Announcements

- HW 8 is out, due **tonight** at 11:59 pm
- Project 4 due 4/12, this **Friday**

CS 188: Artificial Intelligence

Hidden Markov Models

Instructor: Sergey Levine and Stuart Russell --- University of California, Berkeley

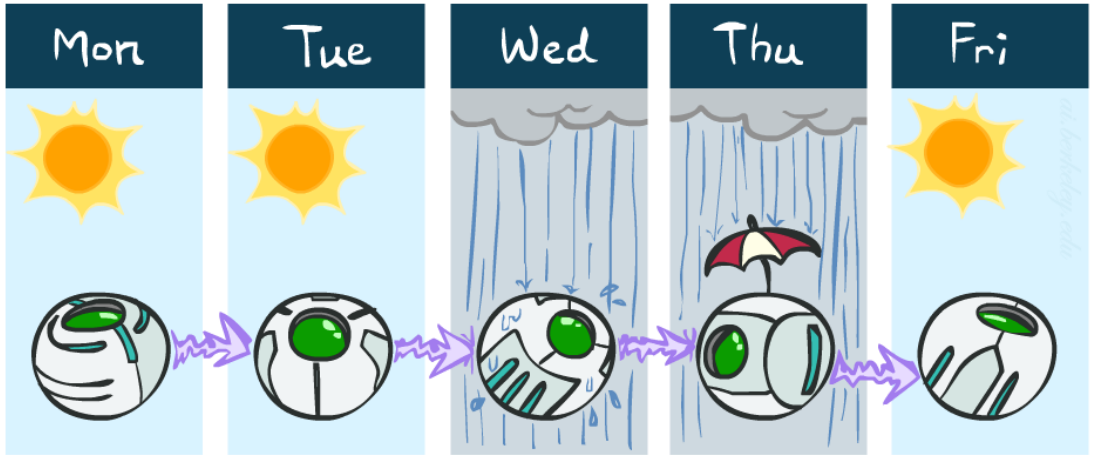
[These slides were created by Dan Klein, Pieter Abbeel, and Sergey Levine. http://ai.berkeley.edu.]

Markov Models

- Basic conditional independence:
  - Past and future independent given the present
  - Each time step only depends on the previous
  - This is called the (first order) Markov property

Example Markov Chain: Weather

- States: X = {rain, sun}

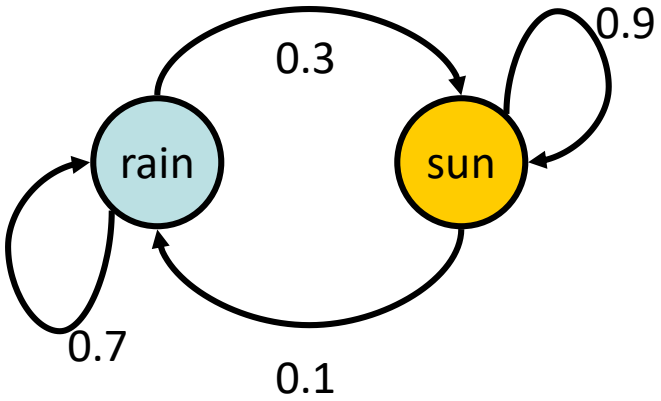


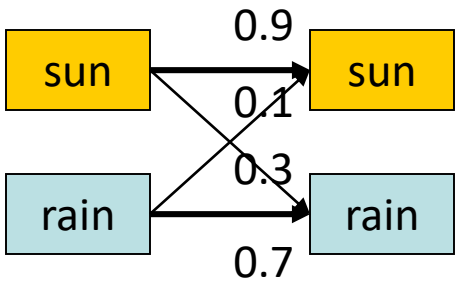
- Initial distribution: 0.5 sun

- 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

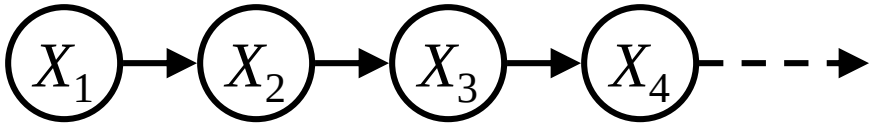
Two new ways of representing the same CPT

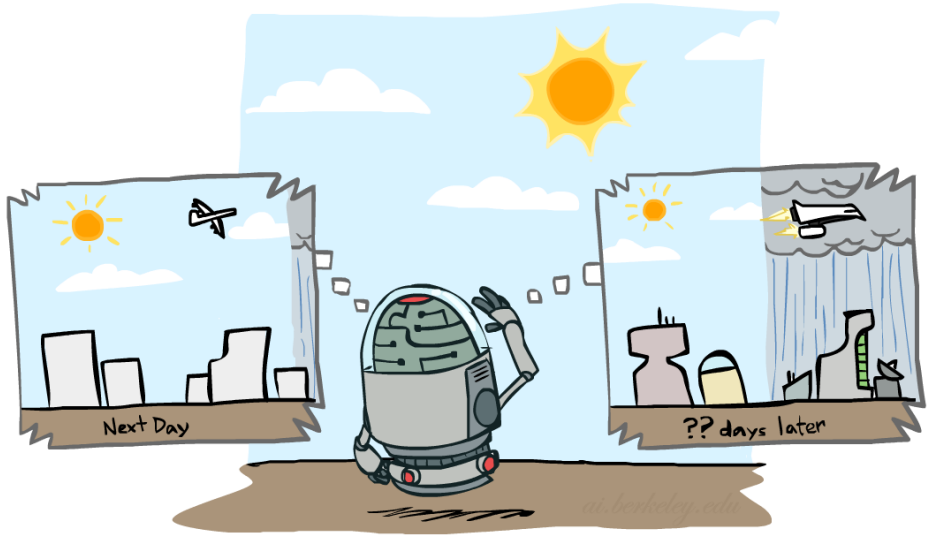




Forward Algorithm

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





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

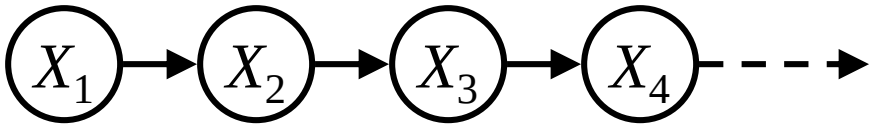
$$P(x_t) = \sum_{x_{t-1}} P(x_{t-1}, x_t)$$

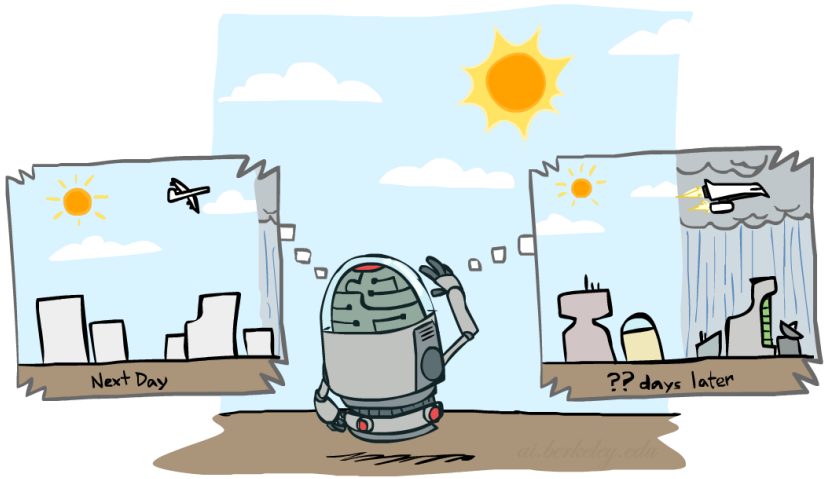
$$= \sum_{x_{t-1}} P(x_t \mid x_{t-1}) P(x_{t-1})$$

Forward simulation

Stationary Distributions

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





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

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

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

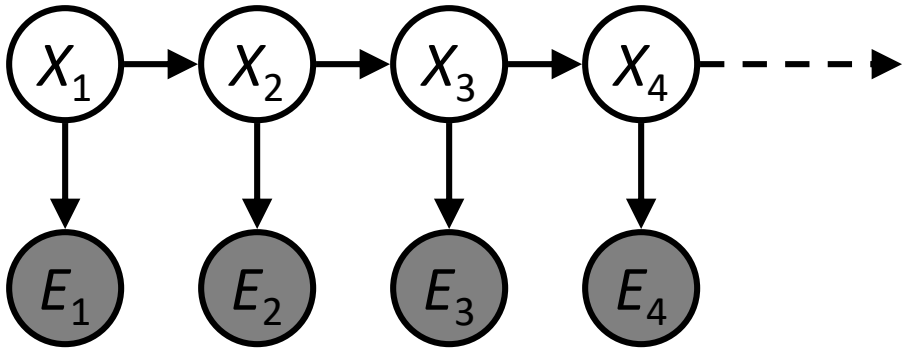
Also: $P_\infty(sun) + P_\infty(rain) = 1$

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

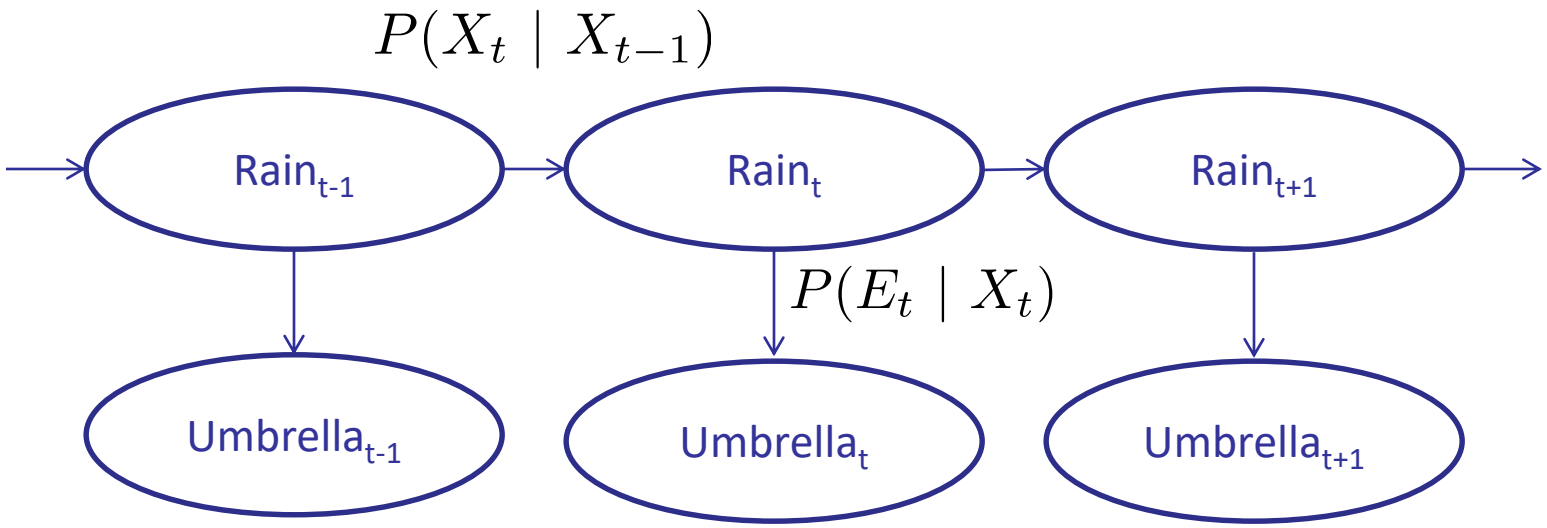
X_{t-1}	X_t	$P(X_t \mid X_{t-1})$
sun	sun	0.9
sun	rain	0.1
rain	sun	0.3
rain	rain	0.7

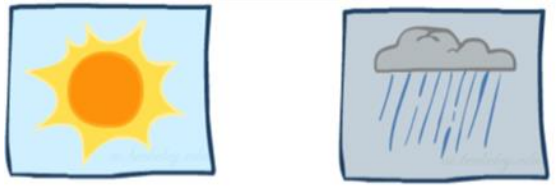
Hidden Markov Models

- 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: $P(E_t \mid X_t)$

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

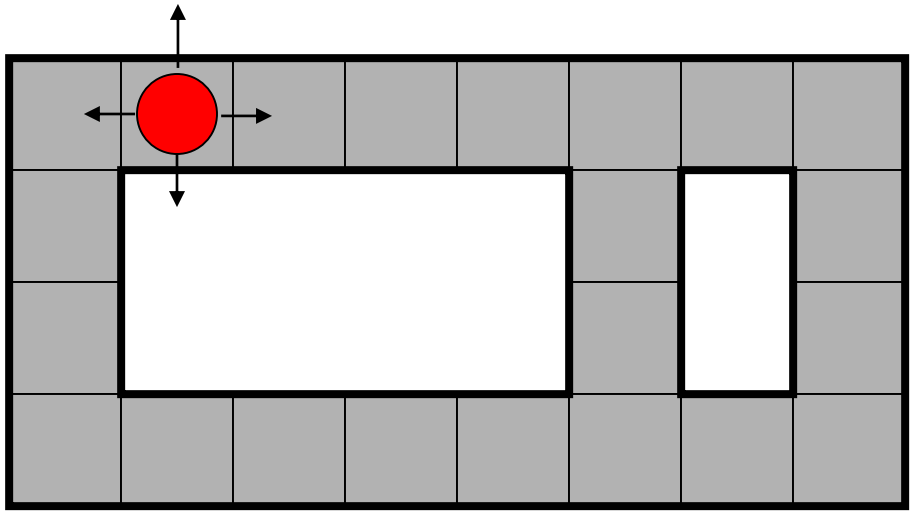
R_t	U_t	$P(U_t \mid R_t)$
+r	+u	0.9
+r	-u	0.1
-r	+u	0.2
-r	-u	0.8

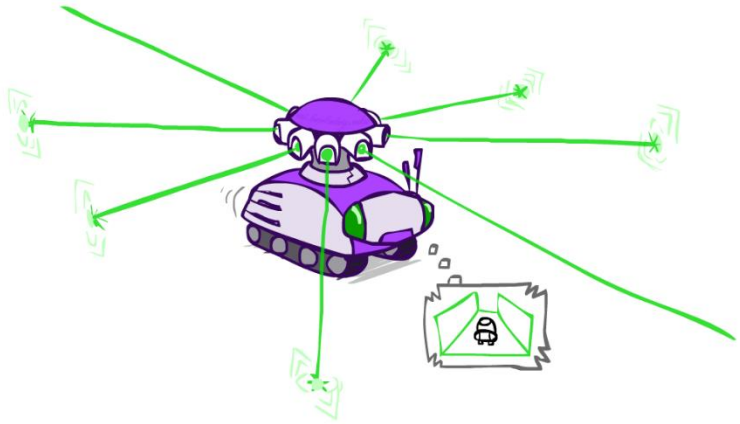
Inference tasks

- ***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

Example: Robot Localization

Example from
Michael Pfeiffer





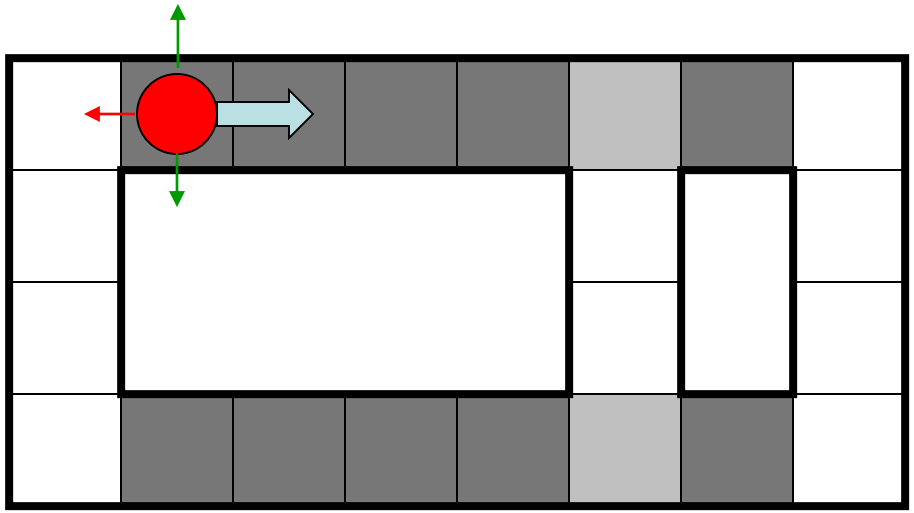
Prob 0 1

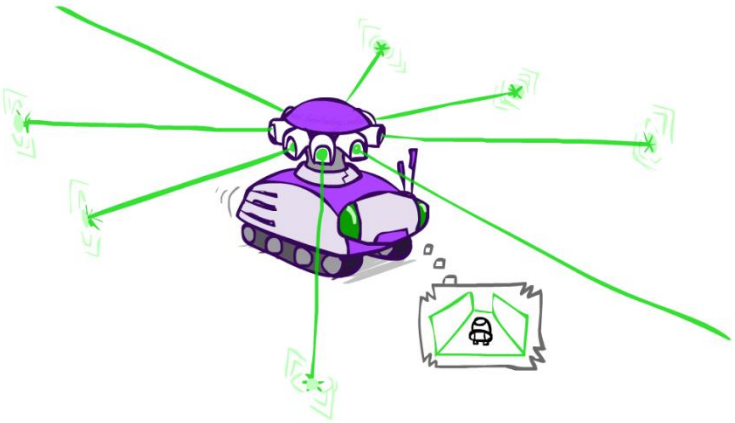
t=0

Sensor model: four bits for wall/no-wall in each direction, never more than 1 mistake

Transition model: action may fail with small prob.

Example: Robot Localization



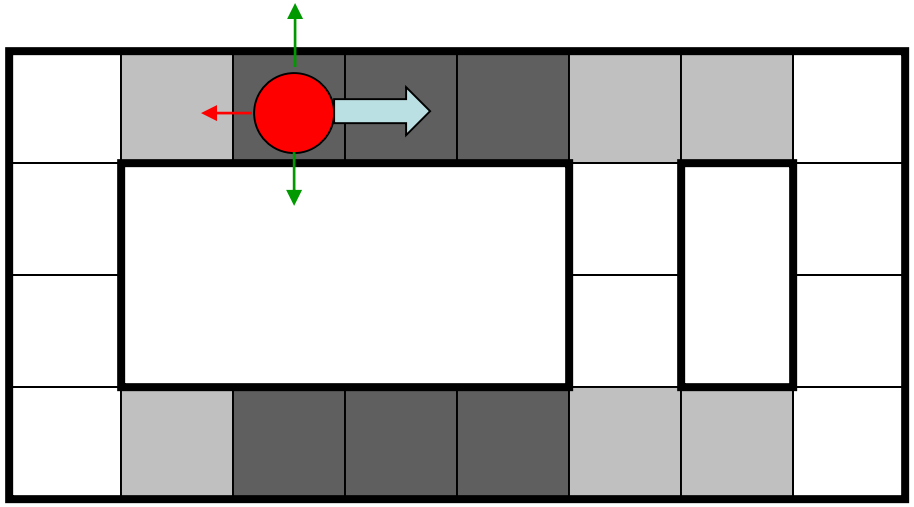


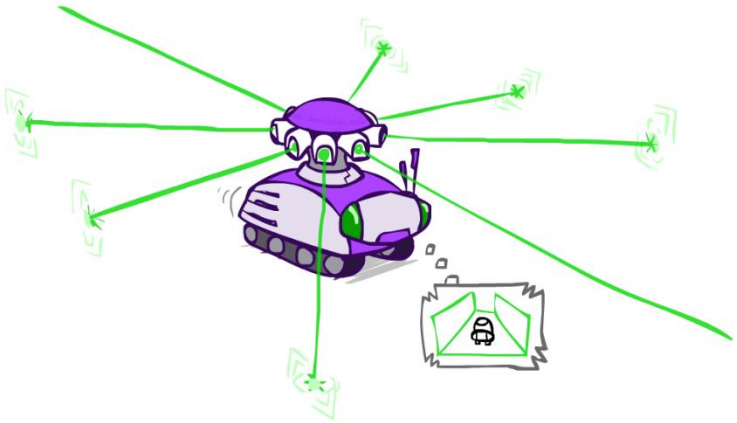
t=1

Lighter grey: was ***possible*** to get the reading,

but ***less likely*** (required 1 mistake)

Example: Robot Localization

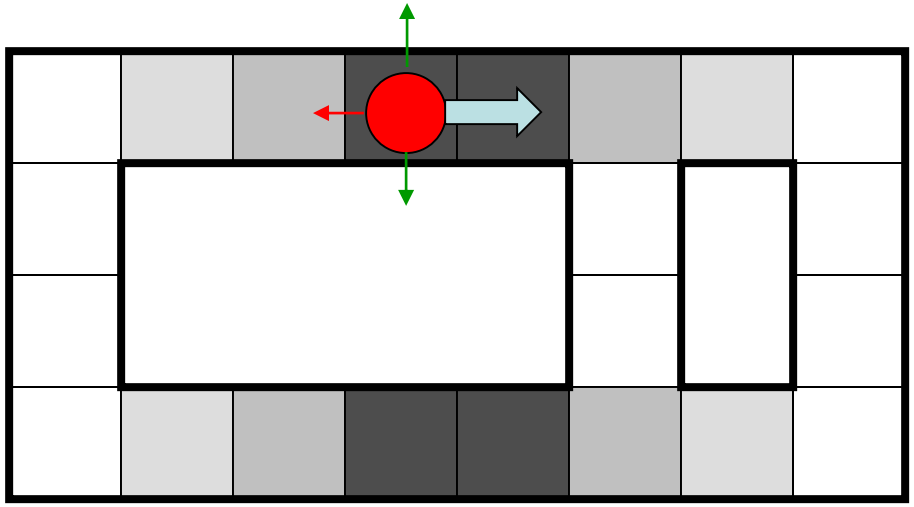




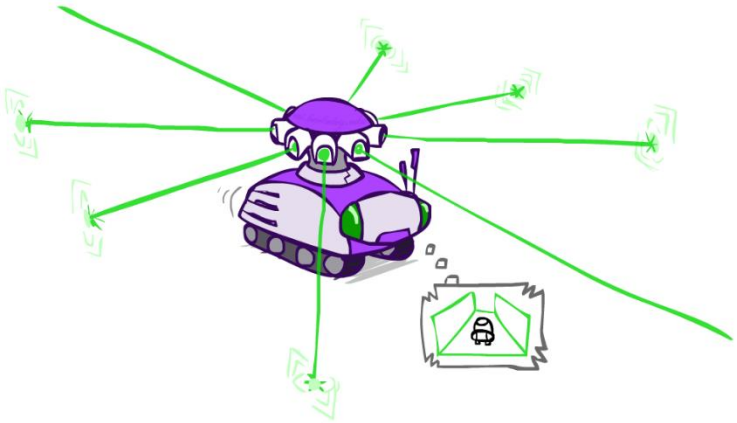
Prob 0 1

t=2

Example: Robot Localization

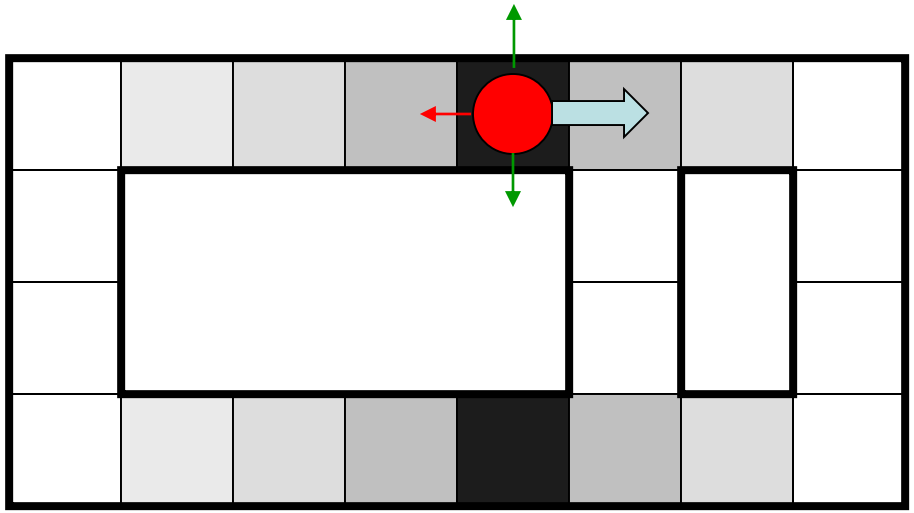


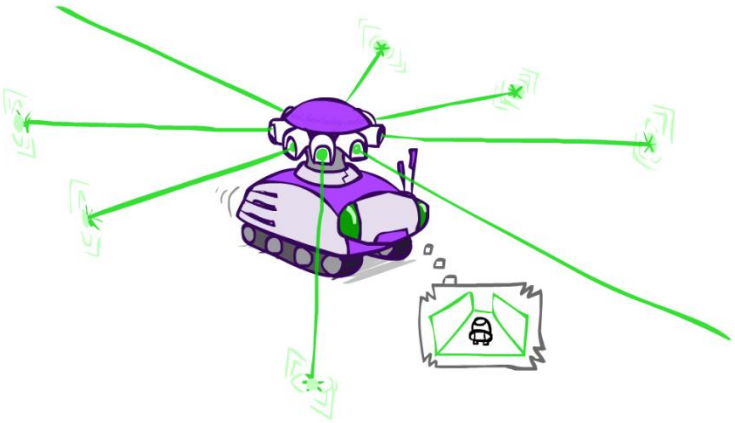
Prob 0 1



t=3

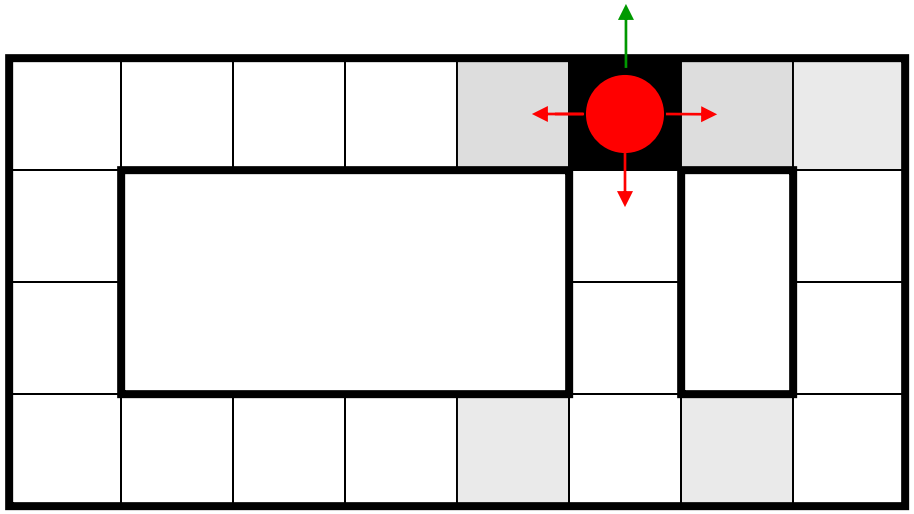
Example: Robot Localization

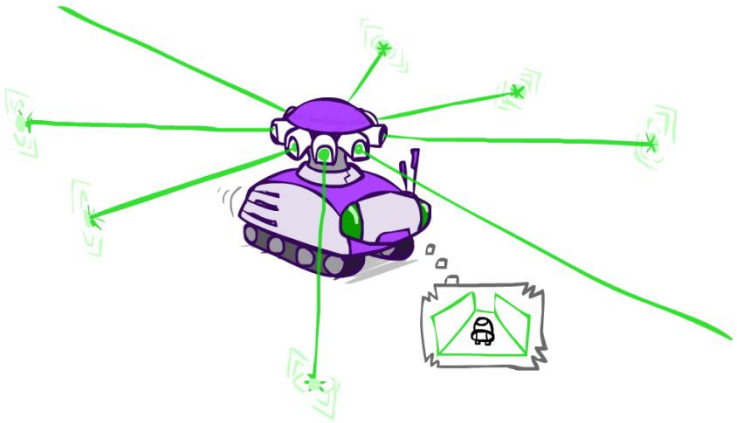




t=4

Example: Robot Localization





Prob 0 1

t=5

Filtering: Find State Given *Past* Evidence

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

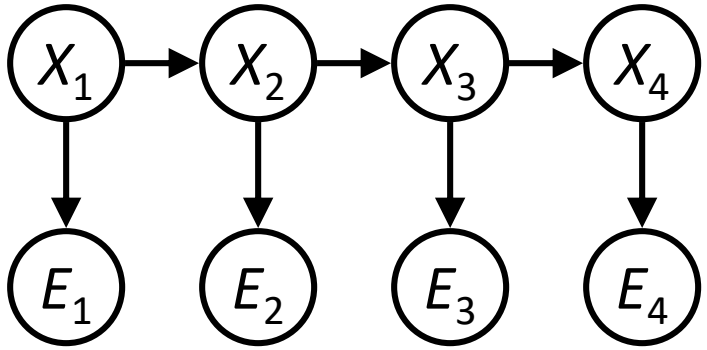
$$f_{1:t}(X) = P(X_t|e_{1:t})$$

- Idea: start with $P(X_1)$ and derive $f_{1:t}$ in terms of $f_{1:t-1}$
  - equivalently, derive $f_{1:t+1}$ in terms of $f_{1:t}$

$$f_{1:t}(X) = P(X_t|e_{1:t}) \xrightarrow{\text{using } e_{t+1}} f_{1:t+1}(X) = P(X_{t+1}|e_{1:t+1})$$

Two Steps: Passage of Time + Observation

$f_{1:t}(X) = P(X_t|e_{1:t})$ $\quad$ $f'_{1:t+1}(X) = P(X_{t+1}|e_{1:t})$

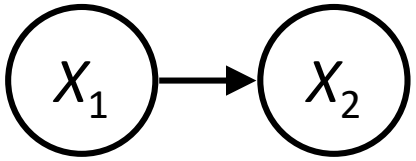


$f_{1:t+1}(X) = P(X_{t+1}|e_{1:t+1})$

Passage of Time

- Assume we have current belief

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



- 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:

$$f'_{1:t+1}(X_{t+1}) = \sum_{x_t} P(X'|x_t)f_{1:t}(x_t)$$

- Basic idea: beliefs get "pushed" through the transitions

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

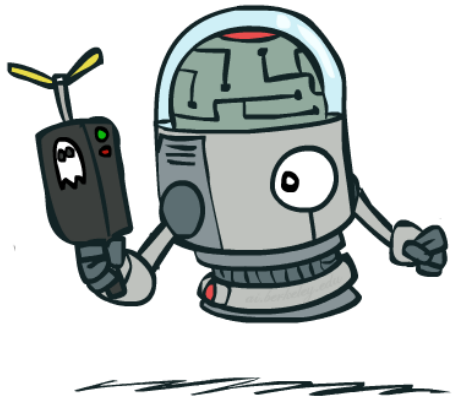
T = 1

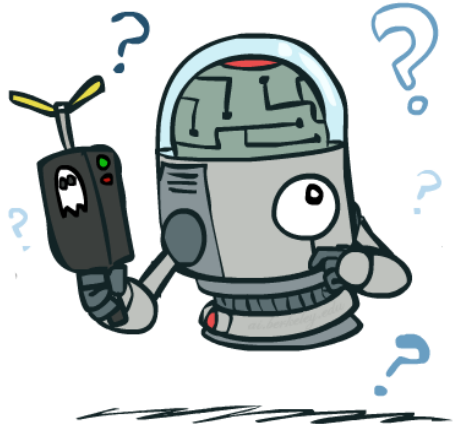
<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01
<0.01	0.76	0.06	0.06	<0.01	<0.01
<0.01	<0.01	0.06	<0.01	<0.01	<0.01

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):

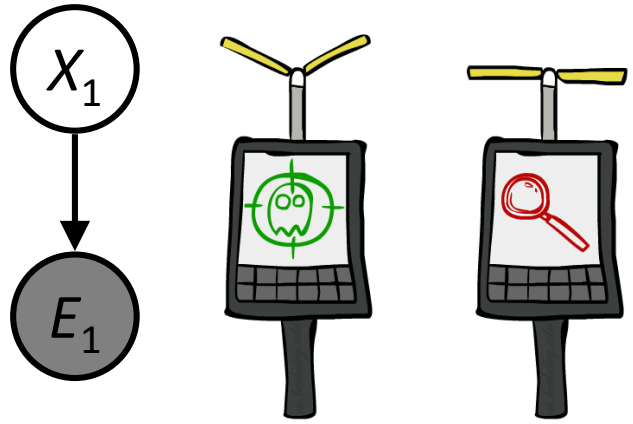
$$f'_{1:t+1}(X) = P(X_{t+1}|e_{1:t})$$

- Then, after evidence comes in:

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

- Or, compactly: $f'_{1:t+1}(X_{t+1}) = \sum_{x_t} P(X'|x_t) f_{1:t}(x_t)$

$$f_{1:t+1}(X_{t+1}) \propto_{X_{t+1}} P(e_{t+1}|X_{t+1}) f'_{1:t+1}(X_{t+1})$$



- 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"

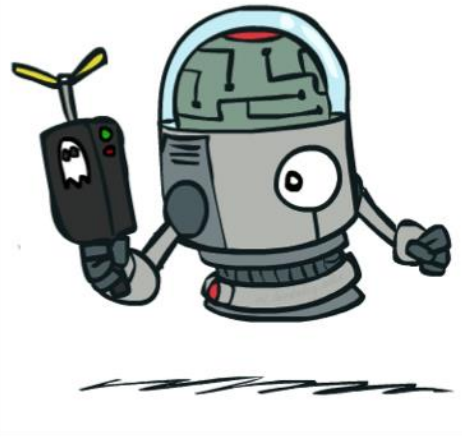
0.05	0.01	0.05	<0.01	<0.01	<0.01
0.02	0.14	0.11	0.35	<0.01	<0.01
0.07	0.03	0.05	<0.01	0.03	<0.01
0.03	0.03	<0.01	<0.01	<0.01	<0.01

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

$$f_{1:t+1}(X_{t+1}) \propto_{X_{t+1}} P(e_{t+1}|X_{t+1}) f'_{1:t+1}(X_{t+1})$$



HMM Filtering Algorithm

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

$$f_{1:t}(X) = P(X_t|e_{1:t}) \propto_{X_t} P(X_t, e_{1:t})$$

- We can derive the following updates

$$P(x_t|e_{1:t}) \propto_X P(x_t, e_{1:t})$$

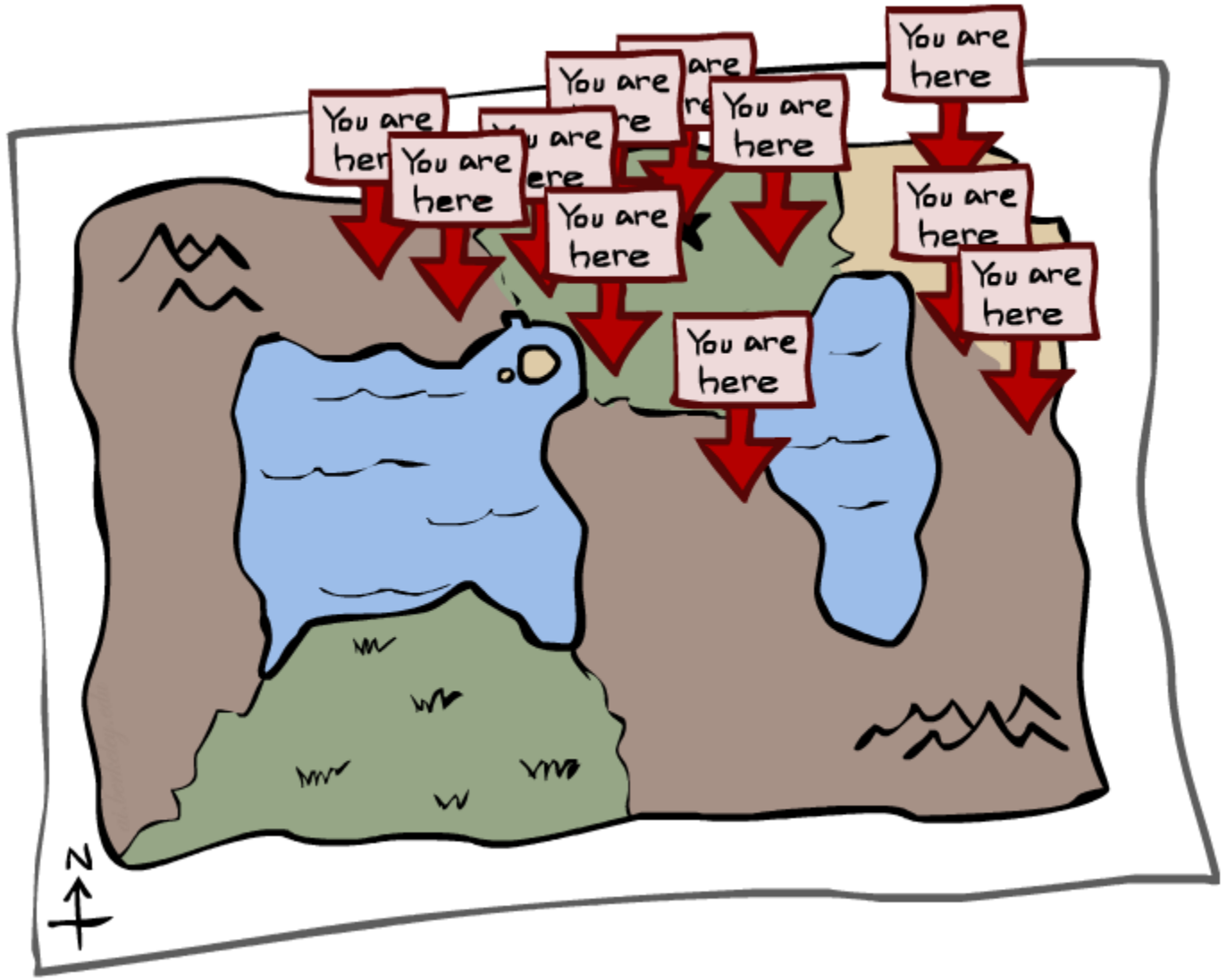
$$= \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t})$$

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

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

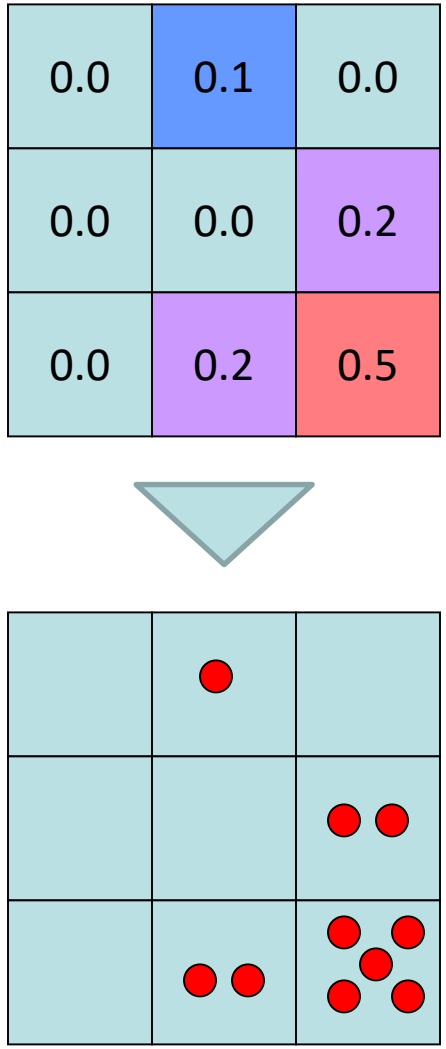
We can normalize as we go if we want to have P(x|e) at each time step, or just once at the end…

Particle Filtering



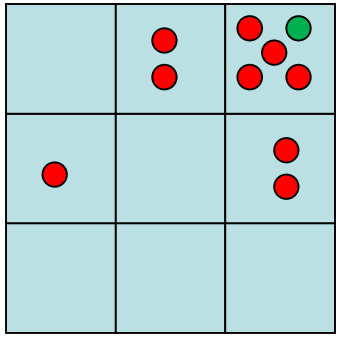
Particle Filtering

- Filtering: approximate solution
- Sometimes |X| is too big to use exact inference
  - |X| may be too big to even store f(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



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)

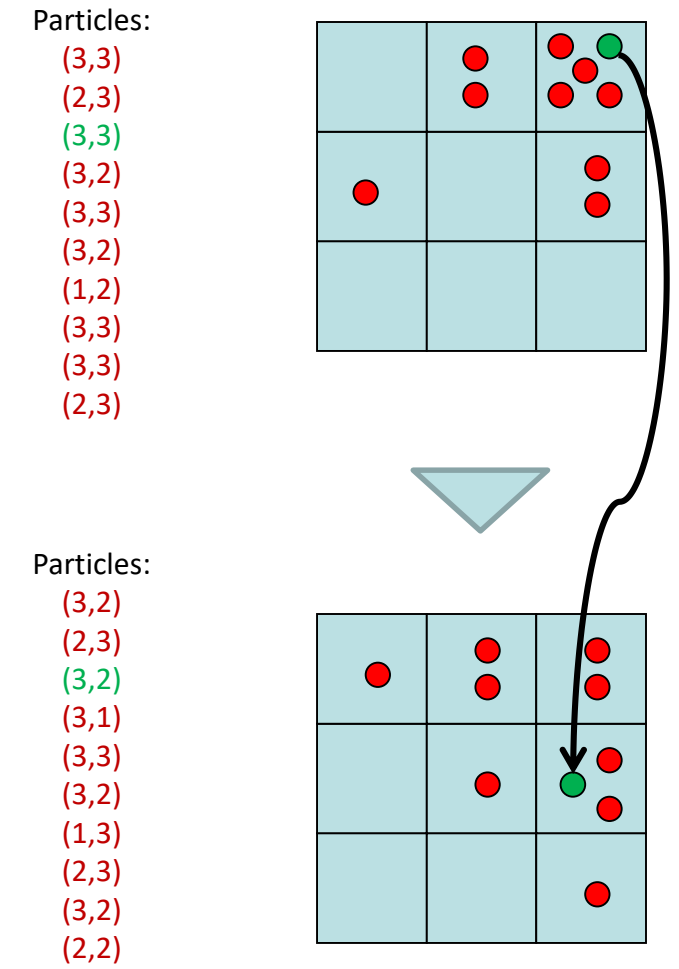
Particle Filtering: Elapse Time

- 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: Observe

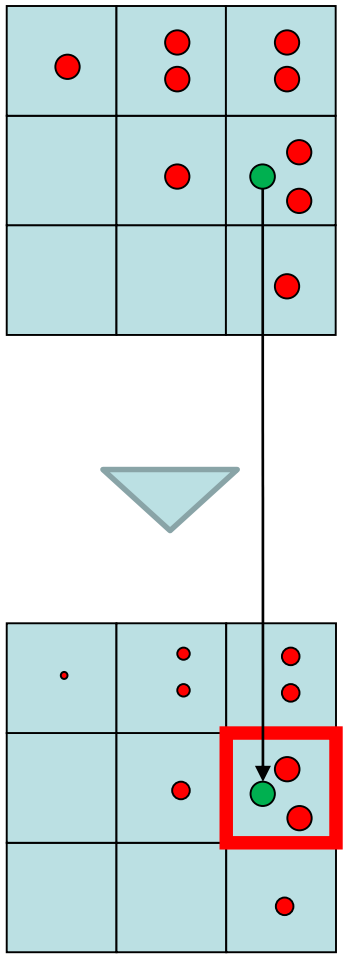
- Slightly trickier:
  - Don't sample observation, fix it
  - Similar to likelihood weighting, downweight samples based on the evidence

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

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

  - As before, the probabilities don't sum to one, 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)



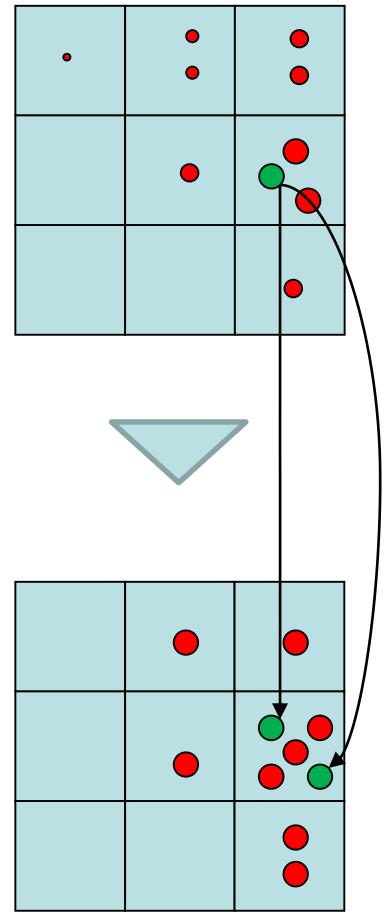
Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4

Particle Filtering: Resample

- Rather than tracking weighted samples, we resample
- N times, we choose 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

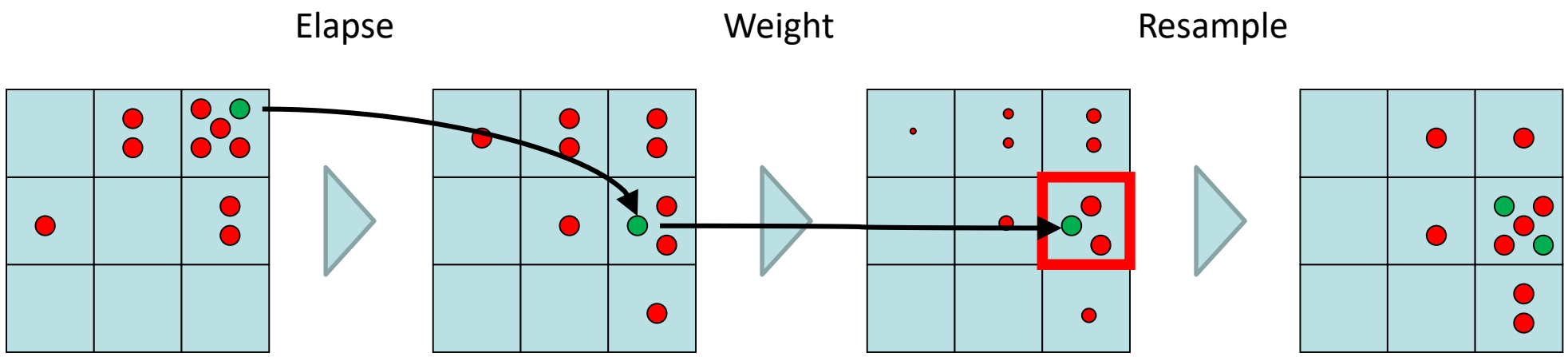
Particles:
(3,2) w=.9
(2,3) w=.2
(3,2) w=.9
(3,1) w=.4
(3,3) w=.4
(3,2) w=.9
(1,3) w=.1
(2,3) w=.2
(3,2) w=.9
(2,2) w=.4

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



Recap: Particle Filtering

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



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

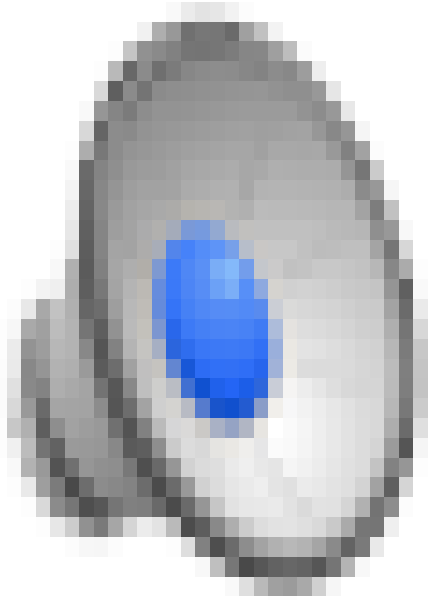
Particles:
(3,2)
(2,3)
(3,2)
(3,1)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(2,2)

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

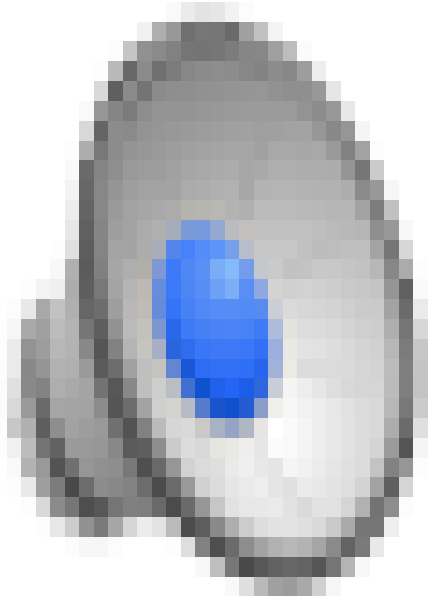
(New) Particles:
(3,2)
(2,2)
(3,2)
(2,3)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(3,2)

[Demos: ghostbusters particle filtering (L15D3,4,5)]

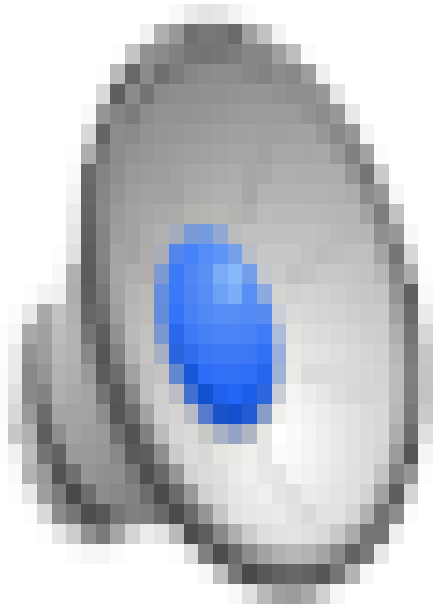
Video of Demo – Moderate Number of Particles



Video of Demo – One Particle

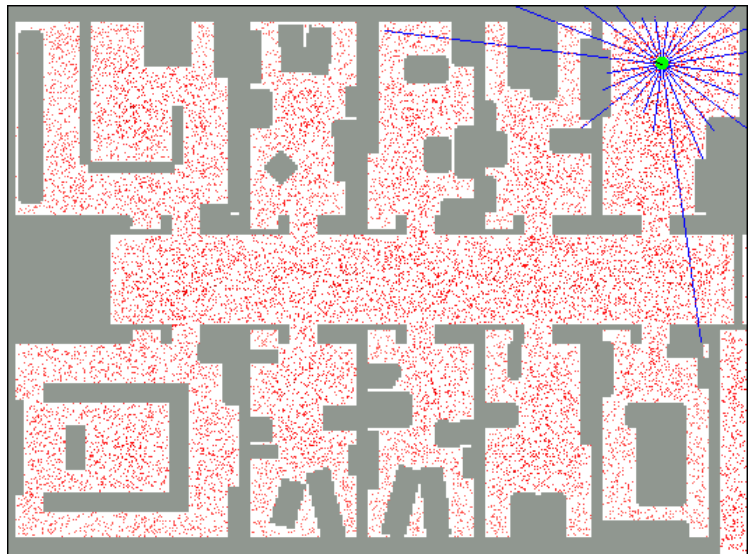


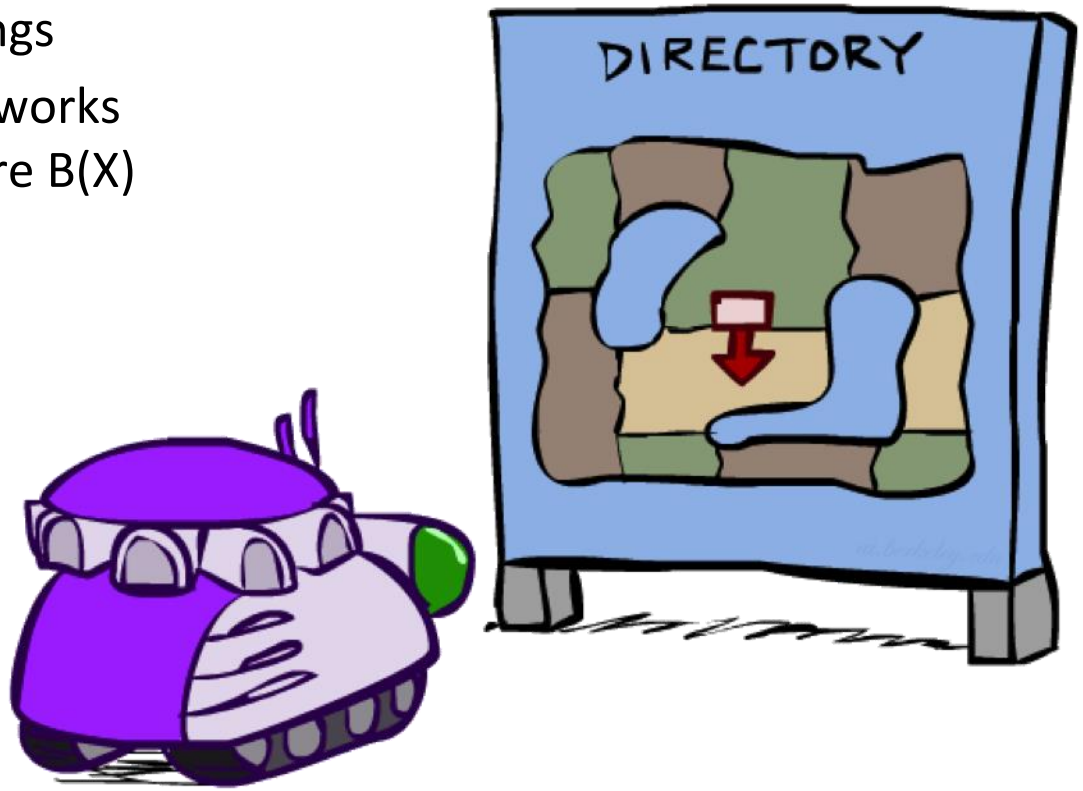
Video of Demo – Huge Number of Particles



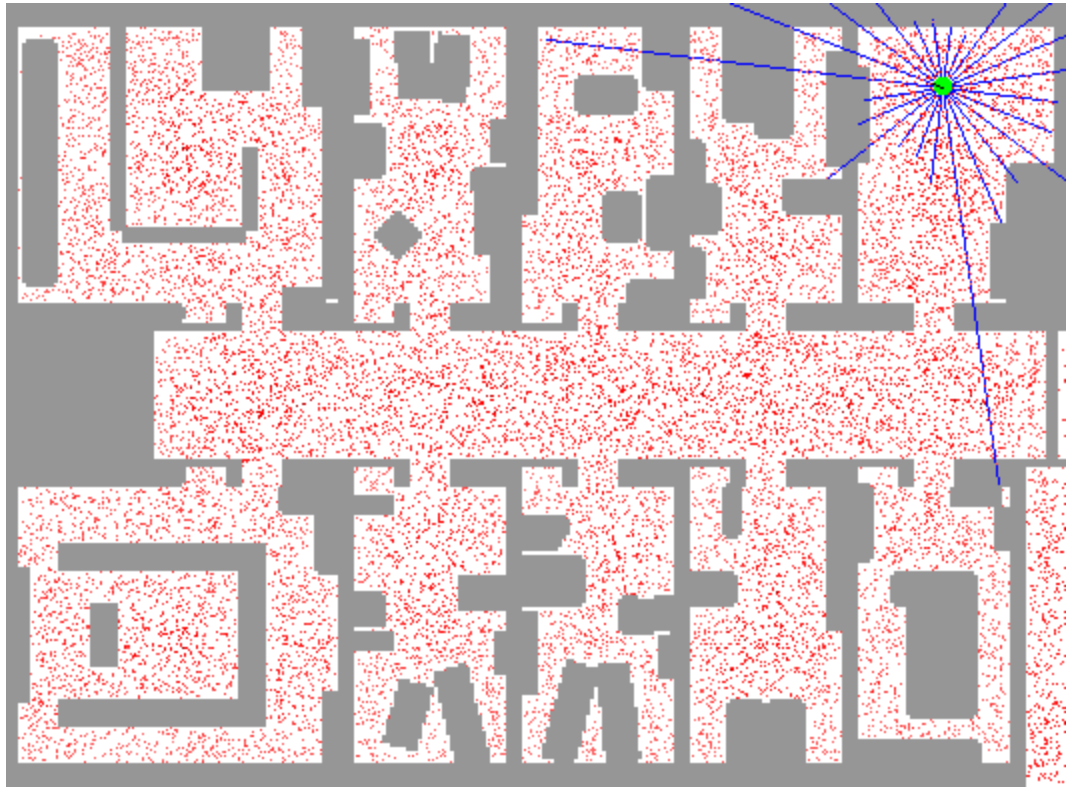
Robot Localization

- 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





Example: Robot Localization



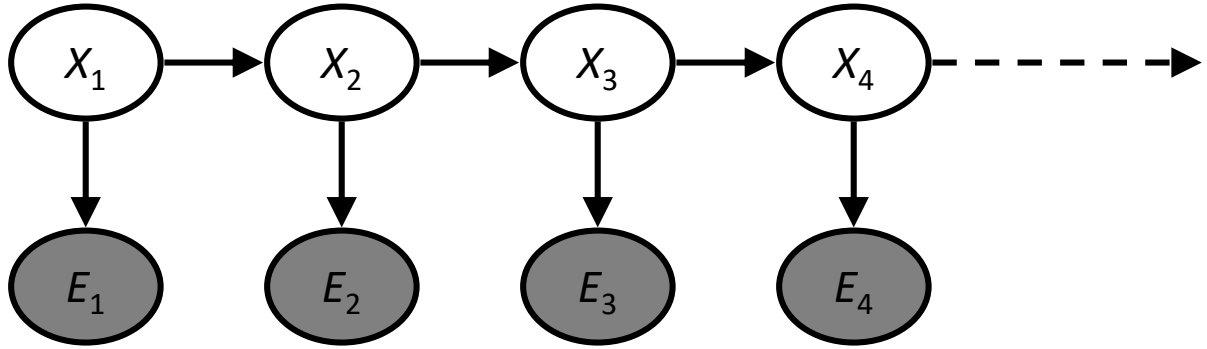
Particle Filter Localization (Sonar)

[Video: global-sonar-uw-annotated.avi]

Most Likely Explanation

HMMs: MLE Queries

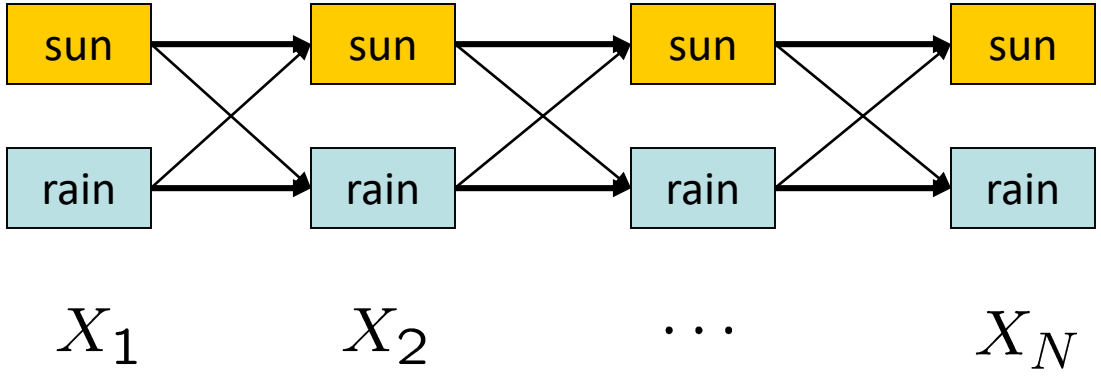
- HMMs defined by
  - States X
  - Observations E
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X|X_{-1})$
  - Emissions: $P(E|X)$



- New query: most likely explanation: $\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t})$
- New method: the Viterbi algorithm

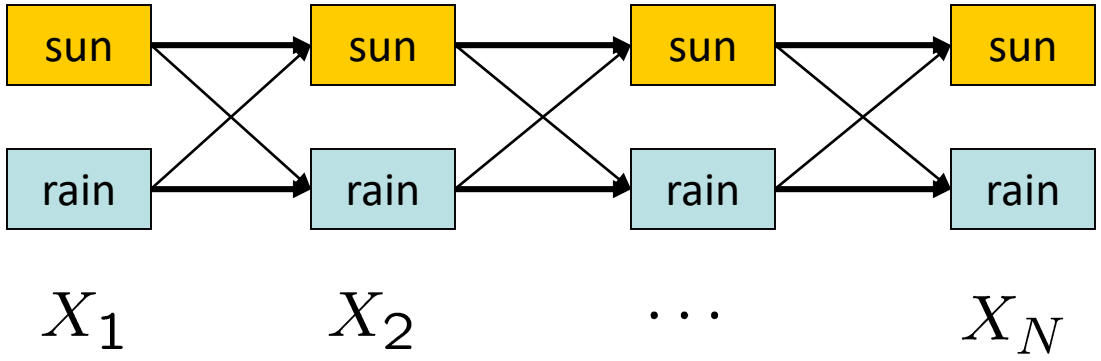
State Trellis

- State trellis: graph of states and transitions over time



- Each arc represents some transition $x_{t-1} \rightarrow x_t$
- Each arc has weight $P(x_t|x_{t-1})P(e_t|x_t)$
- Each path is a sequence of states
- The product of weights on a path is that sequence's probability along with the evidence
- Forward algorithm computes sums of paths, Viterbi computes best paths

Finding the Most Likely Path



Forward Algorithm (Sum)

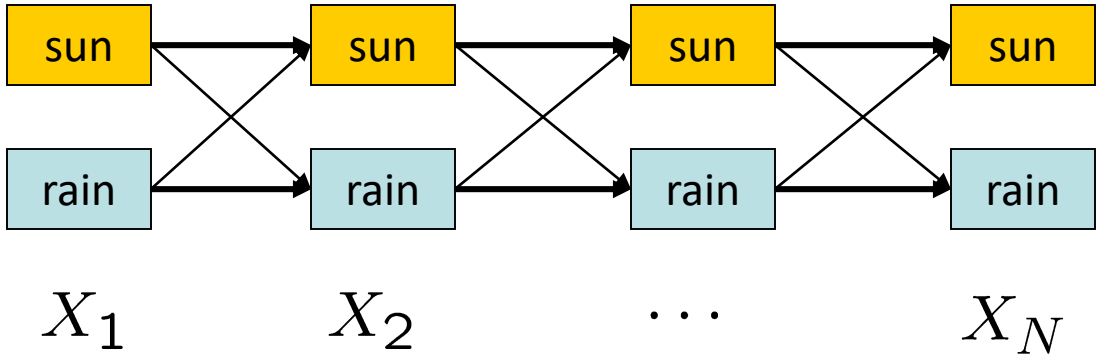
$$f_t[x_t] = P(x_t, e_{1:t})$$
$$= P(e_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) f_{t-1}[x_{t-1}]$$

Viterbi Forward Phase (Max)

$$m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t})$$
$$= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) m_{t-1}[x_{t-1}]$$

probability of best path $1:t$ that ends at x_t

Why is This True?



$$m_1[x_1] = P(e_1|x_1)P(x_1)$$

probability of best path $1:t$ that ends at x_t

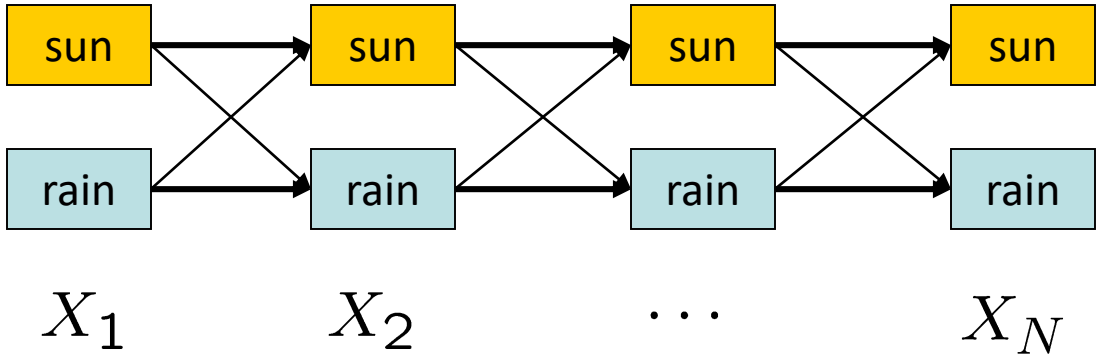
$$m_2[x_2] = \max\{P(e_2|x_2)P(x_2|x_1 = \text{rain})m_1[x_1 = \text{rain}], P(e_2|x_2)P(x_2|x_1 = \text{sun})m_1[x_1 = \text{sun}]\}$$

$$= \max_{x_1} P(e_2|x_2)P(x_2|x_1)m_1[x_1]$$

$$m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t})$$

$$= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

Now What?



$m_N[x_N]$ probability of best path $1:N$ that ends at x_N

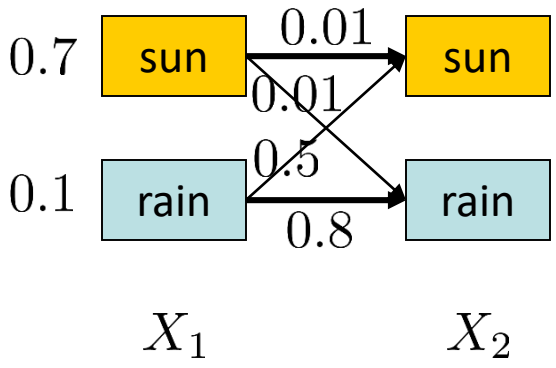
what is the last state on the most likely path?

$$\arg\max_{x_N} m_N[x_N]$$

what is the *second to last* state on the most likely path?

A Tricky Counter-Example

$P(e_1|x_1)P(x_1)$ $P(e_2|x_2)P(x_2|x_1)$



$$\arg\max_{x_1} m_1[x_1] = ? \qquad \text{sun!}$$

$$m_2[x_2] = \max_{x_1} P(e_2|x_2)P(x_2|x_1)m_1[x_1]$$

$$m_2[x_2 = \text{sun}] = \max\{0.7 \times 0.01, 0.1 \times 0.5\} = 0.05$$

$$m_2[x_2 = \text{rain}] = \max\{0.7 \times 0.01, 0.1 \times 0.8\} = 0.08$$

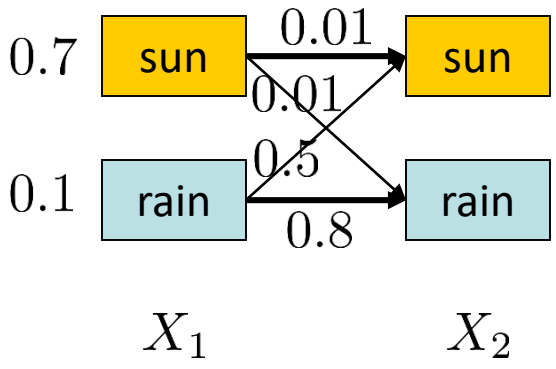
$$\arg\max_{x_2} m_2[x_2] = \text{rain}$$

$$P(x_1 = \text{sun}, x_2 = \text{rain}, e_1, e_2) = 0.7 \times 0.01 = 0.007$$

$$\text{best path } 1:t \text{ that ends at } x_t \quad \neq \quad \text{best path } 1:N \text{ that goes through } x_t$$

What do We Do?

$P(e_1|x_1)P(x_1)$ $P(e_2|x_2)P(x_2|x_1)$



$\arg\max_{x_1} m_1[x_1] = ?$ sun!

$$m_2[x_2] = \max_{x_1} P(e_2|x_2)P(x_2|x_1)m_1[x_1]$$

this path starts at rain

$$m_2[x_2 = \text{sun}] = \max\{0.7 \times 0.01, 0.1 \times 0.5\} = 0.05$$

$$m_2[x_2 = \text{rain}] = \max\{0.7 \times 0.01, 0.1 \times 0.8\} = 0.08$$

this path starts at rain

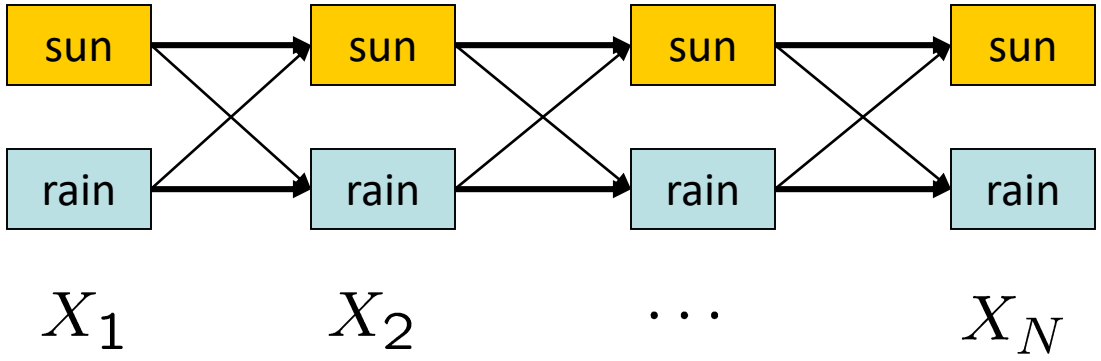
$$\arg\max_{x_2} m_2[x_2] = \text{rain}$$

idea: what if we also save *where* the best path came from?

$$m_t[x_t] = \max_{x_{t-1}} P(e_t|x_t)P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

$$a_t[x_t] = \arg\max_{x_{t-1}} P(e_t|x_t)P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

Follow the Breadcrumbs…



for $t = 1$ to N:

$$m_t[x_t] = \max_{x_{t-1}} P(e_t|x_t)P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

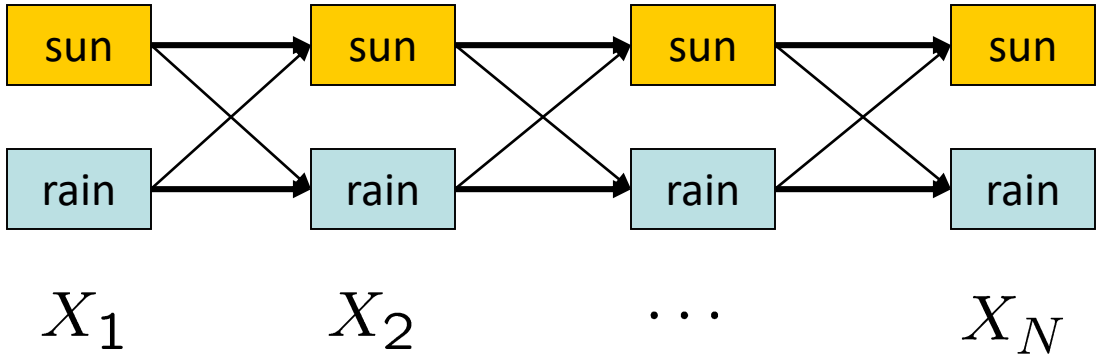
$$a_t[x_t] = \arg\max_{x_{t-1}} P(e_t|x_t)P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

last state on most likely path: $x_N^\star = \arg\max_{x_N} m_N[x_N]$

second to last state on most likely path: $x_{N-1}^\star = a_N[x_N^\star]$

third to last state on most likely path: $x_{N-2}^\star = a_{N-1}[x_{N-1}^\star]$

Follow the Breadcrumbs…



for $t = 1$ to N:

$$m_t[x_t] = \max_{x_{t-1}} P(e_t|x_t)P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

$$a_t[x_t] = \arg\max_{x_{t-1}} P(e_t|x_t)P(x_t|x_{t-1})m_{t-1}[x_{t-1}]$$

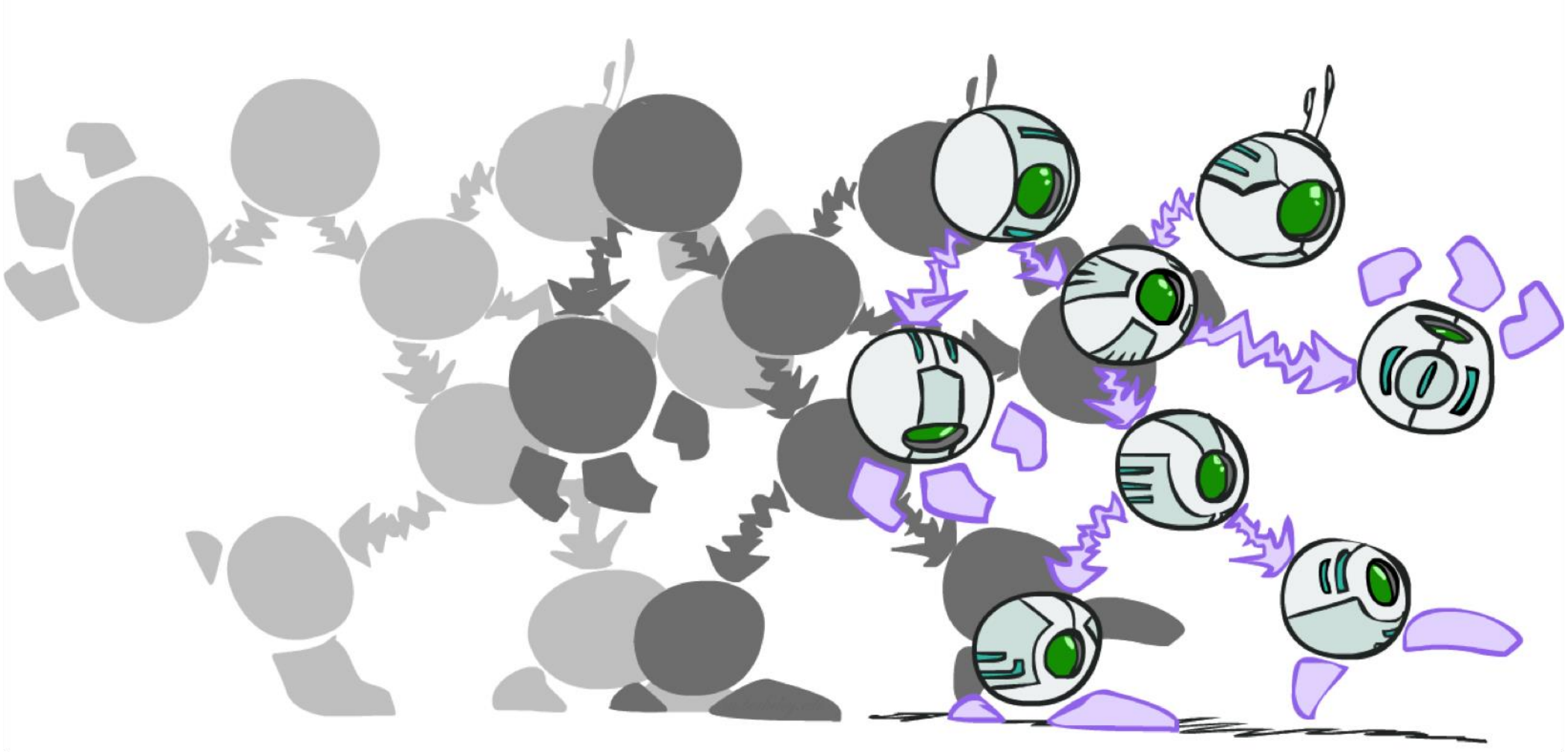
$x_N^\star = \arg\max_{x_N} m_N[x_N]$

for $t = N$ to 2:

$$x_{t-1}^\star = a_t[x_t^\star]$$

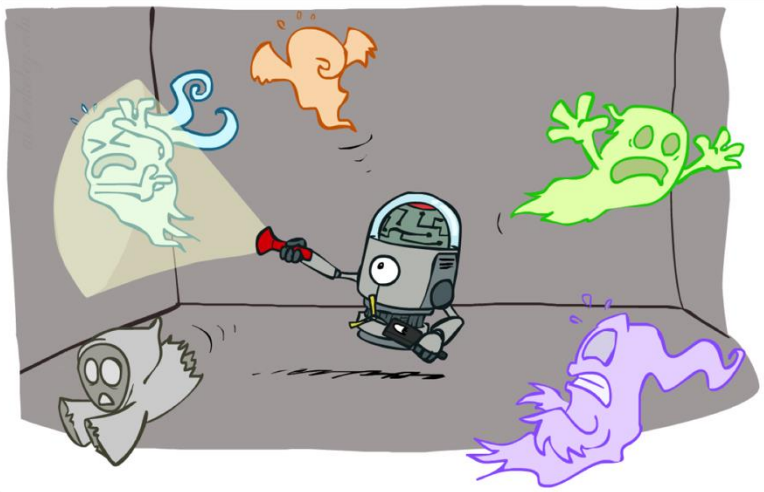
Most Likely Explanation

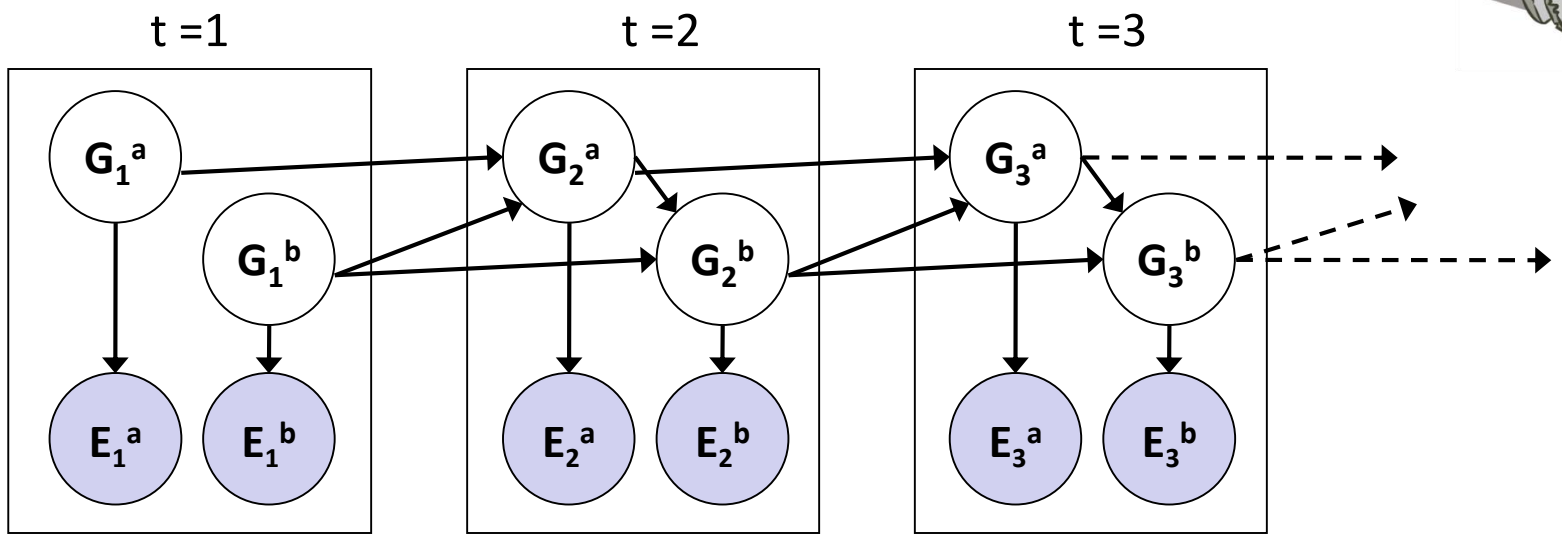
Dynamic Bayes Nets



Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time *t* can condition on those from *t-1*





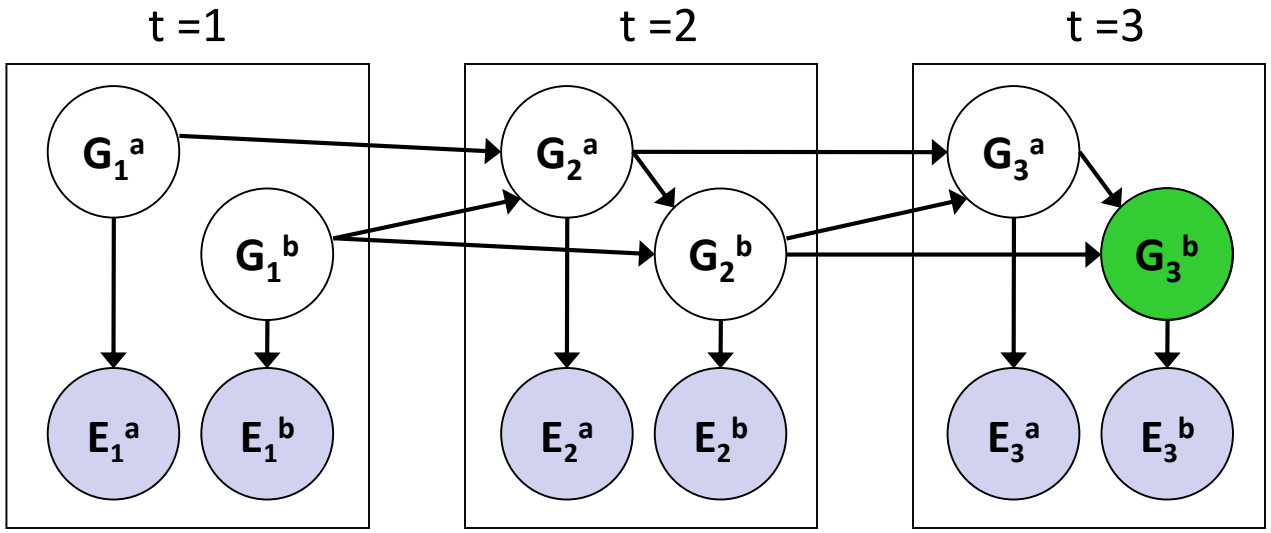
- Dynamic Bayes nets are a generalization of HMMs

DBN Particle Filters

- A particle is a complete sample for a time step
- **Initialize**: Generate prior samples for the t=1 Bayes net
  - Example particle: $\mathbf{G}_1^{\mathbf{a}} = (3,3)$ $\mathbf{G}_1^{\mathbf{b}} = (5,3)$
- **Elapse time**: Sample a successor for each particle
  - Example successor: $\mathbf{G}_2^{\mathbf{a}} = (2,3)$ $\mathbf{G}_2^{\mathbf{b}} = (6,3)$
- **Observe**: Weight each *entire* sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: $P(\mathbf{E}_1^{\mathbf{a}} \mid \mathbf{G}_1^{\mathbf{a}}) * P(\mathbf{E}_1^{\mathbf{b}} \mid \mathbf{G}_1^{\mathbf{b}})$
- **Resample:** Select prior samples (tuples of values) in proportion to their likelihood

Exact Inference in DBNs

- Variable elimination applies to dynamic Bayes nets
- Procedure: "unroll" the network for T time steps, then eliminate variables until $P(X_T | e_{1:T})$ is computed



- Online belief updates: Eliminate all variables from the previous time step; store factors for current time only