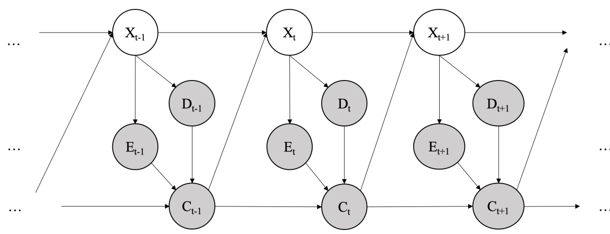
CS 188
Summer 2023

Final Review HMMs Solutions

1 We Are Getting Close...

Mesut is trying to remotely control a car, which has gone out of his view. The unknown state of the car is represented by the random variable X. While Mesut can't see the car itself, his high-tech sensors on the car provides two useful readings: an estimate (E) of the distance to the car in front, and a detection model (D) that detects if the car is headed into a wall. Using these two readings, Mesut applies the controls (C), which determine the velocity of the car by changing the acceleration. The DBN below describes the setup.



(a) For the above DBN, complete the equations for performing updates. (Hint: think about the prediction update and observation update equations in the forward algorithm for HMMs.)

Time elapse: **(i)** = **(ii)** **(iii)** **(iv)** $P(x_{t-1}|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$

(i) ○ $P(x_t)$ ● $P(x_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$ ○ $P(e_t, d_t, c_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$

(ii) ○ $P(c_{0:t-1})$ ○ $P(x_{0:t-1}, c_{0:t-1})$ ○ $P(e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$
○ $P(e_{0:t}, d_{0:t}, c_{0:t})$ ● 1

(iii) ● $\Sigma_{x_{t-1}}$ ○ Σ_{x_t} ○ $\max_{x_{t-1}}$ ○ $\max_{x_t}$ ○ 1

(iv) ○ $P(x_{t-1}|x_{t-2})$ ○ $P(x_{t-1}, x_{t-2})$ ○ $P(x_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$
○ $P(x_t|x_{t-1})$ ○ $P(x_t, x_{t-1})$ ○ $P(x_t, e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$
● $P(x_t|x_{t-1}, c_{t-1})$ ○ $P(x_t, x_{t-1}, c_{t-1})$ ○ 1

Recall the prediction update of forward algorithm: $P(x_t|o_{0:t-1}) = \Sigma_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}|o_{0:t-1})$, where o is the observation. Here it is similar, despite that there are several observations at each time, which means o_t corresponds to e_t, d_t, c_t for each t, and that X is dependent on the C value of the previous time, so we need $P(x_t|x_{t-1}, c_{t-1})$ instead of $P(x_t|x_{t-1})$. Also note that X is independent of D_{t-1}, E_{t-1} given C_{t-1}, X_{t-1}.

Update to incorporate new evidence at time t:
$P(x_t|e_{0:t}, d_{0:t}, c_{0:t}) =$ **(v)** **(vi)** **(vii)** **Your choice for (i)**

(v) ○ $(P(c_t|c_{0:t-1}))^{-1}$ ○ $(P(e_t|e_{0:t-1}) P(d_t|d_{0:t-1}) P(c_t|c_{0:t-1}))^{-1}$
● $(P(e_t, d_t, c_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1}))^{-1}$ ○ $(P(e_{0:t-1}|e_t) P(d_{0:t-1}|d_t) P(c_{0:t-1}|c_t))^{-1}$
○ $(P(e_{0:t-1}, d_{0:t-1}, c_{0:t-1}|e_t, d_t, c_t))^{-1}$ ○ 1

(vi) ○ $\Sigma_{x_{t-1}}$ ○ Σ_{x_t} ○ Σ_{x_{t-1}, x_t} ○ $\max_{x_{t-1}}$ ○ $\max_{x_t}$ ● 1

(vii) ☐ $P(x_t|e_t, d_t, c_t)$ ☐ $P(x_t, e_t, d_t, c_t)$
☐ $P(x_t|e_t, d_t, c_t, c_{t-1})$ ☐ $P(x_t, e_t, d_t, c_t, c_{t-1})$
■ $P(e_t, d_t|x_t)P(c_t|e_t, d_t, c_{t-1})$ ☐ $P(e_t, d_t, c_t|x_t)$ ○ 1

Recall the observation update of forward algorithm: $P(x_t|o_{0:t}) \propto P(x_t, o_t|o_{0:t-1}) = P(o_t|x_t)P(x_t|o_{0:t-1})$.

Here the observations o_t corresponds to e_t, d_t, c_t for each t. Apply the Chain Rule, we are having $P(x_t|e_{0:t}, d_{0:t}, c_{0:t}) \propto P(x_t, e_t, d_t, c_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1}) = P(e_t, d_t, c_t|x_t, c_{t-1})P(x_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$
$= P(e_t, d_t|x_t)P(c_t|e_t, d_t, c_{t-1})P(x_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})$.
Note that in $P(e_t, d_t, c_t|x_t, c_{t-1})$, we cannot omit c_{t-1} due to the arrow between c_t and c_{t-1}.
To calculate the normalizing constant, use Bayes Rule: $P(x_t|e_{0:t}, d_{0:t}, c_{0:t}) = \frac{P(x_t, e_t, d_t, c_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})}{P(e_t, d_t, c_t|e_{0:t-1}, d_{0:t-1}, c_{0:t-1})}$.

(viii) Suppose we want to do the above updates in one step and use normalization to reduce computation. Select all the terms that are not explicitly calculated in this implementation.
DO **NOT** include the choices if their values are 1.

☐ **(ii)** ☐ **(iii)** ☐ **(iv)** ■ **(v)** ☐ **(vi)** ☐ **(vii)** ○ None of the above

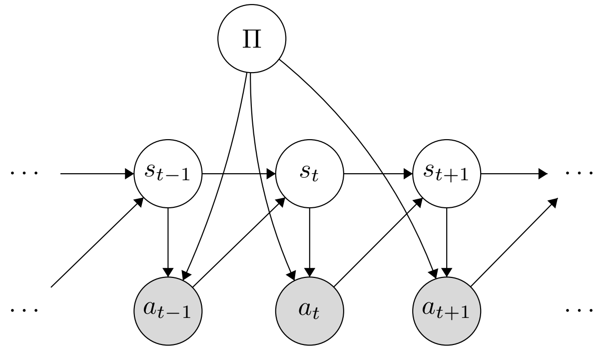
(v) is a constant, so we don't calculate it during implementation and simply do a normalization instead. Everything else is necessary.

2 Particle Filtering Apprenticeship

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 $t > 3$:

☐ $S_t \perp\!\!\!\perp S_{t-2} \mid S_{t-1}$
■ $S_t \perp\!\!\!\perp S_{t-2} \mid S_{t-1}, A_{1:t-1}$
☐ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi$
☐ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi, A_{1:t-1}$
■ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi, S_{t-1}$
■ $S_t \perp\!\!\!\perp S_{t-2} \mid \Pi, S_{t-1}, A_{1:t-1}$
☐ 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.

$$P(\Pi, S_t \mid a_{1:t}) \propto \sum_{s_{t-1}} P(\Pi, s_{t-1} \mid a_{1:t-1}) P(a_t \mid S_t, \Pi) P(S_t \mid s_{t-1}, a_{t-1})$$

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

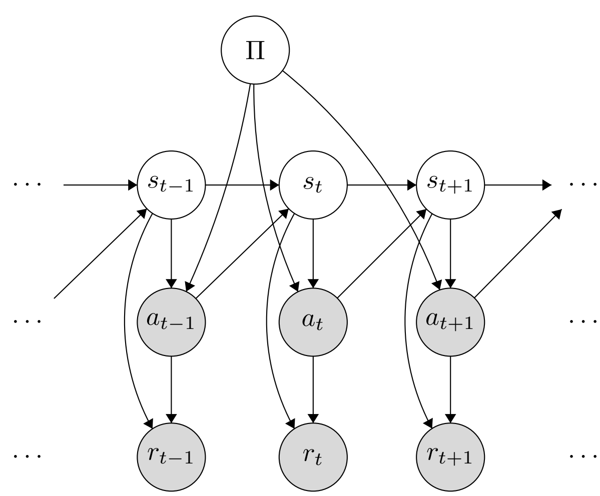
1. Elapse time: for each particle (s_t, π_i), sample a successor s_{t+1} from $P(S_{t+1} \mid s_t, a_t)$.

   The policy π' in the new particle is π_i .

2. Incorporate evidence: To each new particle (s_{t+1}, π'), assign weight $P(a_{t+1} \mid s_{t+1}, \pi')$.

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 $R_t \perp\!\!\!\perp \Pi \mid A_{1:t}$.

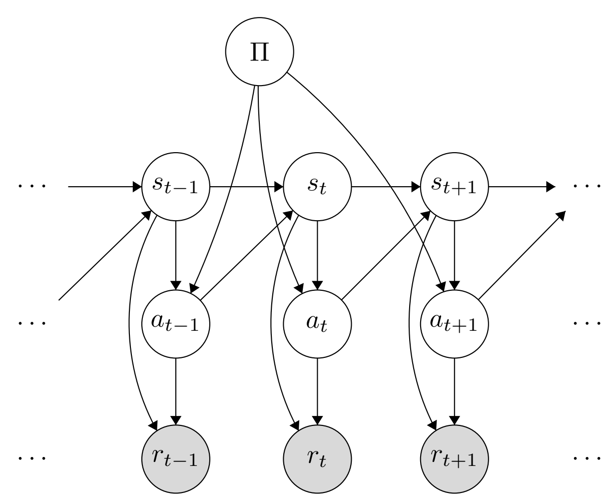
(i) Give an active path connecting R_t and Π when $A_{1:t}$ are observed. Your answer should be an ordered list of nodes in the graph, for example "$S_t, S_{t+1}, A_t, \Pi, A_{t-1}, R_{t-1}$".

R_t, S_t, A_t, Π. This list reversed is also correct, and many other similar (though more complicated) paths are also correct.

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

$$P(\Pi, S_t \mid a_{1:t}, r_{1:t}) \propto \sum_{s_{t-1}} P(\Pi, s_{t-1} \mid a_{1:t-1}, r_{1:t-1}) P(a_t \mid S_t, \Pi) P(S_t \mid s_{t-1}, a_{t-1}) P(r_t \mid a_t, S_t)$$

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



(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.

$$P(\Pi, S_t, A_t \mid r_{1:t}) \propto \sum_{s_{t-1}} \sum_{a_{t-1}} P(\Pi, s_{t-1}, a_{t-1} \mid r_{1:t-1}) P(A_t \mid S_t, \Pi) P(S_t \mid s_{t-1}, a_{t-1}) P(r_t \mid S_t, A_t)$$

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 $P(S_{t+1} \mid s_t, a_t)$.

   Then, sample a successor action a_{t+1} from $P(A_{t+1} \mid s_{t+1}, \pi_i)$.

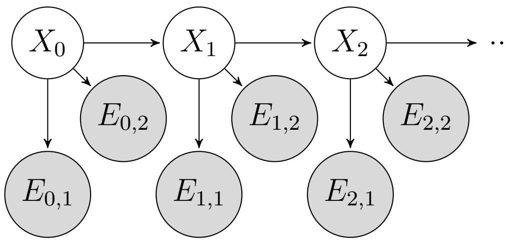
   The policy π' in the new particle is π_i.

2. Incorporate evidence: To each new particle (s_{t+1}, a_{t+1}, π'), assign weight $P(r_{t+1} \mid s_{t+1}, a_{t+1})$.
3. Resample particles from the weighted particle distribution.

3 Particle Filtering

You've chased your arch-nemesis Leland to the Stanford quad. You enlist two robo-watchmen to help find him! The grid below shows the campus, with ID numbers to label each region. Leland will be moving around the campus. His location at time step t will be represented by random variable X_t. Your robo-watchmen will also be on campus, but their locations will be fixed. Robot 1 is always in region 1 and robot 2 is always in region 9. (See the * locations on the map.) At each time step, each robot gives you a sensor reading to help you determine where Leland is. The sensor reading of robot 1 at time step t is represented by the random variable $E_{t,1}$. Similarly, robot 2's sensor reading at time step t is $E_{t,2}$. The Bayes Net to the right shows your model of Leland's location and your robots' sensor readings.

1*	2	3	4	5
6	7	8	9*	10
11	12	13	14	15



In each time step, Leland will either stay in the same region or move to an adjacent region. For example, the available actions from region 4 are (WEST, EAST, SOUTH, STAY). He chooses between all available actions with equal probability, regardless of where your robots are. Note: moving off the grid is not considered an available action.

Each robot will detect if Leland is in an adjacent region. For example, the regions adjacent to region 1 are 1, 2, and 6. If Leland is in an adjacent region, then the robot will report $NEAR$ with probability 0.8. If Leland is not in an adjacent region, then the robot will still report $NEAR$, but with probability 0.3.

For example, if Leland is in region 1 at time step t the probability tables are:

E	$P(E_{t,1} \mid X_t = 1)$	$P(E_{t,2} \mid X_t = 1)$
$NEAR$	0.8	0.3
FAR	0.2	0.7

(a) Suppose we are running particle filtering to track Leland's location, and we start at $t = 0$ with particles $[X = 6,\ X = 14,\ X = 9,\ X = 6]$. Apply a forward simulation update to each of the particles using the random numbers in the table below.

Assign region IDs to sample spaces in numerical order. For example, if, for a particular particle, there were three possible successor regions 10, 14 and 15, with associated probabilities, $P(X = 10)$, $P(X = 14)$ and $P(X = 15)$, and the random number was 0.6, then 10 should be selected if $0.6 \leq P(X = 10)$, 14 should be selected if $P(X = 10) < 0.6 < P(X = 10) + P(X = 14)$, and 15 should be selected otherwise.

Particle at $t = 0$	Random number for update	Particle after forward simulation update
$X = 6$	0.864	11
$X = 14$	0.178	9
$X = 9$	0.956	14
$X = 6$	0.790	11

(b) Some time passes and you now have particles [$X = 6$, $X = 1$, $X = 7$, $X = 8$] at the particular time step, but you have not yet incorporated your sensor readings at that time step. Your robots are still in regions 1 and 9, and both report $NEAR$. What weight do we assign to each particle in order to incorporate this evidence?

Particle	Weight
$X = 6$	0.8 * 0.3
$X = 1$	0.8 * 0.3
$X = 7$	0.3 * 0.3
$X = 8$	0.3 * 0.8

(c) To decouple this question from the previous question, let's say you just incorporated the sensor readings and found the following weights for each particle (these are not the correct answers to the previous problem!):

Particle	Weight
$X = 6$	0.1
$X = 1$	0.4
$X = 7$	0.1
$X = 8$	0.2

Normalizing gives us the distribution

$$X = 1 : 0.4/0.8 = 0.5$$
$$X = 6 : 0.1/0.8 = 0.125$$
$$X = 7 : 0.1/0.8 = 0.125$$
$$X = 8 : 0.2/0.8 = 0.25$$

Use the following random numbers to resample you particles. As on the previous page, **assign region IDs to sample spaces in numerical order.**

Random number:	0.596	0.289	0.058	0.765
Particle:	6	1	1	8