Where are we?

Last time . . .
- Time in reasoning about uncertainty
- Markov assumption, stationarity
- Algorithms for reasoning about temporal processes
- Filtering and prediction

Today . . .
- Time and uncertainty II

Smoothing

Formula for backward message:
\[ P(e_{k+1:t} | x_k) = \sum_{x_{k+1}} P(e_{k+1} | x_{k+1}) P(e_{k+2:t} | x_{k+1}) P(x_{k+1} | x_k) \]

- First term is sensor model; third term is transition model; second is ‘recursive call’
- Define \( b_{k+1:t} = \text{BACKWARD}(b_{k+2:t}, e_{k+1:t}) \)
- The backward phase has to be initialised with \( b_{t+1:t} = P(e_{t+1:t} | x_t) = 1 \) (a vector of 1s) because probability of observing empty sequence is 1
- As before, all this is quite abstract, back to our example

Smoothing

- Smoothing is computation of distribution of past states given current evidence, i.e. \( P(X_k | e_{1:t}) \), \( 1 \leq k < t \)
- Easiest to view as 2-step process (up to \( k \), then \( k+1 \) to \( t \))

\[
\begin{align*}
P(X_k | e_{1:t}) &= P(X_k | e_{1:k}, e_{k+1:t}) \\
&= \alpha P(X_k | e_{1:k}) P(e_{k+1:t} | X_k, e_{1:k}) \\
&= \alpha P(X_k | e_{1:k}) P(e_{k+1:t} | X_k) \\
&= \alpha f_{1:k} b_{k+1:t}
\end{align*}
\]
- Here “backward” message is \( b_{k+1:t} = P(e_{k+1:t} | X_k) \) analogous to forward message
Umbrella World: Compute $P(R_1|u_1, u_2)$

![Diagram of umbrella world]

We have $P(R_1|u_1, u_2) = \alpha P(R_1|u_1)P(u_2|R_1)$

So we'll need to remind ourselves of $P(R_1|u_1)$ from last lecture:

$P(R_1) = \sum R_0 P(R_1|R_0) P(R_0) = \langle 0.7, 0.3 \rangle \times 0.5 + \langle 0.3, 0.7 \rangle \times 0.5 = \langle 0.5, 0.5 \rangle$

$\triangledown$ Update with evidence $U_1 = true$ yields:

$P(R_1|u_1) = \alpha P(u_1|R_1)P(R_1) = \alpha \langle 0.9, 0.2 \rangle \langle 0.5, 0.5 \rangle \approx \langle 0.818, 0.182 \rangle$

$\triangledown$ Forward filtering process yielded $\langle 0.818, 0.182 \rangle$ for first term

$\triangledown$ The second term can be obtained through backward recursion:

$P(u_2|R_1) = \sum R_2 P(u_2|R_2) P(R_2|R_1)$

$= (0.9 \times 1 \times \langle 0.7, 0.3 \rangle) + (0.2 \times 1 \times \langle 0.3, 0.7 \rangle) = \langle 0.69, 0.41 \rangle$

$\triangledown$ Plugged into the above equation this yields

$P(R_1|u_1, u_2) = \alpha \langle 0.818, 0.182 \rangle \times \langle 0.69, 0.41 \rangle \approx \langle 0.883, 0.117 \rangle$

$\triangledown$ So our confidence that it rained on Day 1 increases when we see the umbrella on the second day as well as the first.

$\triangledown$ A simple improved version of this that stores results runs in linear time (forward-backward algorithm)

Finding the most likely sequence

$\triangledown$ Suppose $[true, true, false, true, true]$ is the umbrella sequence for first five days, what is the most likely weather sequence that caused it?

$\triangledown$ Could use smoothing procedure to find posterior distribution for weather at each step and then use most likely weather at each step to construct sequence

$\triangledown$ NO! Smoothing considers distributions over individual time steps, but we must consider joint probabilities over all time steps

$\triangledown$ Actual algorithm is based on viewing each sequence as path through a graph (nodes=states at each time step)

Smoothing Example Continued

$P(R_1|u_1, u_2) = \alpha P(R_1|u_1)P(u_2|R_1)$

$\triangledown$ Forward filtering process yielded $\langle 0.818, 0.182 \rangle$ for first term

$\triangledown$ The second term can be obtained through backward recursion:

$P(u_2|R_1) = \sum R_2 P(u_2|R_2) P(R_2|R_1)$

$= (0.9 \times 1 \times \langle 0.7, 0.3 \rangle) + (0.2 \times 1 \times \langle 0.3, 0.7 \rangle) = \langle 0.69, 0.41 \rangle$

$\triangledown$ Plugged into the above equation this yields

$P(R_1|u_1, u_2) = \alpha \langle 0.818, 0.182 \rangle \times \langle 0.69, 0.41 \rangle \approx \langle 0.883, 0.117 \rangle$

$\triangledown$ So our confidence that it rained on Day 1 increases when we see the umbrella on the second day as well as the first.

$\triangledown$ A simple improved version of this that stores results runs in linear time (forward-backward algorithm)
Finding the most likely sequence

- There is a recursive relationship between most likely paths to $x_{t+1}$ and most likely paths to each state $x_t$
  \[
  \max_{x_1, \ldots, x_t} P(x_1, \ldots, x_t, x_{t+1} | e_{1:t+1}) = \alpha P(e_{t+1} | x_{t+1}) \max_{x_t} (P(x_{t+1} | x_t) \max_{x_1, \ldots, x_{t-1}} P(x_1, \ldots, x_{t-1}, x_t | e_{1:t}))
  \]

- This is like filtering only that the forward message is replaced by
  \[
  m_{1:t} = \max_{x_1, \ldots, x_{t-1}} P(x_1, \ldots, x_{t-1}, X_t | e_{1:t})
  \]

- And summation is now replaced by maximisation

Finding the most likely sequence

- This algorithm (Viterbi algorithm) is similar to filtering
- Runs forward along sequence computing $m$ message in each step
- Progress in example shown in part (b) of diagram above
- In the end it has probability for most likely sequence for reaching each final state
  
  Easy to determine overall most likely sequence

- Has to keep pointers from each state back to the best state that leads to it

Hidden Markov Models

- So far, we have seen a general model for temporal probabilistic reasoning (independent of transition/sensor models)
- In this and the following lecture we are going to look at more concrete models and applications
- **Hidden Markov Models (HMMs):** temporal probabilistic model in which state of the process is described by a single variable
- Like our umbrella example (single variable $\text{Rain}_t$)
- More than one variable can be accommodated, but only by combining them into a single “mega-variable”
- Structure of HMMs allows for a very simple and elegant matrix implementation of basic algorithms

Summary

- The forward-backward algorithm
- Finding the most likely sequence (Viterbi algorithm)
- Talked about HMMs
- HMMs: single state variable, simplifies algorithms (see other courses for these)
- Huge significance, for example in speech recognition:
  \[
  P(\text{words} | \text{signal}) = \alpha P(\text{signal} | \text{words}) P(\text{words})
  \]
- Vast array of applications, but also limits.
- Next time: **Dynamic Bayesian Networks**