

ROB311 Quiz 3

Hanhee Lee

March 29, 2025

Contents

| | | |
|----------|--|-----------|
| 1 | Reinforcement Learning | 2 |
| 1.1 | Running Average Update Rule | 2 |
| 1.2 | Q-Learning Algorithm | 3 |
| 1.3 | Modified Q-Learning Algorithm | 3 |
| 1.4 | Training vs. Testing | 4 |
| 1.4.1 | K Sims, 1 Test | 4 |
| 1.4.2 | K Tests | 4 |
| 2 | Partially Observable MDPs (POMDPs) | 5 |
| 2.1 | Bayesian Network | 6 |
| 2.2 | Belief (Probability Distribution) Over the States: | 6 |
| 2.3 | Examples | 7 |
| 3 | Estimating the Optimal Quality Function | 10 |
| 3.1 | Estimating the Optimal Quality Function | 10 |
| 3.2 | Exploration versus Exploitation | 10 |
| 3.2.1 | Simplified Case: | 10 |
| 3.3 | Alternate Policies | 12 |

Partially Observable Probabilistic Decision Problems

1 Reinforcement Learning

Summary: In a RL problem, $p(\cdot | \cdot, \cdot)$ and/or $r(\cdot, \cdot)$ unknown, so we have to estimate q-star empirically.

Equation

$$q^*(s, a) = \lim_{K \rightarrow \infty} \bar{R}_K$$

- $\bar{R}_K = \frac{1}{K} \sum_{k=1}^K r_k$: empirical average reward.
- r_k : reward obtained in the k^{th} simulation.
- K : # of times action a taken in state s (# of simulations)
- $\gamma = 0$

$$q^*(s, a) \leftarrow q^*(s, a) + \frac{1}{N(s, a)} (r(s, a, s') - q^*(s, a))$$

- $N(s, a)$: # of times action a taken in state s .
- $\gamma = 0$

$$q^*(s, a) \leftarrow q^*(s, a) + \frac{1}{N(s, a)} \left(\left[r(s, a, s') + \gamma \max_{a'} q^*(s', a') \right] - q^*(s, a) \right)$$

- Using old q^* values to estimate.
- $\gamma \neq 0$

$$\pi(a | s) = \begin{cases} 1 & a = \arg \max_{a'} q^*(s, a) \\ 0 & \text{otherwise} \end{cases}$$

1.1 Running Average Update Rule

Definition:

$$\bar{x} \leftarrow \bar{x} + \alpha(x_{\text{new}} - \bar{x}).$$

- α : learning rate

1.2 Q-Learning Algorithm

Algorithm:

```

1 procedure Q_LEARNING():
2   for each episode do
3     set initial state  $s \leftarrow s_0$ 
4     while  $s \notin \mathcal{T}$  do #  $\mathcal{T}$ : terminal states
5       randomly choose an action in  $\mathcal{A}(s)$ 
6       get next state,  $s'$ , and reward  $r$ 
7       update  $N(s, a)$  and  $q^*(s, a)$  as follows:
8
9       
$$q^*(s, a) \leftarrow q^*(s, a) + \frac{1}{N(s, a)} \left( r(s, a, s') + \gamma \max_{a'} q^*(s', a') - q^*(s, a) \right)$$

10
11      
$$N(s, a) \leftarrow N(s, a) + 1$$

12
13       $s \leftarrow s'$ 
14    end while
15  end for

```

- **Note:** Possible infinite while loop if \mathcal{T} is not reached.

1.3 Modified Q-Learning Algorithm

Algorithm:

```

1 procedure Q_LEARNING():
2   for each episode do
3      $l \leftarrow 0$ 
4     set initial state  $s \leftarrow s_0$ 
5     while  $s \notin \mathcal{T}$  and  $l < l_{\max}$  do
6       randomly choose an action in  $\mathcal{A}(s)$ 
7       get next state,  $s'$ , and reward  $r$ 
8       update  $N(s, a)$  and  $q^*(s, a)$  as follows:
9
10      
$$q^*(s, a) \leftarrow q^*(s, a) + \frac{1}{N(s, a)} \left( r(s, a, s') + \gamma \max_{a'} q^*(s', a') - q^*(s, a) \right)$$

11
12      
$$N(s, a) \leftarrow N(s, a) + 1$$

13
14       $l \leftarrow l + 1$ 
15       $s \leftarrow s'$ 
16    end while
17  end for

```

Notes: Choice of γ and l_{\max} are coupled:

- $\gamma \approx 1$ requires large l_{\max}
- $\gamma \approx 0$ requires small l_{\max}

1.4 Training vs. Testing

Notes: Episodes are classified as either:

- training (sim): reward accumulated during episode does not count
- testing (test): reward accumulated during episode counts

1.4.1 K Sims, 1 Test

Notes:

1. select actions randomly during K simulations
2. extract optimal policy, π^*
3. use π^* during test

1.4.2 K Tests

Notes:

- maximize average reward over K tests
- must balance between exploration and exploitation
- Common ways to balance exploration and exploitation: ε -greedy strategy, UCB algorithm

| Strategy | Description |
|-----------------------|--|
| ε -greedy | <p>choose optimal action with probability $\varepsilon(k)$</p> <ul style="list-style-type: none"> • In episode k, choose the optimal action with probability $\varepsilon(k)$, where: <ul style="list-style-type: none"> – $\varepsilon(0) \approx 0$ – $\varepsilon(k)$ is increasing as you keep exploring. – $\varepsilon(k) \rightarrow 1$ as $k \rightarrow \infty$ • Common choice for $\varepsilon(k)$ is $1 - \frac{1}{k}$. |
| UCB algorithm | <p>choose action that maximizes $\text{UCB}(\cdot)$</p> $\text{UCB}(s, a) = \begin{cases} q^*(s, a) + C \sqrt{\frac{\log k}{N(s, a)}}, & \text{if } N(s, a) > 0 \\ \infty, & \text{otherwise} \end{cases}$ <ul style="list-style-type: none"> • In episode k, choose the action that maximizes $\text{UCB}(\cdot)$. • C: exploration parameter • $N(s, a)$: # of times a taken from s. |

2 Partially Observable MDPs (POMDPs)

Summary: In a POMDPs, we assume that:

- environment modelled using state space, \mathcal{S}
- single agent
- S_t = state after transition t
- A_t = action inducing transition t
- stochastic state transitions with memoryless property:

$$S_T \perp S_0, A_1, \dots, A_{T-1}, S_{T-2} \mid S_{T-1}, A_T$$

- R_t = reward for transition t , i.e., (S_{T-1}, A_T, S_T)
- O_t = observation of S_t
 - Measurement of a state (i.e. approximation, so may not be exact)
 - **Key:** Since actual state is unknown, so are legal actions.

| Name | Function: |
|-------------------------------|---|
| Initial state distribution | $p_0(s) := \mathbb{P}[S_0 = s]$ |
| Transition distribution | $p(s' s, a) := \mathbb{P}[S_t = s' A_t = a, S_{t-1} = s]$ <ul style="list-style-type: none"> • Assume $\mathcal{A}(s) = \mathcal{A}(s') := \mathcal{A} \forall s, s'$ (i.e. since actual state is unknown, so are legal actions, so assume all actions are legal): <ul style="list-style-type: none"> – if $a \notin \mathcal{A}(s)$, then $p(s' s, a) = 0$ for all $s' \neq s$ |
| Reward function | $r(s, a, s') := \text{reward for transition } (s, a, s')$ <ul style="list-style-type: none"> • Assume $\mathcal{A}(s) = \mathcal{A}(s') := \mathcal{A} \forall s, s'$ (i.e. since actual state is unknown, so are legal actions, so assume all actions are legal): <ul style="list-style-type: none"> – if $a \notin \mathcal{A}(s)$, then $r(s, a, s') = 0$ for all s' |
| Policy for choosing actions | $\pi_t(a o_0, \dots, o_t) := \mathbb{P}[A_t = a O_0 = o_0, \dots, O_t = o_t]$ <ul style="list-style-type: none"> • Observe that policy is now time-dependent. • Special Case: If we assume the agent cannot use past observations, $A_t \perp O_0, \dots, O_{t-1} \mid O_t$, policy becomes time-independent, $\pi_t(a o_0, \dots, o_t) = \pi_0(a o_t).$ <ul style="list-style-type: none"> – Only need to specify π_0. |
| Measurement model | $m(o s) := \mathbb{P}[O_t = o S_t = s]$ |
| Belief after t observations | $b_t(s_t a_{1:t}, o_{0:t}) = \mathbb{P}[S_t = s_t A_t = a_t, O_{0:t} = o_{0:t}]$ $b_t(s_t a_{1:t}, o_{0:t}) = m(o_t s_t) \sum_{s_{t-1}} p(s_t s_{t-1}, a_t) b_{t-1}(s_{t-1} a_{1:t-1}, o_{0:t-1})$ <ul style="list-style-type: none"> • b_t: Probability distribution • $b_0(s_0) = \mathbb{P}[S_0 = s_0]$: Initial belief distribution • Only holds for $t \geq 1$. <ul style="list-style-type: none"> – @t: Measurement before and after action for the belief is the same except at $t = 0$ b/c of initial belief. • For $t = 0$ (assuming uniform prior): $b_0(s_0 o_0) = \frac{m(o_0 s_0)}{\sum_s m(o_0 s)}$. |

2.1 Bayesian Network

Notes: $S_0, O_0, A_1, R_1, S_1, O_1, A_2, R_2, S_2, O_2, \dots$ form a Bayesian network:

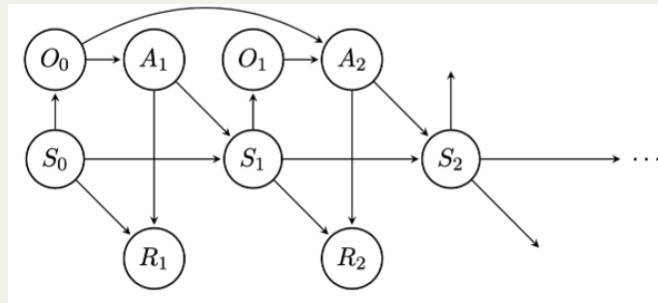


Figure 1

- Assuming $A_t \perp O_0, \dots, O_{t-1} \mid O_t$. WHERE DOES THIS COME INTO PLAY.

2.2 Belief (Probability Distribution) Over the States:

Notes: Assume actual state is the most likely state.

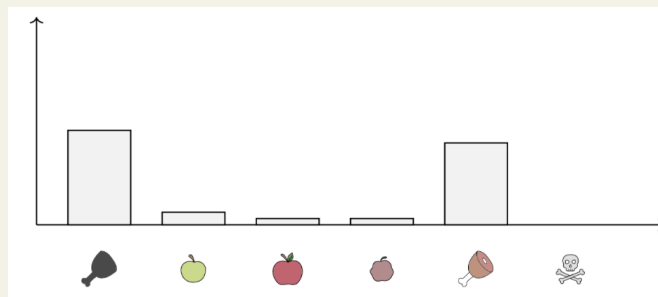


Figure 2

- Usually assume uniform distribution before you observe anything.
- Flow:** Measurement \rightarrow Take action \rightarrow Update belief \rightarrow Take action.

2.3 Examples

Example:

1. **Given:**

- Now suppose Cavemen wants to feed child:
 - Cannot know satiety of child exactly.
 - Whether apple is edible or not must be inferred from senses.
- Possible observations for the apple:



Figure 3

- Possible states for the child's satiety:

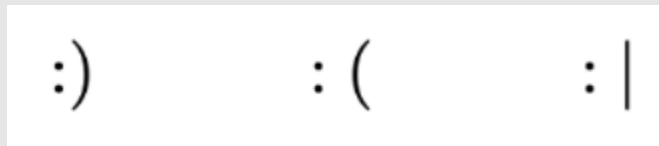


Figure 4

- Measurement distribution for the apple:












| |  |  |  |  |  |
|---|---|---|---|--|---|
|  | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |
|  | 0.2 | 0.6 | 0.2 | 0.0 | 0.0 |
|  | 0.0 | 0.3 | 0.4 | 0.3 | 0.0 |
|  | 0.0 | 0.0 | 0.0 | 0.2 | 0.8 |
|  | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |
|  | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 |

Figure 5: $m(o_1|s) = P(o_1|s)$

- $\sum = 1$ across the rows
- What is the probability of observing a certain state of the apple given the true state?
- Measurement distribution for child's satiety:







| | :) | :(| : |
|---|-----|-----|-----|
|  | 0.0 | 0.8 | 0.2 |
|  | 0.0 | 0.8 | 0.2 |
|  | 0.0 | 0.8 | 0.2 |
|  | 0.0 | 0.8 | 0.2 |
|  | 0.8 | 0.2 | 0.0 |
|  | 0.0 | 0.0 | 1.0 |

Figure 6: $m(o_2|s) = P(o_2|s)$

– $\sum = 1$ across the rows

– What is the probability of observing a certain state of the child given the true state?

- **Key:** Assume independence between the observations of the child's satiety and the apple's edibility:
 $P(o|s) = P(o_1|s) \cdot P(o_2|s)$.

2. Problem

- Initial distribution, $b_0(s_0)$ over states is uniform.
- Action sequence is $\langle a_1, a_2, a_3 \rangle = \langle \text{seed}, \text{wait}, \text{wait} \rangle$.
- Observation sequence is $\langle o_0, o_1, o_2, o_3 \rangle = \langle (:(\text{no apple}), :(\text{ga})), :(\text{ra}), :(|\text{ra})) \rangle$.
- Find state distribution: $b_3(s_3 | a_{1:3}, o_{0:3})$.












| |  |  |  |  |  | :) | :(| : |
|---|---|---|---|---|---|-----|-----|-----|
|  | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.8 | 0.2 |
|  | 0.2 | 0.6 | 0.2 | 0.0 | 0.0 | 0.0 | 0.8 | 0.2 |
|  | 0.0 | 0.3 | 0.4 | 0.3 | 0.0 | 0.0 | 0.8 | 0.2 |
|  | 0.0 | 0.0 | 0.0 | 0.2 | 0.8 | 0.0 | 0.8 | 0.2 |
|  | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.8 | 0.2 |
|  | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |

Figure 7

3. Solution:

Example: FINISH THE EXAMPLE

$$b_0(s_0 | o_0) = \frac{m(o_0 | s_0)}{\sum_s m(o_0 | s)} \quad (1)$$

$$b_t(s_t | a_{1:t}, o_{0:t}) = m(o_t | s_t) \sum_{s_{t-1}} p(s_t | s_{t-1}, a_t) b_{t-1}(s_{t-1} | a_{1:t-1}, o_{0:t-1}) \quad (2)$$

| s | $b_0(s)$ | $b_0(s o_0)$ | $b_1(s o_{0:1}, a_1)$ | $b_2(s o_{0:2}, a_{1:2})$ | $b_3(s o_{0:3}, a_{1:3})$ |
|---|----------|----------------|-------------------------|-----------------------------|-----------------------------|
| No meat | 1/6 | 0.4545 | | | |
| <ul style="list-style-type: none"> $\sum_s m((:, \text{no apple}) s) = (1.0)(0.8) + (0.2)(0.8) + (0.0)(0.8) + (0.0)(0.8) + (1.0)(0.8) + (1.0)(0.0) = 1.76$ $b_0(\text{No meat} (:, \text{no apple})) = \frac{(1.0)(0.8)}{1.76} = \frac{0.8}{1.76} = 0.4545$ | | | | | |
| Green apple | 1/6 | 0.0909 | | | |
| <ul style="list-style-type: none"> $b_0(\text{Green apple} o_0) = \frac{(0.2)(0.8)}{1.76} = \frac{0.16}{1.76} = 0.0909$ | | | | | |
| Red apple | 1/6 | 0 | | | |
| <ul style="list-style-type: none"> $b_0(\text{Red apple} o_0) = \frac{(0.0)(0.8)}{1.76} = \frac{0.0}{1.76} = 0$ | | | | | |
| Rotten apple | 1/6 | 0 | | | |
| <ul style="list-style-type: none"> $b_0(\text{Rotten apple} o_0) = \frac{(0.0)(0.8)}{1.76} = \frac{0.0}{1.76} = 0$ | | | | | |
| Meat | 1/6 | 0.4545 | | | |
| <ul style="list-style-type: none"> $b_0(\text{Meat} o_0) = \frac{(1.0)(0.8)}{1.76} = \frac{0.8}{1.76} = 0.4545$ | | | | | |
| Dead | 1/6 | 0 | | | |
| <ul style="list-style-type: none"> $b_0(\text{Dead} o_0) = \frac{(1.0)(0.0)}{1.76} = \frac{0.0}{1.76} = 0$ | | | | | |

3 Estimating the Optimal Quality Function

3.1 Estimating the Optimal Quality Function

Motivation: The agent need not know the model of the environment. However, it must actually make moves, even when learning.

If the agent doesn't have a model, it must estimate q^* , \mathcal{A}^* , and π^* .

Definition: When the environment is in state s , the agent can take an action a and:

- **Update \hat{q} :** $\hat{q}(s, a; t) \leftarrow (1 - \alpha)\hat{q}(s, a; t) + \alpha \left(r' + \gamma \max_{a'} \hat{q}(s', a'; t + 1) \right)$
 – $0 \leq \alpha \leq 1$: learning rate
- **Compute $\hat{\mathcal{A}}$:** $\hat{\mathcal{A}}(s; t) = \arg \max_{a' \in \mathcal{A}(s)} \hat{q}(s, a'; t)$
- **Compute $\hat{\pi}$:** $\hat{\pi}(a' | s; t) = 0 \ \forall a' \notin \hat{\mathcal{A}}(s; t)$

3.2 Exploration versus Exploitation

Motivation: To ensure \hat{q} converges to q^* and the agent's expected return is maximized, the agent must balance exploration and exploitation.

Definition:

- **Exploitation:** Choose the most promising actions based on current knowledge.
 – Use optimal policy: $\hat{\pi}(\cdot, \cdot; t)$
- **Exploration:** Choose the least tried actions to improve current knowledge.
 – Choose actions randomly

3.2.1 Simplified Case:

Example:

- **Given:** Assume the environment is stateless, but rewards are random.



Figure 8



Figure 9

- $\mu(a)$: expected reward for action a (unknown to the agent):
 – $0 \leq \mu(a) \leq 1$ for all a .

- **Best-case expected return:** (with $\gamma = 1$ under π^*) from transition t is:

$$u^*(t) := (T - t) \max_{a'} \mu(a')$$

where in this case:

$$\pi^*(a; t) = 0 \quad \text{if } a \notin \arg \max_{a'} \mu(a').$$

- **Estimation of $\mu(\cdot)$.** Since the agent does not have a model, it must estimate $\mu(\cdot)$.

The agent can take an action a and:

1. **Update** $n(\cdot)$ and $\hat{\mu}(\cdot)$:

$$n(a) \leftarrow n(a) + 1$$

$$\hat{\mu}(a) \leftarrow \left(1 - \frac{1}{n(a)}\right) \hat{\mu}(a) + \frac{1}{n(a)} r'$$

2. **Compute $\hat{\pi}$:**

$$\hat{\pi}(a; t) = 0 \quad \text{for all } a \notin \arg \max_{a'} \hat{\mu}(a').$$

- **Alternate Policies** We want to compare the expected return under various policies. The expected return from transition t under a policy ρ is:

$$u^\rho(t) := \mathbb{E}^\pi[G_t] = \sum_{a'} \rho(a'; t) (\mu(a') + u^\rho(t+1)).$$

3.3 Alternate Policies

Summary: To ensure the agent's expected return is maximized, the agent must strike a balance exploration and exploitation.

In the following cases, the expected return from transition t is

$$u^{\text{avg}}(t) \equiv \frac{T-t}{|\mathcal{A}|} \sum_a \mu(a)$$

We want to choose ρ so that $u^\rho > u^{\text{avg}}$.

| Policy | Function: |
|------------------------|---|
| Exploitation only | Choose a random action, same for all transitions |
| Exploration only | Choose a random action, different for each transition |
| Softmax | Apply a soft-max over \hat{u} $\rho(a; t) = \left[\sum_{a'} \exp \left(\frac{\hat{\mu}(a')}{\tau} \right) \right]^{-1} \exp \left(\frac{\hat{\mu}(a)}{\tau} \right)$ <ul style="list-style-type: none"> Choose a temperature value decrease with t. $\tau(t) \in [0, \infty), \tau \rightarrow 0$ |
| ϵ -greedy | Use $\hat{\pi}$ w/ prob. $1 - \epsilon$, otherwise take a random action $\rho(a; t) = \epsilon \frac{1}{ \mathcal{A} } + (1 - \epsilon) \hat{\pi}(a; t)$ <ul style="list-style-type: none"> Choose an exploration rate decrease w/ t. $\epsilon(t) \in [0, 1], \epsilon \rightarrow 0$ |
| Upper confidence bound | Choose the action with the highest $\text{ucb}(\cdot)$ $\rho(a; t) = 0 \text{ if } a \notin \arg \max_{a'} \text{ucb}(a'; t)$ <ul style="list-style-type: none"> Compute $\text{ucb}(\cdot)$ for each action. $\text{ucb}(a; t) = \hat{\mu}(a) + \sqrt{\frac{\ln t}{n(a)}}$ |

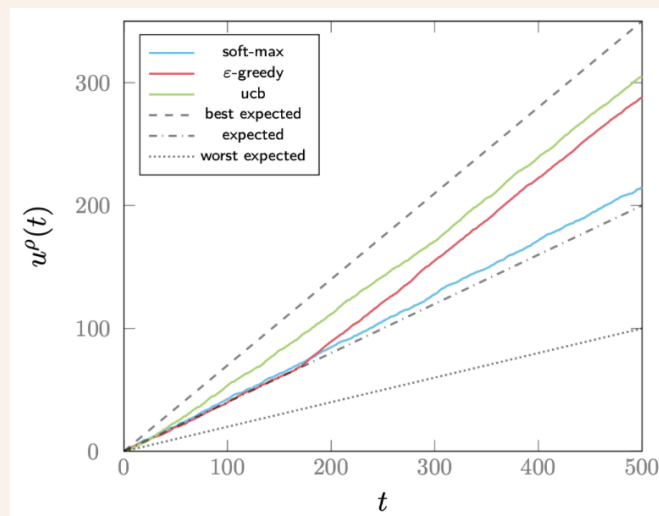


Figure 10