CS 188: Artificial Intelligence

Lecture 17: HMMs and Particle Filtering

Pieter Abbeel --- UC Berkeley

Many slides over this course adapted from Dan Klein, Stuart Russell, Andrew Moore

Reasoning over Time or Space

- Often, we want to reason about a sequence of observations
  - Speech recognition
  - Robot localization
  - User attention
  - Medical monitoring
- Need to introduce time (or space) into our models

5

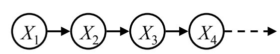
Outline

- Markov Models
  (= a particular Bayes net)
- Hidden Markov Models (HMMs)
  - Representation
    (= another particular Bayes net)
  - Inference
    - Forward algorithm (= variable elimination)
    - Particle filtering (= likelihood weighting with some tweaks)
    - Viterbi (= variable elimination, but replace sum by max
      = graph search)
- Dynamic Bayes' Nets
  - Representation
    - (= yet another particular Bayes' net)
  - Inference: forward algorithm and particle filtering

6

Markov Models

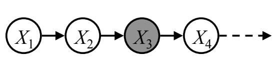
- A Markov model is a chain-structured BN
  - Each node is identically distributed (stationarity)
  - Value of X at a given time is called the state
  - As a BN:



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

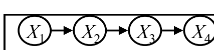
- Parameters: called transition probabilities or dynamics, specify how the state evolves over time (also, initial state probabilities)
- Same as MDP transition model, but no choice of action

Conditional Independence



- Basic conditional independence:
  - Past and future independent of the present
  - Each time step only depends on the previous
  - This is called the (first order) Markov property
- Note that the chain is just a (growing) BN
  - We can always use generic BN reasoning on it if we truncate the chain at a fixed length

8



Query: $P(X_4)$

- Slow answer: inference by enumeration
  - Enumerate all sequences of length t which end in s
  - Add up their probabilities

$$P(X_4) = \sum_{x_1,x_2,x_3} P(x_1, x_2, x_3, X_4)$$

$$\begin{aligned}
&= P(X_1 = +x_1)P(X_2 = +x_2|X_1 = +x_1)P(X_3 = +x_3|X_2 = +x_2)P(X_4|X_3 = +x_3) \\
&+P(X_1 = +x_1)P(X_2 = +x_2|X_1 = +x_1)P(X_3 = -x_3|X_2 = +x_2)P(X_4|X_3 = -x_3) \\
&+P(X_1 = +x_1)P(X_2 = -x_2|X_1 = +x_1)P(X_3 = +x_3|X_2 = -x_2)P(X_4|X_3 = +x_3) \\
&+P(X_1 = +x_1)P(X_2 = -x_2|X_1 = +x_1)P(X_3 = -x_3|X_2 = -x_2)P(X_4|X_3 = -x_3) \\
&+P(X_1 = -x_1)P(X_2 = +x_2|X_1 = -x_1)P(X_3 = +x_3|X_2 = +x_2)P(X_4|X_3 = +x_3) \\
&+P(X_1 = -x_1)P(X_2 = +x_2|X_1 = -x_1)P(X_3 = -x_3|X_2 = +x_2)P(X_4|X_3 = -x_3) \\
&+P(X_1 = -x_1)P(X_2 = -x_2|X_1 = -x_1)P(X_3 = -x_3|X_2 = -x_2)P(X_4|X_3 = -x_3) \\
&+P(X_1 = -x_1)P(X_2 = -x_2|X_1 = -x_1)P(X_3 = -x_3|X_2 = -x_2)P(X_4|X_3 = -x_3)
\end{aligned}$$

- = join on X_1, X_2, X_3, then sum over x_1, x_2, x_3

9

Query: $P(X_4)$

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

- Fast answer: variable elimination
  - Order: X_1, X_2, X_3

$$
\begin{aligned}
P(X_4) &= \sum_{x_1,x_2,x_3} P(x_1,x_2,x_3,X_4) \\
&= \sum_{x_3}\sum_{x_2}\sum_{x_1} P(X_4|x_3)P(x_3|x_2)P(x_2|x_1)P(x_1) \\
&= \sum_{x_3}\sum_{x_2} P(X_4|x_3)P(x_3|x_2)\sum_{x_1} P(x_2|x_1)P(x_1) \\
&= \sum_{x_3}\sum_{x_2} P(X_4|x_3)P(x_3|x_2)P(x_2) \\
&= \sum_{x_3} P(X_4|x_3)\sum_{x_2} P(x_3|x_2)P(x_2) \\
&= \sum_{x_3} P(X_4|x_3)P(x_3) \\
&= P(X_4)
\end{aligned}
$$

10

Query P(X_t)

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

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

- Variable elimination in order X_1, X_2, …, X_{t-1} computes for k = 2, 3, …, t

$$P(x_k) = \sum_{x_{k-1}} P(x_k|x_{k-1})P(x_{k-1})$$

Forward simulation

= "mini-forward algorithm"

Note: common thread in this lecture: special cases of algorithms we already know, and they have a special name in the context of HMMs for historical reasons.

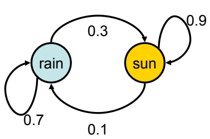
11

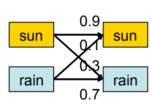
Example Markov Chain: Weather

- States: X = {rain, sun}
- CPT $P(X_t \mid X_{t-1})$:

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

Two new ways of representing the same CPT, that are often used for Markov models (These are not BNs!)





12

Example Run of Mini-Forward Algorithm

- From initial observation of sun

$$\begin{pmatrix}1.0\\0.0\end{pmatrix}\ \begin{pmatrix}0.9\\0.1\end{pmatrix}\ \begin{pmatrix}0.84\\0.16\end{pmatrix}\ \begin{pmatrix}0.804\\0.196\end{pmatrix} \Rightarrow \begin{pmatrix}0.75\\0.25\end{pmatrix}$$

$P(X_1)$ $P(X_2)$ $P(X_3)$ $P(X_4)$ $P(X_\infty)$

- From initial observation of rain

$$\begin{pmatrix}0.0\\1.0\end{pmatrix}\ \begin{pmatrix}0.3\\0.7\end{pmatrix}\ \begin{pmatrix}0.48\\0.52\end{pmatrix}\ \begin{pmatrix}0.588\\0.412\end{pmatrix} \Rightarrow \begin{pmatrix}0.75\\0.25\end{pmatrix}$$

$P(X_1)$ $P(X_2)$ $P(X_3)$ $P(X_4)$ $P(X_\infty)$

- From yet another initial distribution $P(X_1)$:

$$\begin{pmatrix}p\\1-p\end{pmatrix}\ \ldots \Rightarrow \begin{pmatrix}0.75\\0.25\end{pmatrix}$$

$P(X_1)$ $P(X_\infty)$

14

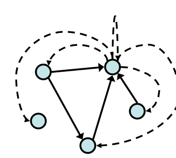
Stationary Distributions

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

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

Application of Markov Chain 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)

16

Application of Markov Chain Stationary Distribution: Gibbs Sampling*

- Each joint instantiation over all hidden and query variables is a state. Let X = H \union Q
- Transitions:
  - With probability 1/n resample variable X_j according to $P(X_j \mid x_1, x_2, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n, e_1, \ldots, e_m)$
- Stationary distribution:
  - = conditional distribution $P(X_1, X_2, \ldots, X_n | e_1, \ldots, e_m)$

  →When running Gibbs sampling long enough we get a sample from the desired distribution!

We did not prove this, all we did is stating this result.

17

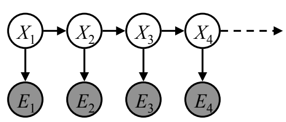
Outline

- ✓ Markov Models
  - (= a particular Bayes net)
- Hidden Markov Models (HMMs)
  - Representation
    - (= another particular Bayes net)
  - Inference
    - Forward algorithm (= variable elimination)
    - Particle filtering (= likelihood weighting with some tweaks)
    - Viterbi (= variable elimination, but replace sum by max = graph search)
- Dynamic Bayes' Nets
  - Representation
    - (= yet another particular Bayes' net)
  - Inference: forward algorithm and particle filtering

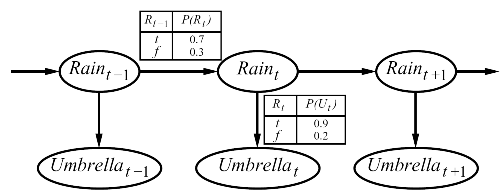
18

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 S
  - You observe outputs (effects) at each time step
  - As a Bayes' net:



Example



- An HMM is defined by:
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X|X_{-1})$
  - Emissions: $P(E|X)$

Ghostbusters HMM

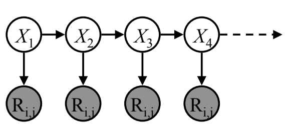
- $P(X_1)$ = uniform
- P(X|X') = usually move clockwise, but sometimes move in a random direction or stay in place
- $P(R_{ij}|X)$ = same sensor model as before: red means close, green means far away.

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

$P(X_1)$

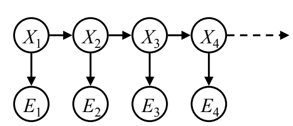
1/6	1/6	1/2
0	1/6	0
0	0	0

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



Conditional Independence

- HMMs have two important independence properties:
  - Markov hidden process, future depends on past via the present
  - Current observation independent of all else given current state



- Quiz: does this mean that evidence variables are guaranteed to be independent?
  - [No, they tend to correlated by the hidden state]

Real HMM Examples

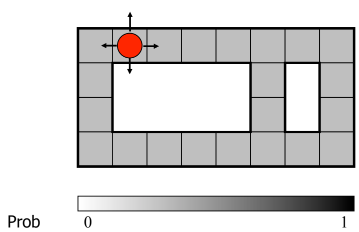
- Speech recognition HMMs:
  - Observations are acoustic signals (continuous valued)
  - States are specific positions in specific words (so, tens of thousands)
- Machine translation HMMs:
  - Observations are words (tens of thousands)
  - States are translation options
- Robot tracking:
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)

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)
- The Kalman filter was invented in the 60's and first implemented as a method of trajectory estimation for the Apollo program

Example: Robot Localization

Example from Michael Pfeiffer

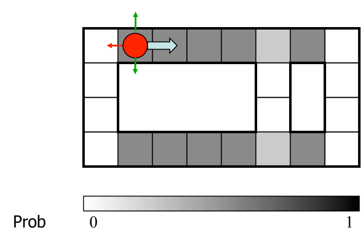


t=0

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.

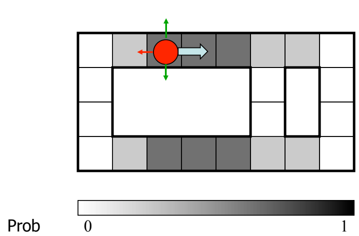
Example: Robot Localization



t=1

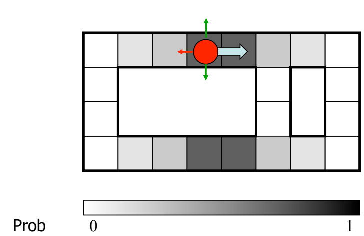
Lighter grey: was possible to get the reading, but less likely b/c required 1 mistake

Example: Robot Localization



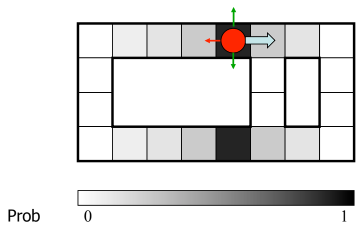
t=2

Example: Robot Localization



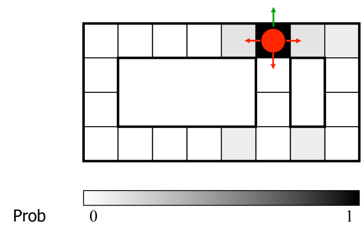
t=3

Example: Robot Localization



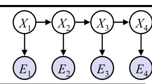
t=4

Example: Robot Localization



t=5

Query: $P(X_4|e_1,e_2,e_3,e_4)$ --- Variable Elimination, X_1, X_2, X_3



$$
\begin{aligned}
P(X_4|e_1,e_2,e_3,e_4) \propto P(X_4,e_1,e_2,e_3,e_4) &= \sum_{x_1,x_2,x_3} P(x_1,x_2,x_3,X_4,e_1,e_2,e_3,e_4) \\
&= \sum_{x_3}\sum_{x_2}\sum_{x_1} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)P(x_3|x_2)P(e_2|x_2)P(x_2|x_1)P(e_1|x_1)P(x_1) \\
&= \sum_{x_3}\sum_{x_2}\sum_{x_1} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)P(x_3|x_2)P(e_2|x_2)P(x_2|x_1)P(x_1,e_1) \\
&= \sum_{x_3}\sum_{x_2} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)P(x_3|x_2)P(e_2|x_2)\sum_{x_1} P(x_2|x_1)P(x_1,e_1) \\
&= \sum_{x_3}\sum_{x_2} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)P(x_3|x_2)P(e_2|x_2)P(x_2,e_1) \\
&= \sum_{x_3}\sum_{x_2} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)P(x_3|x_2)P(x_2,e_1,e_2) \\
&= \sum_{x_3} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)\sum_{x_2} P(x_3|x_2)P(x_2,e_1,e_2) \\
&= \sum_{x_3} P(e_4|X_4)P(X_4|x_3)P(e_3|x_3)P(x_3,e_1,e_2) \\
&= \sum_{x_3} P(e_4|X_4)P(X_4|x_3)P(x_3,e_1,e_2,e_3) \\
&= P(e_4|X_4)\sum_{x_3} P(X_4|x_3)P(x_3,e_1,e_2,e_3) \\
&= P(e_4|X_4)P(x_4,e_1,e_2,e_3) \\
&= P(X_4,e_1,e_2,e_3,e_4)
\end{aligned}
$$

Re-occurring computation:

$$
\begin{aligned}
& P(x_t,e_1,e_2,\ldots,e_t) \\
= \; & P(e_t|x_t)\sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1},e_1,e_2,\ldots,e_{t-1})
\end{aligned}
$$

32

The 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

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

$$
\begin{aligned}
P(x_t|e_{1:t}) \propto_X \; & P(x_t, e_{1:t}) \\
= & \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t}) \\
= & \sum_{x_{t-1}} P(x_{t-1}, e_{1:t-1})P(x_t|x_{t-1})P(e_t|x_t) \\
= & \; P(e_t|x_t)\sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}, e_{1:t-1})
\end{aligned}
$$

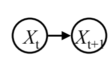
- = exactly variable elimination in order X_1, X_2, …

Belief Updating = the forward algorithm broken down into two steps and with normalization

- Forward algorithm: $P(x_t, e_{1:t}) = P(e_t|x_t)\sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}, e_{1:t-1})$
- Can break this down into:
  - Time update: $P(x_t, e_{1:t-1}) = \sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}, e_{1:t-1})$
  - Observation update: $P(x_t, e_{1:t}) = P(e_t|x_t)P(x_t, e_{1:t-1})$
- Normalizing in the observation update gives:
  - Time update: $P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|e_{1:t-1})$
  - Observation update: $P(x_t|e_{1:t}) \propto P(e_t|x_t)P(x_t|e_{1:t-1})$
- Notation: $B_t(x_t) = P(x_t|e_{1:t})$, $\quad B'_t(x_t) = P(x_t|e_{1:t-1})$
  - Time update: $B'_t(x_t) = \sum_{x_{t-1}} P(x_t|x_{t-1})B_{t-1}(x_{t-1})$
  - Observation update: $B_t(x_t) = P(e_t|x_t)B'_t(x_t)$

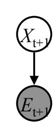
Belief updates can also easily be derived from basic probability

- **Passage of Time**
  - Given: $P(X_t)$, $P(X_{t+1} \mid X_t)$
  - Query: $P(x_{t+1})$ $\forall$ x_{t+1}



$$
\begin{aligned}
P(x_{t+1}) &= \sum_{x_t} P(x_t, x_{t+1}) \\
&= \sum_{x_t} P(x_t)P(x_{t+1}|x_t)
\end{aligned}
$$

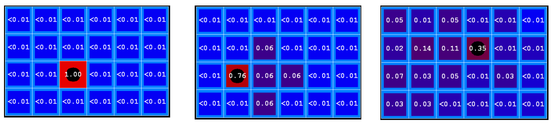
- **Observation**
  - Given: $P(X_{t+1})$, $P(e_{t+1} \mid X_{t+1})$
  - Query: $P(x_{t+1} \mid e_{t+1})$ $\forall$ x_{t+1}



$$
\begin{aligned}
P(x_{t+1}|e_{t+1}) &= P(x_{t+1}, e_{t+1})/P(e_{t+1}) \\
&\propto P(x_{t+1}, e_{t+1}) \\
&= P(x_{t+1})P(e_{t+1}|x_{t+1})
\end{aligned}
$$

Example: Passage of Time

- As time passes, uncertainty "accumulates"



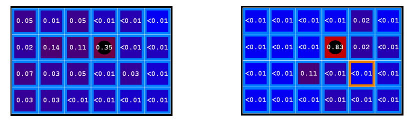
T = 1 T = 2 T = 5

$$B'(X') = \sum_x P(X'|x)B(x)$$

Transition model: ghosts usually go clockwise

Example: Observation

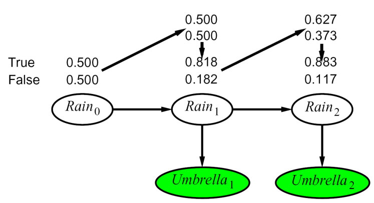
- As we get observations, beliefs get reweighted, uncertainty "decreases"



Before observation After observation

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

Example HMM



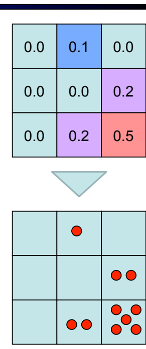
Outline

- Markov Models
  - (= a particular Bayes net)
- Hidden Markov Models (HMMs)
  - Representation
    - (= another particular Bayes net)
  - Inference
    - Forward algorithm (= variable elimination)
    - Particle filtering (= likelihood weighting with some tweaks)
    - Viterbi (= variable elimination, but replace sum by max = graph search)
- Dynamic Bayes' Nets
  - Representation
    - (= yet another particular Bayes' net)
  - Inference: forward algorithm and particle filtering

43

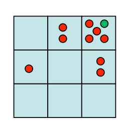
Particle Filtering

- Filtering: 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



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 will 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)

45

Particle Filtering: Elapse Time

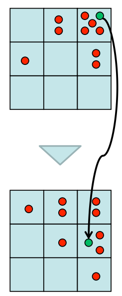
- Each particle is moved by sampling its next position from the transition model

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

  - This is like prior sampling – samples' frequencies reflect the transition probs
  - 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)

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)



Particle Filtering: Observe

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

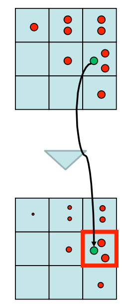
    $w(x) = P(e|x)$

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

  - As before, the probabilities don't sum to one, since most have been downweighted (in fact they sum to 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

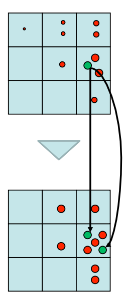


Particle Filtering: Resample

- 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

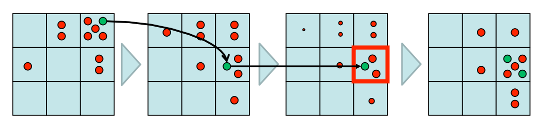
(New) Particles:
(3,2)
(2,2)
(3,2)
(2,3)
(3,3)
(3,2)
(1,3)
(2,3)
(3,2)
(3,2)



Recap: Particle Filtering

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

Elapse Weight Resample



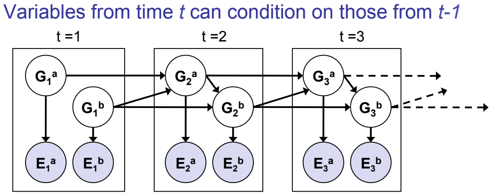
Particles:	Particles:	Particles:	(New) Particles:
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,3)	(2,3) w=.2	(2,2)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(3,2)	(3,1)	(3,1) w=.4	(2,3)
(3,3)	(3,3)	(3,3) w=.4	(3,3)
(3,2)	(3,2)	(3,2) w=.9	(3,2)
(1,2)	(1,3)	(1,3) w=.1	(1,3)
(3,3)	(2,3)	(2,3) w=.2	(2,3)
(3,3)	(3,2)	(3,2) w=.9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

Outline

- ✓ Markov Models
  (= a particular Bayes net)
- Hidden Markov Models (HMMs)
  - ✓ Representation
    (= another particular Bayes net)
  - Inference
    - ✓ Forward algorithm (= variable elimination)
    - ✓ Particle filtering (= likelihood weighting with some tweaks)
    - Viterbi (= variable elimination, but replace sum by max = graph search)
- Dynamic Bayes' Nets
  - Representation
    - (= yet another particular Bayes' net)
  - Inference: forward algorithm and particle filtering

Dynamic Bayes Nets (DBNs)

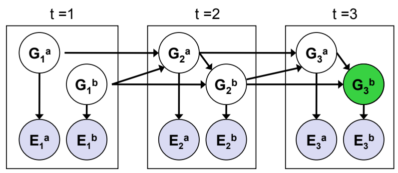
- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time *t* can condition on those from *t-1*



- Discrete valued dynamic Bayes nets are also HMMs

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

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: $\mathbf{G_1^a}$ = (3,3) $\mathbf{G_1^b}$ = (5,3)
- **Elapse time**: Sample a successor for each particle
  - Example successor: $\mathbf{G_2^a}$ = (2,3) $\mathbf{G_2^b}$ = (6,3)
- **Observe**: Weight each *entire* sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: $P(\mathbf{E_1^a} | \mathbf{G_1^a}) * P(\mathbf{E_1^b} | \mathbf{G_1^b})$
- **Resample:** Select prior samples (tuples of values) in proportion to their likelihood

Trick I to Improve Particle Filtering Performance: Low Variance Resampling

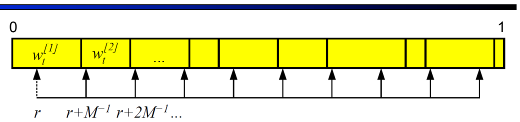


Figure 4.7 Principle of the low variance resampling procedure. We choose a random number r and then select those particles that correspond to $u = r + (m-1) \cdot M^{-1}$ where $m = 1, \ldots, M$.

- Advantages:
  - More systematic coverage of space of samples
  - If all samples have same importance weight, no samples are lost
  - Lower computational complexity

Trick II to Improve Particle Filtering Performance: Regularization

- If no or little noise in transitions model, all particles will start to coincide

→ regularization: introduce additional (artificial) noise into the transition model

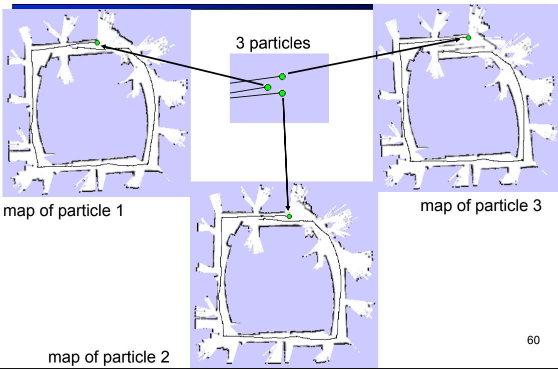
Robot Localization

- 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
- Demos: global-floor.gif

SLAM

- SLAM = Simultaneous Localization And Mapping
  - We do not know the map or our location
  - State consists of position AND map!
  - Main techniques: Kalman filtering (Gaussian HMMs) and particle methods

Particle Filter Example



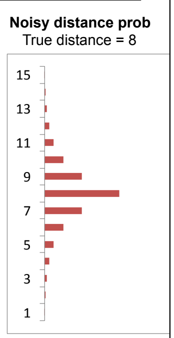
60

SLAM

- DEMOS
  - Intel-lab-raw-odo.wmv
  - Intel-lab-scan-matching.wmv
  - visionSlam_heliOffice.wmv

P4: Ghostbusters 2.0 (beta)

- **Plot:** Pacman's grandfather, Grandpac, learned to hunt ghosts for sport.
- He was blinded by his power, but could hear the ghosts' banging and clanging.
- **Transition Model:** All ghosts move randomly, but are sometimes biased
- **Emission Model:** Pacman knows a "noisy" distance to each ghost

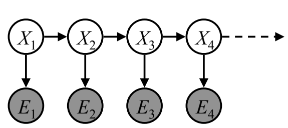


Outline

- ✓ Markov Models
  (= a particular Bayes net)
- Hidden Markov Models (HMMs)
  - ✓ Representation
    (= another particular Bayes net)
  - Inference
    - ✓ Forward algorithm (= variable elimination)
    - ✓ Particle filtering (= likelihood weighting with some tweaks)
    - Viterbi (= variable elimination, but replace sum by max
      = graph search)
- ✓ Dynamic Bayes' Nets
  - Representation
    - (= yet another particular Bayes' net)
  - Inference: forward algorithm and particle filtering

63

Best Explanation Queries



- Query: most likely seq:

$$\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t})$$

64

Best Explanation Query Solution Method 1: Search

$$\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t}) = \arg\max_{x_{1:t}} \prod_{i=1}^{t} P(x_t|x_{t-1})P(e_t|x_t)$$

slight abuse of notation, assuming $P(x_1|x_0) = P(x_1)$

$$= \arg\max_{x_{1:t}} \sum_{i=1}^{t} \log(P(x_t|x_{t-1})P(e_t|x_t))$$

- States: {(), $+x_1$, $-x_1$, $+x_2$, $-x_2$, …, $+x_t$, $-x_t$}
- Start state: ()
- Actions: in state x_k, choose any assignment for state x_{k+1}
- Cost: $\log\left(P(x_{k+1}|x_k)P(e_{k+1}|x_{k+1})\right)$
- Goal test: goal(x_k) = true iff k == t

→ Can run uniform cost graph search to find solution
→ Uniform cost graph search will take O(t d^2). Think about this!

Best Explanation Query Solution Method 2: Viterbi Algorithm (= max-product version of forward algorithm)

$$x^*_{1:T} = \arg\max_{x_{1:T}} P(x_{1:T}|e_{1:T}) = \arg\max_{x_{1:T}} P(x_{1:T}, e_{1:T})$$

$$\begin{aligned} m_t[x_t] &= \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t}) \\ &= \max_{x_{1:t-1}} P(x_{1:t-1}, e_{1:t-1}) P(x_t|x_{t-1}) P(e_t|x_t) \\ &= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) \max_{x_{1:t-2}} P(x_{1:t-1}, e_{1:t-1}) \\ &= P(e_t|x_t) \max_{x_{t-1}} P(x_t|x_{t-1}) m_{t-1}[x_{t-1}] \end{aligned}$$

Viterbi computational complexity: O(t d^2)

Compare to forward algorithm:

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

66

Further readings

- We are done with Part II Probabilistic Reasoning
- To learn more (beyond scope of 188):
  - Koller and Friedman, Probabilistic Graphical Models (CS281A)
  - Thrun, Burgard and Fox, Probabilistic Robotics (CS287)