(X_2|X_1)P(E_2|X_2)P(X_3|X_2)P(E_3|X_3)$$

- More generally:

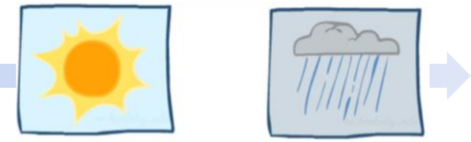
$$P(X_1, E_1, \ldots, X_T, E_T) = P(X_1)P(E_1|X_1)\prod_{t=2}^{T} P(X_t|X_{t-1})P(E_t|X_t)$$

- 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?

Filtering / Monitoring

- Filtering, or monitoring, is the task of tracking the distribution $B_t(X) = P_t(X_t \mid e_1, \ldots, e_t)$ (the belief state) over time
- We start with $B_1(X)$ in an initial setting, usually uniform
- As time passes, or we get observations, we update B(X)

Filtering: Weather HMM



Marginalize $B(+r)=B'(+r)*P(+r|+r) + B'(-r)*P(+r|-r)$

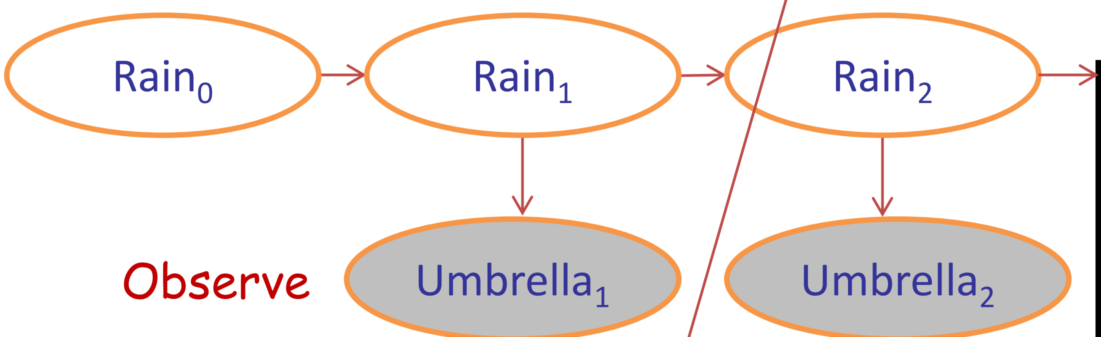
$B'(+r) = 0.5$
$B'(-r) = 0.5$

$B'(+r) = 0.627$
$B'(-r) = 0.373$

$B(+r)=0.5*0.7+$
$0.5*0.3$
$= 0.5$

$B(+r) = 0.5$
$B(-r) = 0.5$

$B(+r) = 0.883$
$B(-r) = 0.117$



Observe

R_t	R_{t+1}	$P(R_{t+1}\|R_t)$
+r	+r	0.7
+r	-r	0.3
-r	+r	0.3
-r	-r	0.7

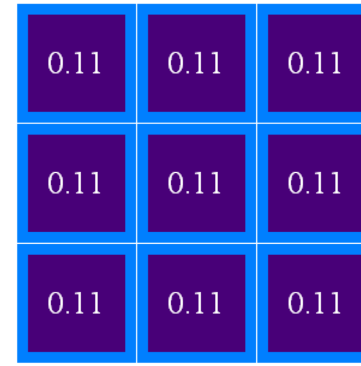
R_t	U_t	$P(U_t\|R_t)$
+r	+u	0.9
+r	-u	0.1
-r	+u	0.2
-r	-u	0.8

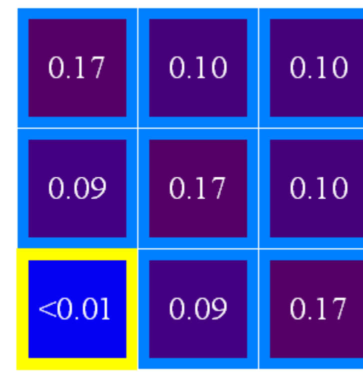
$B(+r)=0.5*0.9 = 0.45$
$B(-r)=0.5*0.2 = 0.1$

Normalize
$B(+r)=0.818$
$B(-r)=0.182$

Ghostbusters Filtering (project 3)

- Let's say we have two distributions:
  - Prior distribution over ghost location: P(G)
    - Let's say this is uniform
  - Sensor reading model: P(R | G)
    - Given: we know what our sensors do
    - R = reading color measured at (1,1)
    - E.g. P(R = yellow | G=(1,1)) = 0.1



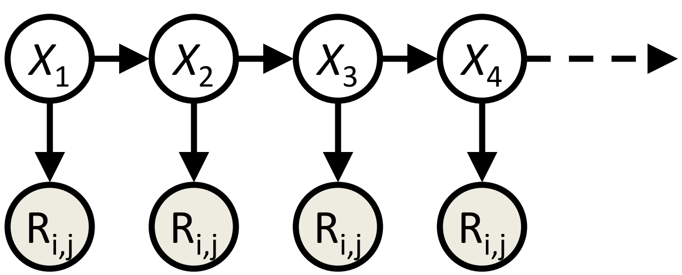


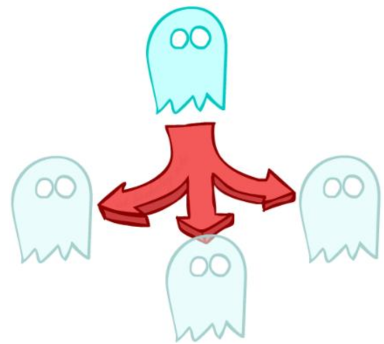
- Can calculate posterior distribution P(G|r) over ghost locations given a sensor reading, with Bayes' rule:

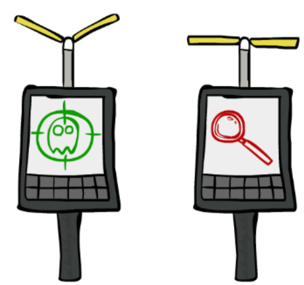
$$P(g|r) \propto P(r|g)P(g)$$

Example: Ghostbusters HMM

- $P(X_1)$ = uniform
- $P(X|X')$ = usually move clockwise, but sometimes move in a random direction or stay in place
- $P(R_{ij}|X)$ = same sensor model as before: red means close, green means far away.







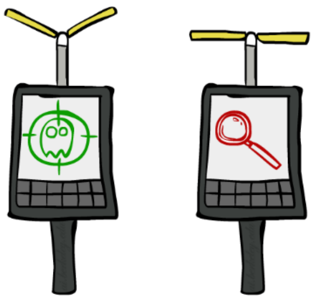
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

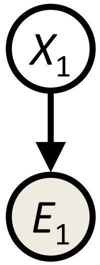
$P(X_1)$

1/6	1/6	1/2
0	1/6	0
0	0	0

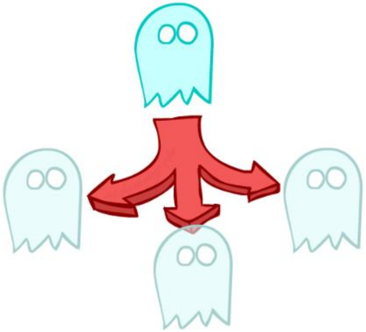
$P(X|X'=<1,2>)$

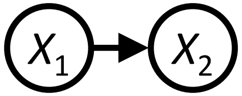
Ghost Localization: Base Cases (time=1)





+time





$$P(X_1|e_1)$$

$$P(x_1|e_1) = P(x_1, e_1)/P(e_1)$$
$$\propto_{X_1} P(x_1, e_1)$$
$$= P(x_1)P(e_1|x_1)$$

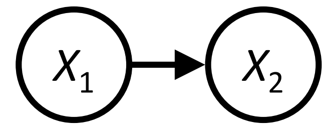
$$P(X_2)$$

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

Ghost Localization: Time = 2

- Assume we have current belief P(X | evidence before now)

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



- Then, after one time step passes:

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

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

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

- Or compactly:

$$B'(X_{t+1}) = \sum_{x_t} P(X'|x_t)B(x_t)$$

- Basic idea: beliefs get "pushed" through the transitions
  - With the "B" notation, we have to be careful about what time step t the belief is about, and what evidence it includes

Example: Passage of Time

- As time passes, uncertainty “accumulates”

(Transition model: ghosts usually go clockwise)

<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	1.00	<0.01	<0.01	<0.01
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01

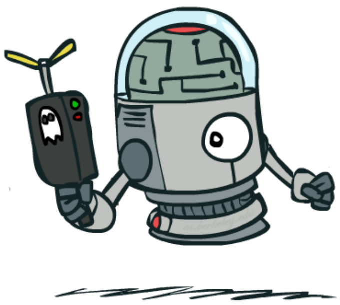
T = 1

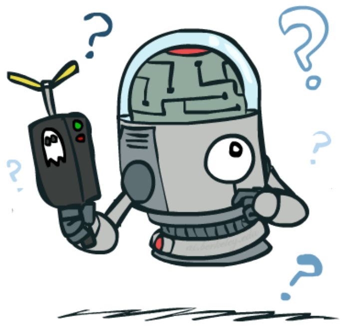
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01
<0.01	0.76	0.06	0.06	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01

T = 2

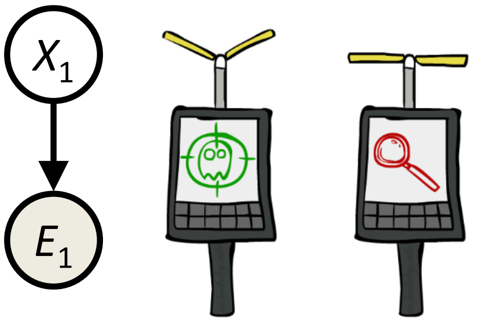
0.05	0.01	0.05	<0.01	<0.01	<0.01
0.02	0.14	0.11	0.35	<0.01	<0.01
0.07	0.03	0.05	<0.01	0.03	<0.01
0.03	0.03	<0.01	<0.01	<0.01	<0.01

T = 5





Observation



- Assume we have current belief P(X | previous evidence):

$$B'(X_{t+1}) = P(X_{t+1}|e_{1:t})$$

- Then, after evidence comes in:

$$\begin{aligned} P(X_{t+1}|e_{1:t+1}) &= P(X_{t+1}, e_{t+1}|e_{1:t})/P(e_{t+1}|e_{1:t}) \\ &\propto_{X_{t+1}} P(X_{t+1}, e_{t+1}|e_{1:t}) \\ &= P(e_{t+1}|e_{1:t}, X_{t+1})P(X_{t+1}|e_{1:t}) \\ &= P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t}) \end{aligned}$$

- Or, compactly:

$$B(X_{t+1}) \propto_{X_{t+1}} P(e_{t+1}|X_{t+1})B'(X_{t+1})$$

- Basic idea: beliefs "reweighted" by likelihood of evidence
- Unlike passage of time, we have to renormalize

Example: Observation

- As we get observations, beliefs get reweighted, uncertainty "decreases"

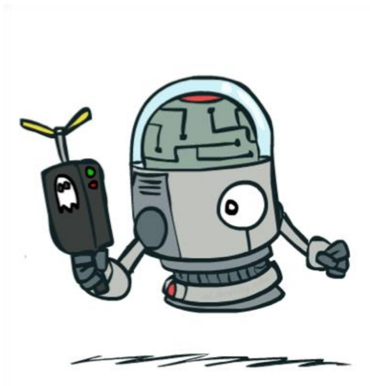
0.05	0.01	0.05	<0.01	<0.01	<0.01
0.02	0.14	0.11	0.35	<0.01	<0.01
0.07	0.03	0.05	<0.01	0.03	<0.01
0.03	0.03	<0.01	<0.01	<0.01	<0.01

Before observation

<0.01	<0.01	<0.01	<0.01	0.02	<0.01
<0.01	<0.01	<0.01	0.83	0.02	<0.01
<0.01	<0.01	0.11	<0.01	<0.01	<0.01
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01

After observation

$$B(X) \propto P(e|X)B'(X)$$



HMM Inference: Forward Algorithm

- We are given evidence at each time and want to know

$$B_t(X) = P(X_t|e_{1:t})$$

- We can derive the following updates

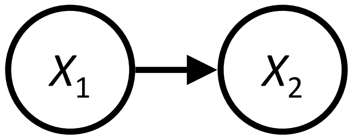
$$
\begin{aligned}
P(x_t|e_{1:t}) &\propto_X P(x_t, e_{1:t}) \\
&= \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t}) \\
&= \sum_{x_{t-1}} P(x_{t-1}, e_{1:t-1}) P(x_t|x_{t-1}) P(e_t|x_t) \\
&= \boxed{P(e_t|x_t)} \boxed{\sum_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}, e_{1:t-1})} \\
&= \boxed{P(e_t|x_t)} \sum_{x_{t-1}} P(x_t|x_{t-1}) \boxed{B'(x_{t-1})}
\end{aligned}
$$

We can normalize as we go if we want to have P(x|e) at each time step, or just once at the end…

Online Belief Updates

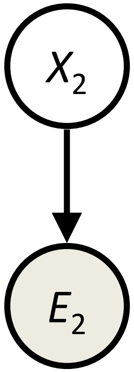
- Every time step, we start with current P(X | evidence)
- Step 1: Passage of Time (Elapse Time):

$$P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$$



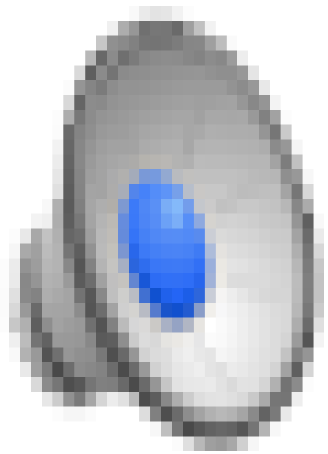
- Step 2: Observe:

$$P(x_t|e_{1:t}) \propto_X P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$



Pacman – Sonar (P4)

Ghost Pacman – Sonar (with beliefs)



Example: Passage of Time

- As time passes, uncertainty "accumulates"

(Transition model: ghosts usually go clockwise)

<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	1.00	<0.01	<0.01	<0.01
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01

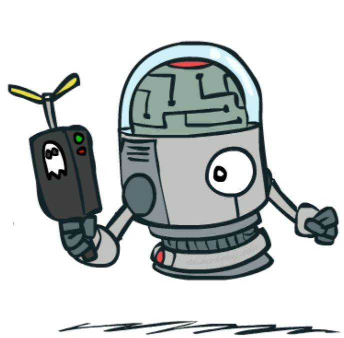
T = 1

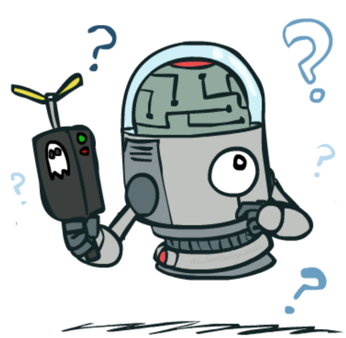
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01
<0.01	0.76	0.06	0.06	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01

T = 2

0.05	0.01	0.05	<0.01	<0.01	<0.01
0.02	0.14	0.11	0.35	<0.01	<0.01
0.07	0.03	0.05	<0.01	0.03	<0.01
0.03	0.03	<0.01	<0.01	<0.01	<0.01

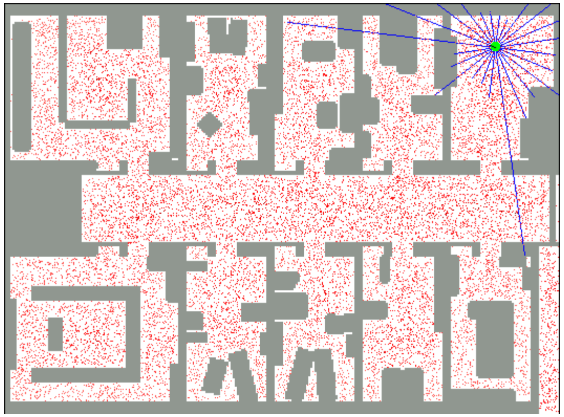
T = 5

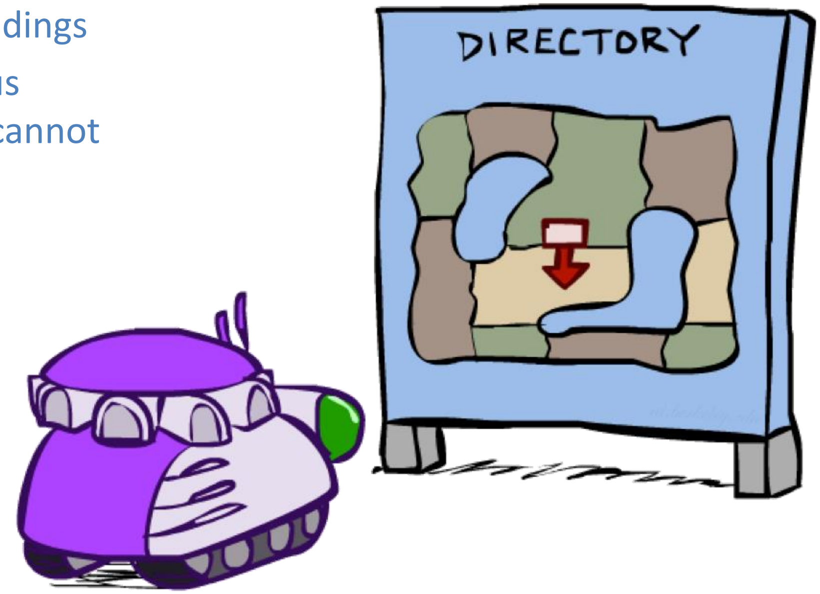




Robot Localization

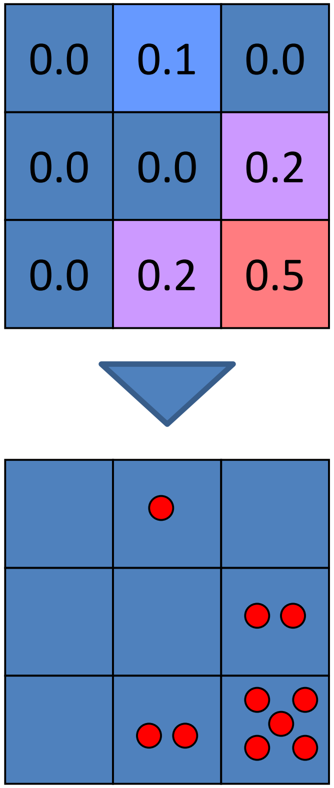
- In robot localization:
  - We know the map, but not the robot's position
  - Observations may be vectors of range finder readings
  - State space and readings are typically continuous (works basically like a very fine grid) and so we cannot store B(X)
  - Particle filtering is a main technique





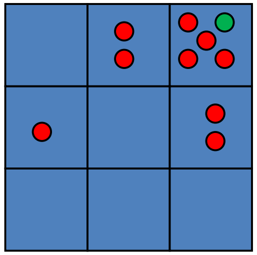
Approximate Inference: Particle Filtering

- Filtering: need approximate solution
- Sometimes |X| is too big to use exact inference
  - |X| may be too big to even store B(X)
  - E.g. X is continuous
- Solution: approximate inference
  - Track samples of X, not all values
  - Samples are called particles
  - Time per step is linear in the number of samples
  - But: number needed may be large
  - In memory: list of particles, not states
- This is how robot localization works in practice
- Particle is just new name for sample



New Representation: Particles

- Our representation of P(X) is now a list of N particles (samples)
  - Generally, N << |X|
  - Storing map from X to counts would defeat the point
- P(x) approximated by number of particles with value x
  - So, many x may have P(x) = 0!
  - More particles, more accuracy
- For now, all particles have a weight of 1



Particles:
(3,3)
(2,3)
(3,3)
(3,2)
(3,3)
(3,2)
(1,2)
(3,3)
(3,3)
(2,3)

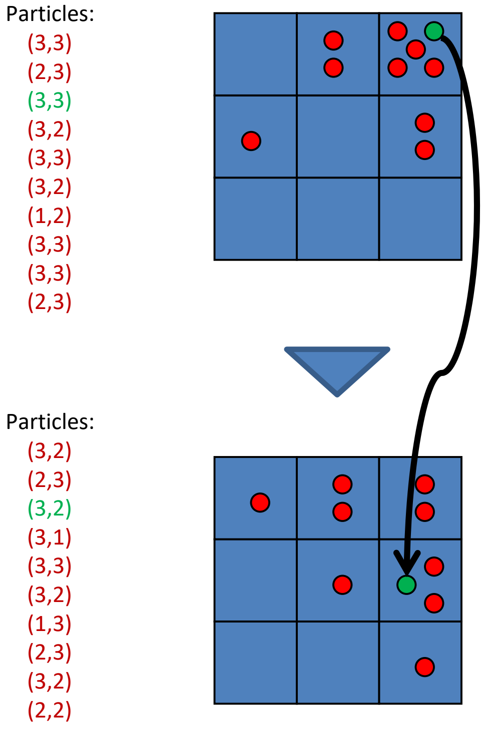
Particle Filtering 1: Elapse Time

- Each particle is moved by sampling its next position from the transition model

$$x' = \text{sample}(P(X'|x))$$

  - This is like prior sampling – samples' frequencies reflect the transition probabilities
  - Here, most samples move clockwise, but some move in another direction or stay in place

- This captures the passage of time
  - If enough samples, close to exact values before and after (consistent)



Particle Filtering 2: Observe

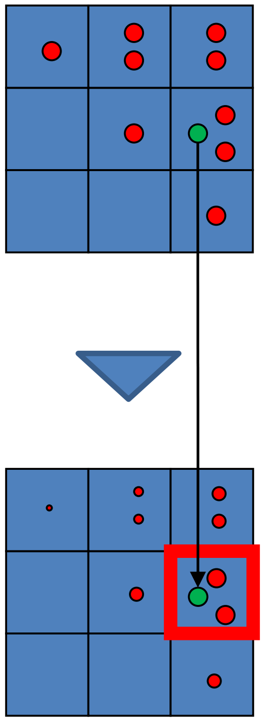
- Slightly trickier:
  - Don't sample observation, fix it
  - Similar to likelihood weighting, downweight samples based on the evidence

$$w(x) = P(e|x)$$

$$B(X) \propto P(e|X)B'(X)$$

  - Note: probabilities don't sum to 1, since all have been downweighted (in fact they now sum to (N times) an approximation of P(e))

Particles:
(3,2)
(2,3)
(3,2)
(3,1)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(2,2)



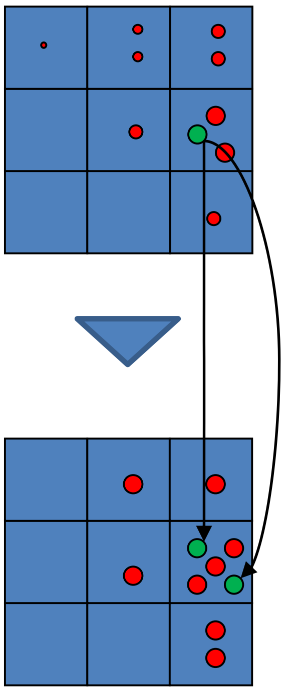
Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4

Particle Filtering 3: Resample

- Instead of tracking weighted samples, we resample
- N times, we choose 1 particle from our weighted sample distribution (i.e. draw with replacement)
- This is equivalent to renormalizing the distribution
- Now the update is complete for this time step, continue with the next one

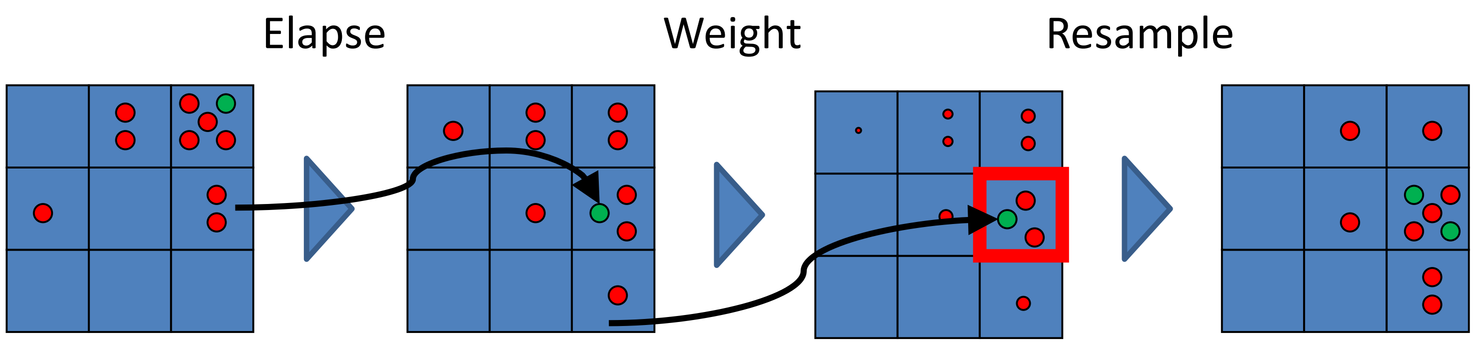
Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4

(New) Particles:
(3,2)
(2,2)
(3,2)
(2,3)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(3,2)



Summary: Particle Filtering

- Particles: track samples of states rather than an explicit distribution



Particles:	Particles:	Particles:	(New) Particles:
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,3)	(2,3) w=.2	(2,2)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(3,2)	(3,1)	(3,1) w=.4	(2,3)
(3,3)	(3,3)	(3,3) w=.4	(3,3)
(3,2)	(3,2)	(3,2) w=.9	(3,2)
(1,2)	(1,3)	(1,3) w=.1	(1,3)
(3,3)	(2,3)	(2,3) w=.2	(2,3)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,2)	(2,2) w=.4	(3,2)

Robot Localization (Sonar)

Project 3: Ghostbusters!

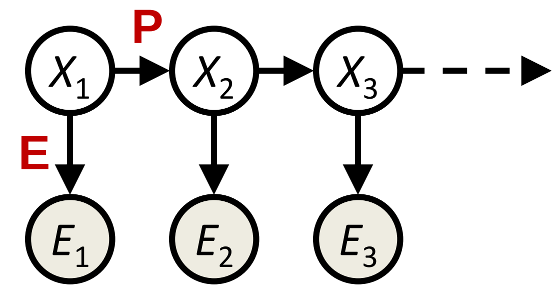
- Due **Wed March 21st**
- www.mathcs.emory.edu/~eugene/cs425/p3/

  - **Midterm review (optional): 5:30pm, W301**

  - **Midterm Exam (March 8th)**
    - **Closed book, notes (1 sheet of notes allowed)**

Remaining HMM Issues (after break)

- Where do **P**, **E** probabilities come from?
  - Training HMMs
  - Dealing with unobserved values



- What's the most likely explanation (a.k.a. decoding)
  - Needed for speech recognition, language parsing, …
  - Solution: Viterbi algoritm (most likely explanation)
  - What is we want multiple explanations? (beam search)

Techniques so far (on Exam)

- Search
  - Uninformed (UCS)
  - Informed (A*)
  - Scalability: ID, RM
- Adversarial Search
  - MiniMax
  - Alpha-Beta Pruning
  - ExpectiMax
- Uncertainty:
  - Bayes Nets, Inference
  - Markov Chains, Hidden Markov Models (filtering/prediction)