Chapter 3 - Finite Markov Decision Processes

Stéphane Liem NGUYEN

August 15, 2021

Exercises with (***corrected***) were corrected based on the Errata
These are my own answers and mistakes or errors are possible.

***Exercise 3.1* (p. 51)** Devise three example tasks of your own that fit into the MDP framework, identifying for each its states, actions, and rewards. Make the three examples as *different* from each other as possible. The framework is abstract and flexible and can be applied in many different ways. Stretch its limits in some way in at least one of your examples.

***Exercise 3.2* (p. 51)** Is the MDP framework adequate to usefully represent *all* goal-directed learning tasks? Can you think of any clear exceptions ?

***Exercise 3.3* (p. 51)** Consider the problem of driving. You could define the actions in terms of the accelerator, steering wheel, and brake, that is, where your body meets the machine. Or you could define them farther out—say, where the rubber meets the road, considering your actions to be tire torques. Or you could define them farther in—say, where your brain meets your body, the actions being muscle twitches to control your limbs. Or you could go to a really high level and say that your actions are your choices of *where* to drive. What is the right level, the right place to draw the line between agent and environment? On what basis is one location of the line to be preferred over another? Is there any fundamental reason for preferring one location over another, or is it a free choice?

***Exercise 3.4* (p. 53)** Give a table analogous to that in Example 3.3 (recycling robot), but for $p(s', r|s, a)$. It should have columns for s, a, s', r, and $p(s', r|s, a)$, and a row for every 4-tuple for which $p(s', r|s, a) > 0$.

***Exercise 3.5* (p. 55)** The equations in Section 3.1 (Agent-Environment Interface) are for the continuing case and need to be modified (very slightly) to apply to episodic tasks. Show that you know the modifications needed by giving the modified version of (3.3).

Based on the book at page 54: "Episodes can all be considered to end in the same terminal state, with different rewards for the different outcomes. [...] In episodic tasks we sometimes need to distinguish the set of all nonterminal states, denoted $\mathcal{S}$, from the set of all states plus the terminal state, denoted $\mathcal{S}^+$.", we can just add the terminal state to set of possible next states.

We can rewrite the modified version of (3.3) as follows:

$$\sum_{s' \in \mathcal{S}^+} \sum_{r \in \mathcal{R}} p(s', r | s, a) = 1, \quad \forall s \in \mathcal{S}, \forall a \in \mathcal{A}(s) \tag{1}$$

where s' can be the terminal state.

***Exercise 3.6* (p. 56)** Suppose you treated pole-balancing as an episodic task but also used discounting, with all rewards zero except for -1 upon failure. What then would the return be at each time? How does this return differ from that in the discounted, continuing formulation of this task?

The return for the discounted, *episodic* formulation would be at each time

$$\begin{aligned} G_t &\doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots + \gamma^{T-1} R_T \\ &= -\gamma^{T-1} \end{aligned} \tag{2}$$

while the return for the discounted, *continuing* formulation would be at each time

$$G_t \doteq \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} = -\sum_{k=0}^{\infty} \gamma^k \mathbf{1}_{\text{failure}} \tag{3}$$

***Exercise 3.7* (p. 56)** Imagine that you are designing a robot to run a maze. You decide to give it a reward of +1 for escaping from the maze and a reward of zero at all other times. The task seems to break down naturally into episodes—the successive runs through the maze—so you decide to treat it as an episodic task, where the goal is to maximize expected total reward (3.7). After running the learning agent for a while, you find that it is showing no improvement in escaping from the maze. What is going wrong? Have you effectively communicated to the agent what you want it to achieve?

We can just recall that the return G_t from (3.7) (for episodic tasks) is just the sum of the future rewards.

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T \tag{4}$$

Because of how we defined the rewards and because we use the undiscounted formulation of the return, G_t is either 1 or 0 (if a game can terminate without escape from the maze) and what we try to maximize is the expected amount of games where we escape from the maze. For a given state, its value (not estimation) would be the average amount of future games where we will escape if we follow some policy.

Work In Progress

***Exercise 3.8* (p. 56)** Suppose $\gamma = 0.5$ and the following sequence of rewards is received $R_1 = -1, R_2 = 2, R_3 = 6, R_4 = 3$, and $R_5 = 2$, with $T = 5$. What are $G_0, G_1, \ldots, G_5$? Hint: Work backwards.

- $G_5 = 0$

- $G_4 = R_5 + \gamma \cdot G_5 = 2 + 0.5 \cdot 0 = 2$
- $G_3 = R_4 + \gamma \cdot G_4 = 3 + 0.5 \cdot 2 = 4$
- $G_2 = R_3 + \gamma \cdot G_3 = 6 + 0.5 \cdot 4 = 8$
- $G_1 = R_2 + \gamma \cdot G_2 = 2 + 0.5 \cdot 8 = 6$
- $G_0 = R_1 + \gamma \cdot G_1 = -1 + 0.5 \cdot 6 = 2$

***Exercise 3.9* (p. 56)** Suppose $\gamma = 0.9$ and the reward sequence is $R_1 = 2$ followed by an infinite sequence of 7s. What are G_1 and G_0?

Instead of having an episodic task as the previous exercise, we use the discounted, continuing formulation of the return

$$G_t \doteq \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{5}$$

G_1 is directly given by

$$G_1 \doteq \sum_{k=0}^{\infty} \gamma^k R_{k+2} = 7 \sum_{k=0}^{\infty} \gamma^k = 7 \cdot \frac{1}{1-\gamma} = 70 \tag{6}$$

and $G_0 = R_1 + \gamma \cdot G_1 = 2 + 0.9 \cdot 70 = 65$

***Exercise 3.10* (p. 56)** Prove the second equality in (3.10).

We want to prove that if the rewards at all time steps are constant +1, then the return is $\frac{1}{1-\gamma} < \infty$ for $0 \leq \gamma < 1$

$$G_t = \sum_{k=0}^{\infty} \gamma^k = \frac{1}{1-\gamma} \tag{7}$$

We first pass the $1 - \gamma$ on the other side and do some manipulations to complete the proof

$$(1-\gamma) \sum_{k=0}^{\infty} \gamma^k = 1 \iff \sum_{k=0}^{\infty} \gamma^k - \sum_{k=1}^{\infty} \gamma^k = \gamma^0 + 0 = 1 \tag{8}$$

***Exercise 3.11* (p. 58)** If the current state is S_t, and actions are selected according to a stochastic policy π, then what is the expectation of R_{t+1} in terms of π and the four-argument dynamics function p (3.2)?

***Exercise 3.12* (p. 58)** Give an equation for v_π in terms of q_π and π.

***Exercise 3.13* (p. 58)** Give an equation for q_π in terms of v_π and the four-argument p.

Exercise 3.14 **(p. 60)** The Bellman equation (3.14) must hold for each state for the value function v_π shown in Figure 3.2 (right) of Example 3.5 (Gridworld). Show numerically that this equation holds for the center state, valued at $+0.7$, with respect to its four neighboring states, valued at $+2.3$, $+0.4$, -0.4, and $+0.7$. (These numbers are accurate only to one decimal place.)

Exercise 3.15 **(p. 61)** In the gridworld example, rewards are positive for goals, negative for running into the edge of the world, and zero the rest of the time. Are the signs of these rewards important, or only the intervals between them? Prove, using (3.8), that adding a constant c to all the rewards adds a constant, v_c, to the values of all states, and thus does not affect the relative values of any states under any policies. What is v_c in terms of c and γ?

Exercise 3.16 **(p. 61)** Now consider adding a constant c to all the rewards in an episodic task, such as maze running. Would this have any effect, or would it leave the task unchanged as in the continuing task above? Why or why not? Give an example.

Exercise 3.17 **(p. 61)** What is the Bellman equation for action values, that is, for q_π? It must give the action value $q_\pi(s, a)$ in terms of the action values, $q_\pi(s', a')$, of possible successors to the state–action pair (s, a). Hint: The backup diagram in Figure 1 corresponds to this equation. Show the sequence of equations analogous to (3.14), but for action values.

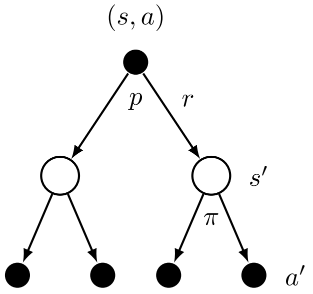

Figure 1: q_π backup diagram

Exercise 3.18 **(p. 62)** The value of a state depends on the values of the actions possible in that state and on how likely each action is to be taken under the current policy. We can think of this in terms of a small backup diagram rooted at the state and considering each possible action:

Give the equation corresponding to this intuition and diagram for the value at the root node, $v_\pi(s)$, in terms of the value at the expected leaf node, $q_\pi(s, a)$, given $S_t = s$. This equation should include an expectation conditioned on following the policy, π. Then give a second equation in which the expected value is written out explicitly in terms of $\pi(a|s)$ such that no expected value notation appears in the equation.

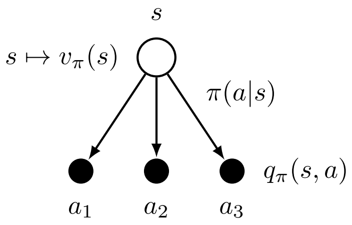

Figure 2: Small backup diagram showing v_π in terms of q_π and the policy π

***Exercise 3.19* (p. 62)** The value of an action, $q_\pi(s,a)$, depends on the expected next reward and the expected sum of the remaining rewards. Again we can think of this in terms of a small backup diagram, this one rooted at an action (state-action pair) and branching to the possible next states:

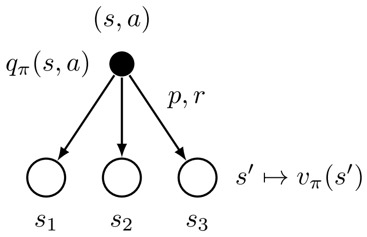

Figure 3: Small backup diagram showing q_π in terms of v_π and the dynamics function p

Give the equation corresponding to this intuition and diagram for the action value, $q_\pi(s,a)$, in terms of the expected next reward, R_{t+1}, and the expected next state value, $v_\pi(St+1)$, given that $S_t = s$ and $A_t = a$. This equation should include an expectation but *not* one conditioned on following the policy. Then give a second equation, writing out the expected value explicitly in terms of $p(s',r|s,a)$ defined by (3.2), such that no expected value notation appears in the equation.

***Exercise 3.20* (p. 66)** Draw or describe the optimal state-value function for the golf example.

***Exercise 3.21* (p. 66)** Draw or describe the contours of the optimal action-value function for putting, $q_*(s, \textbf{putter})$, for the golf example.

***Exercise 3.22* (p. 66)** Consider the continuing MDP shown in Figure 4. The only decision to be made is that in the top state, where two actions are available, **left** and **right**. The numbers show the rewards that are received deterministically after each action. There are exactly two deterministic policies, π_{left} and π_{right}. What policy is optimal if $\gamma = 0$? If $\gamma = 0.9$? If $\gamma = 0.5$?

***Exercise 3.23* (p. 67)** Give the Bellman equation for q_* for the recycling robot.

***Exercise 3.24* (p. 67)** Figure 3.5 (Gridworld) gives the optimal value of the best state of the gridworld as 24.4, to one decimal place. Use your knowledge of the optimal policy and (3.8) to express this value symbolically, and then to compute it to three decimal places.

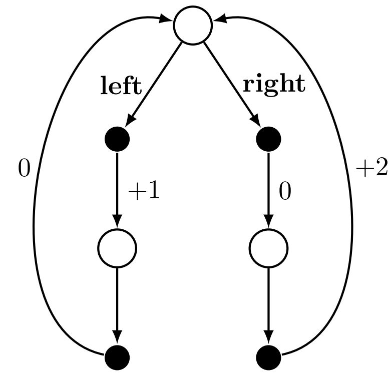

Figure 4: MDP transition graph

***Exercise 3.25* (p. 67)** Give an equation for v_* in terms of q_*.

From page 63 of the book, "Intuitively, the Bellman optimality equation expresses the fact that the value of a state under an optimal policy must equal the expected return for the best action from that state"

$$v_*(s) = \sum_a \pi(a|s) q_*(s,a) = \max_a q_*(s,a) \tag{9}$$

where $a \in \mathcal{A}(s)$

***Exercise 3.26* (p. 67)** Give an equation for q_* in terms of v_* and the four-argument p.

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a) \left[r + \gamma v_*(s')\right] \tag{10}$$

***Exercise 3.27* (p. 67)** Give an equation for π_* in terms of q_*.

From page 68 of the book, "Any policy that is greedy with respect to the optimal value functions must be an optimal policy." and from page 65, "With q_* the agent does not even have to do a one-step-ahead search: for any state s, it can simply find any action that maximizes $q_*(s,a)$", translating it into a formula gives us one possible optimal value function:

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \arg\max\limits_{a'} q_*(s,a') \\ 0 & \text{else} \end{cases} \tag{11}$$

if we suppose that the argmax gives one of the maximizing actions if many actions achieve the maximum.

***Exercise 3.28* (p. 67)** Give an equation for π_* in terms of v_* and the four-argument p.

By using the previous exercise and Exercise 3.26 where

$$q_*(s,a') = \sum_{s',r} p(s',r|s,a') \left[r + \gamma v_*(s')\right] \tag{12}$$

we get

$$\pi_*(a|s) = \begin{cases} 1 & \text{if } a = \arg\max\limits_{a'} \sum_{s',r} p(s',r|s,a') \left[r + \gamma v_*(s')\right] \\ 0 & \text{else} \end{cases} \tag{13}$$

***Exercise 3.29* (p. 67)** Rewrite the four Bellman equations for the four value functions (v_π, v_*, q_π, and q_*) in terms of the three argument function p (3.4) and the two-argument function r (3.5).

Let's just first recall how to obtain the three argument function p and the two-argument function r from the 4 argument function p describing the MDP dynamics:

$$p(s'|s,a) \doteq \sum_{r\in\mathcal{R}} p(s',r|s,a) \tag{14}$$

$$r(s,a) \doteq \mathbb{E}[R_{t+1}|S_t = s, A_t = a] = \sum_{r\in\mathcal{R}} r \cdot p(r|s,a) = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p(s',r|s,a) \tag{15}$$

We can also recall the 4 Bellman equations for the four value functions:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\,[r + \gamma v_\pi(s')] \tag{16}$$

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)\,[r + \gamma v_*(s')] \tag{17}$$

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a) \left[r + \gamma \sum_{a'} \pi(a'|s') q_\pi(s',a') \right] \tag{18}$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a) \left[r + \gamma \max_{a'} q_*(s',a') \right] \tag{19}$$

where $a \in \mathcal{A}(s)$, s' and $s \in \mathcal{S}$ and $r \in \mathcal{R}$. By replacing all the 4 Bellman equations with (3.4) and (3.5) from the book:

$$v_\pi(s) = \sum_a \pi(a|s) \left[r(s,a) + \gamma \sum_{s'} p(s'|s,a) v_\pi(s') \right] \tag{20}$$

$$v_*(s) = \max_a \left\{ r(s,a) + \gamma \sum_{s'} p(s'|s,a) v_*(s') \right\} \tag{21}$$

$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \sum_{a'} \pi(a'|s') q_\pi(s',a') \tag{22}$$

$$q_*(s,a) = r(s,a) + \gamma \sum_{s'} p(s'|s,a) \max_{a'} q_*(s',a') \tag{23}$$

As remark, the Bellman equation with two-argument function r and three-argument function p could be used directly to obtain the last equality for $v_*(\mathbf{h})$ or the equation for $v_*(\mathbf{l})$ in Example 3.9 on the Bellman Optimality Equation for Recycling Robot (p. 65).

Proof of the formula used in the first equality of Example 3.9 **Bellman Optimality Equation for Recycling Robot (p. 65)** We want to prove that

$$v_*(s) = \max_a \sum_{s'} p(s'|s,a)\left[r(s,a,s') + \gamma v_*(s')\right] \tag{24}$$

Recall the definition of $r(s,a,s')$ that is the expected reward for state–action–next-state triples ((3.6) in the book at page 49)

$$\begin{aligned} r(s,a,s') \doteq \mathbb{E}[R_{t+1}|S_t = s, A_t = a, S_{t+1} = s'] &= \sum_{r\in\mathcal{R}} r \cdot p(r|s,a,s') \\ &= \sum_{r\in\mathcal{R}} r \cdot \frac{p(s',r|s,a)}{p(s'|s,a)} \end{aligned} \tag{25}$$

And this gives us $\sum_{s'} p(s'|s,a) \cdot r(s,a,s') = \sum_{s',r} r \cdot p(s',r|s,a)$ and we substitute it in the Bellman equation for v_*.

$$\begin{aligned} v_*(s) &= \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_*(s')\right] \\ &= \max_a \left\{ \sum_{s'} p(s'|s,a) \cdot r(s,a,s') + \gamma \sum_{s',r} p(s',r|s,a) v_*(s') \right\} \\ &= \max_a \sum_{s'} p(s'|s,a)\left[r(s,a,s') + \gamma v_*(s')\right] \end{aligned} \tag{26}$$

This proves the formula.