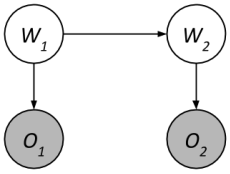
CS 188
Summer 2021

Regular Discussion 5B

HMMs

State variables W_t and observation (evidence) variables (O_t), which are supposed to be shaded below. Transition model $P(W_{t+1}|W_t)$. Sensor model $P(O_t|W_t)$. The joint distribution of the HMM can be factorized as

$$P(W_1, ..., W_T, O_1, ...O_T) = P(W_1) \prod_{t=1}^{T-1} P(W_{t+1}|W_t) \prod_{t=1}^{T} P(O_t|W_t) \tag{1}$$



Define the following belief distribution

- $B(W_t) = P(W_t|O_1, ..., O_t)$: Belief about state W_t given all the observations up until (and including) timestep t.
- $B'(W_t) = P(W_t|O_1, ..., O_{t-1})$: Belief about state W_t given all the observations up until (but not including) timestep t.

Forward Algorithm

- *Prediction update*: $B'(W_{t+1}) = \sum_{w_t} P(W_{t+1}|w_t)B(w_t)$
- *Observation update*: $B(W_{t+1}) \propto P(O_{t+1}|W_{t+1})B'(W_{t+1})$

Particle Filtering

The Hidden Markov Model analog to Bayes' net sampling is called **particle filtering**, and involves simulating the motion of a set of particles through a state graph to approximate the probability (belief) distribution of the random variable in question.

Instead of storing a full probability table mapping each state to its belief probability, we'll instead store a list of n particles, where each particle is in one of the d possible states in the domain of our time-dependent random variable.

Once we've sampled an initial list of particles, the simulation takes on a similar form to the forward algorithm, with a time elapse update followed by an observation update at each timestep:

- *Prediction update* - Update the value of each particle according to the transition model. For a particle in state W_t, sample the updated value from the probability distribution given by $Pr(W_{t+1}|w_t)$. Note the similarity of the prediction update to prior sampling with Bayes' nets, since the frequency of particles in any given state reflects the transition probabilities.

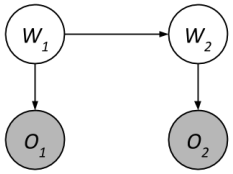
- *Observation update* - During the observation update for particle filtering, we use the sensor model $Pr(O_t|W_t)$ to weight each particle according to the probability dictated by the observed evidence and the particle's state. Specifically, for a particle in state w_t with sensor reading o_t, assign a weight of $Pr(o_t|w_t)$. The algorithm for the observation update is as follows:

  1. Calculate the weights of all particles as described above.
  2. Calculate the total weight for each state.
  3. If the sum of all weights across all states is 0, reinitialize all particles.
  4. Else, normalize the distribution of total weights over states and resample your list of particles from this distribution.

  Note the similarity of the observation update to likelihood weighting, where we again downweight samples based on our evidence.

1 HMMs

Consider the following Hidden Markov Model. O_1 and O_2 are supposed to be shaded.



W_1	$P(W_1)$
0	0.3
1	0.7

W_t	W_{t+1}	$P(W_{t+1}\|W_t)$
0	0	0.4
0	1	0.6
1	0	0.8
1	1	0.2

W_t	O_t	$P(O_t\|W_t)$
0	a	0.9
0	b	0.1
1	a	0.5
1	b	0.5

Suppose that we observe $O_1 = a$ and $O_2 = b$.
Using the forward algorithm, compute the probability distribution $P(W_2|O_1 = a, O_2 = b)$ one step at a time.

(a) Compute $P(W_1, O_1 = a)$.

(b) Using the previous calculation, compute $P(W_2, O_1 = a)$.

(c) Using the previous calculation, compute $P(W_2, O_1 = a, O_2 = b)$.

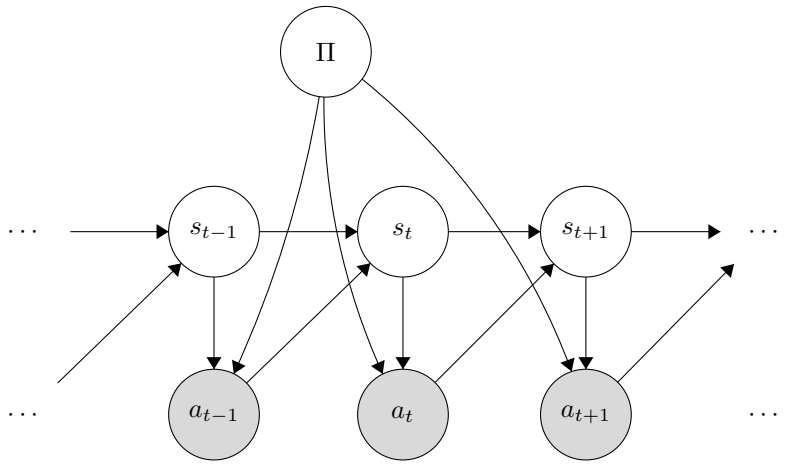
(d) Finally, compute $P(W_2|O_1 = a, O_2 = b)$.

2 Particle Filtering Apprenticeship

Consider a modified version of the apprenticeship problem. We are observing an agent's actions in an MDP and are trying to determine which out of a set $\{\pi_1, \ldots, \pi_n\}$ the agent is following. Let the random variable Π take values in that set and represent the policy that the agent is acting under. We consider only *stochastic* policies, so that A_t is a random variable with a distribution conditioned on S_t and Π. As in a typical MDP, S_t is a random variable with a distribution conditioned on S_{t-1} and A_{t-1}. The full Bayes net is shown below.

The agent acting in the environment knows what state it is currently in (as is typical in the MDP setting). Unfortunately, however, we, the observer, cannot see the states S_t. Thus we are forced to use an adapted particle filtering algorithm to solve this problem. Concretely, we will develop an efficient algorithm to estimate $P(\Pi \mid a_{1:t})$.

(a) The Bayes net for part (a) is



(i) Select all of the following that are guaranteed to be true in this model for $t > 10$:

- ☐ $S_t \perp\!\!\!\perp S_{t-2} \mid S_{t-1}$
- ☐ $S_t \perp\!\!\!\perp S_{t-2} \mid S_{t-1}, A_{1:t-1}$
- ☐ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi$
- ☐ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi, A_{1:t-1}$
- ☐ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi, S_{t-1}$
- ☐ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi, S_{t-1}, A_{1:t-1}$
- ☐ None of the above

We will compute our estimate for $P(\Pi \mid a_{1:t})$ by coming up with a recursive algorithm for computing $P(\Pi, S_t \mid a_{1:t})$. (We can then sum out S_t to get the desired distribution; in this problem we ignore that step.)

(ii) Write a recursive expression for $P(\Pi, S_t \mid a_{1:t})$ in terms of the CPTs in the Bayes net above. Hint: Think of the forward algorithm.

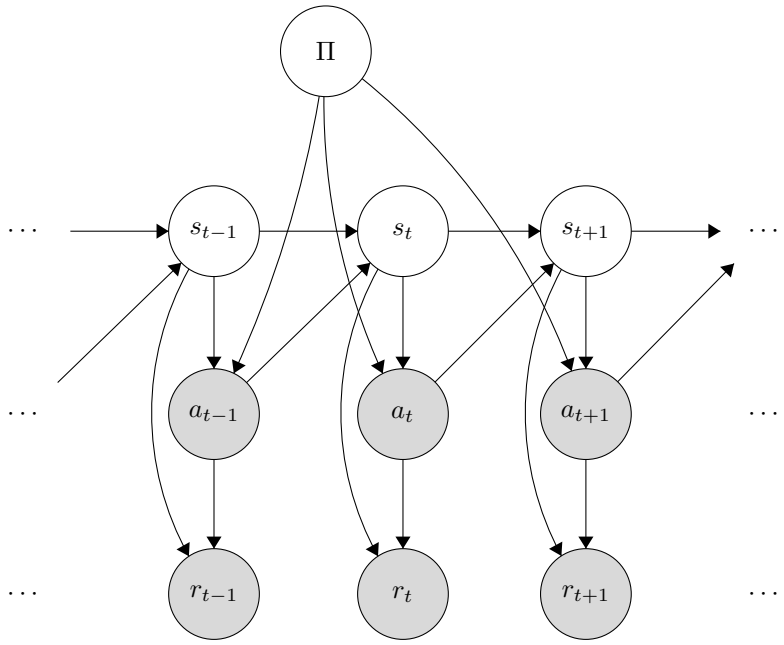
$P(\Pi, S_t \mid a_{1:t}) \propto$ _____

We now try to adapt particle filtering to approximate this value. Each particle will contain a single state s_t and a potential policy π_i.

(iii) The following is pseudocode for the body of the loop in our adapted particle filtering algorithm. Fill in the boxes with the correct values so that the algorithm will approximate $P(\Pi, S_t \mid a_{1:t})$.

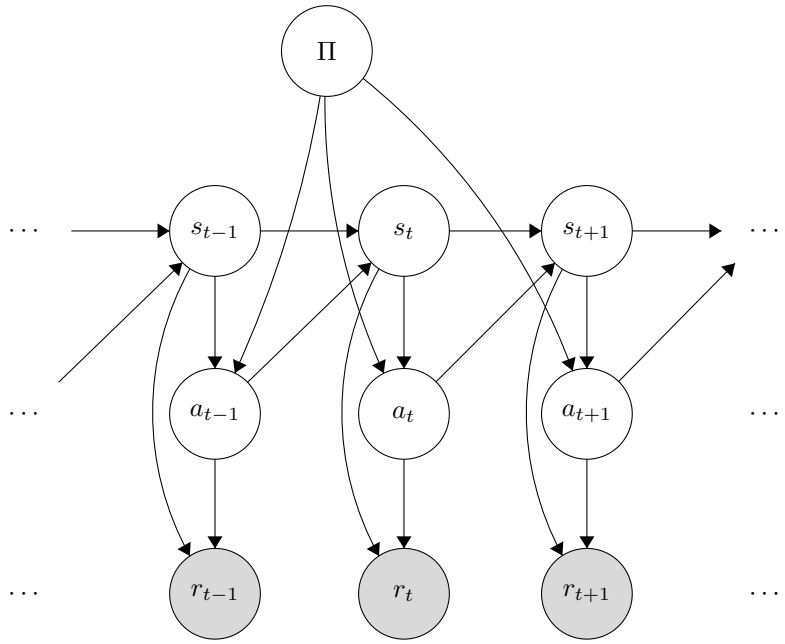
1. Elapse time: for each particle (s_t, π_i), sample a successor s_{t+1} from
   [ ]. The policy π' in the new particle is [ ].
2. Incorporate evidence: To each new particle (s_{t+1}, π'), assign weight
   [ ].
3. Resample particles from the weighted particle distribution.

(b) We now observe the acting agent's actions *and* rewards at each time step (but we still don't know the states). Unlike the MDPs in lecture, here we use a stochastic reward function, so that R_t is a random variable with a distribution conditioned on S_t and A_t. The new Bayes net is given by



Notice that the observed rewards do in fact give useful information since d-separation does not give that $R_t \perp\!\!\!\perp \Pi \mid A_{1:t}$. Give an active path connecting R_t and Π when $A_{1:t}$ are observed. Your answer should be an ordered list of nodes in the graph, for example "$S_t, S_{t+1}, A_t, \Pi, A_{t-1}, R_{t-1}$".

(c) We now observe *only* the sequence of rewards and no longer observe the sequence of actions. The new Bayes net is:



We will compute our estimate for $P(\Pi \mid r_{1:t})$ by coming up with a recursive algorithm for computing $P(\Pi, S_t, A_t \mid r_{1:t})$. (We can then sum out S_t and A_t to get the desired distribution; in this problem we ignore that step.)

(i) Write a recursive expression for $P(\Pi, S_t, A_t \mid r_{1:t})$ in terms of the CPTs in the Bayes net above.

$P(\Pi, S_t, A_t \mid r_{1:t}) \propto$ _____

We now try to adapt particle filtering to approximate this value. Each particle will contain a single state s_t, a single action a_t, and a potential policy π_i.

(ii) The following is pseudocode for the body of the loop in our adapted particle filtering algorithm. Fill in the boxes with the correct values so that the algorithm will approximate $P(\Pi, S_t, A_t \mid r_{1:t})$.

1. Elapse time: for each particle (s_t, a_t, π_i), sample a successor state s_{t+1} from [ ]. Then, sample a successor action a_{t+1} from [ ]. The policy π' in the new particle is [ ].
2. Incorporate evidence: To each new particle (s_{t+1}, a_{t+1}, π'), assign weight

 .

3. Resample particles from the weighted particle distribution.