Attempts to exercise in Reinforcement Learning book Chapter 4

Mengliao Wang

July 9, 2017

Exercise 4.1:

According to $q_\pi(s,a) = r(s,a,s') + \gamma \sum_{s',r} p(r,s' \mid s,a) v_\pi(s')$

We have $q_\pi(11, down) = -1 + v(T) = -1$.

Similarly we have $q_\pi(7, down) = -1 + v(11)$. Here $v(11)$ would take iterations to calculate, but given enough runs, it will converge to -14 as shown in Figure 4.1. Thus the optimal action value $q_\pi(7, down) = -15$.

Exercise 4.2:

If the dynamic of original states are unchanged, then $v_\pi(s), s = 1, 2, .., 14$ will not be changed. So we have value function

$$\begin{aligned} v_\pi(15) &= 0.25 \times [-1 + v_\pi(12)] + 0.25 \times [-1 + v_\pi(13)] + 0.25 \times [-1 + v_\pi(14)] + 0.25 \times [-1 + v_\pi(15)] \quad (1) \\ &= 0.25 \times (-23) + 0.25 \times (-21) + 0.25 \times (-15) + 0.25 \times [-1 + v_\pi(15)] \quad (2) \end{aligned}$$

By solving the equation above we have $v_\pi(15) = -20$.

If grid 13 will go down to the new grid 15, then we have the following equations:

$$\begin{aligned} v_\pi(15) &= 0.25 \times [-1 + v_\pi(12)] + 0.25 \times [-1 + v_\pi(13)] + 0.25 \times [-1 + v_\pi(14)] + 0.25 \times [-1 + v_\pi(15)] \quad (3) \\ &= 0.25 \times (-23) + 0.25 \times (-1 + v_\pi(13)) + 0.25 \times (-15) + 0.25 \times [-1 + v_\pi(15)] \quad (4) \\ &= -10 + 0.25 \times [v_\pi(13) + v_\pi(15)] \quad (5) \\ v_\pi(13) &= 0.25 \times [-1 + v_\pi(12)] + 0.25 \times [-1 + v_\pi(9)] + 0.25 \times [-1 + v_\pi(14)] + 0.25 \times [-1 + v_\pi(15)] \quad (6) \\ &= 0.25 \times (-23) + 0.25 \times (-21) + 0.25 \times (-15) + 0.25 \times [-1 + v_\pi(15)] \quad (7) \\ &= -15 + 0.25 \times v_\pi(15) \quad (8) \end{aligned}$$

By resolving the equations above we still have $v_\pi(15) = v_\pi(13) = -$.

Exercise 4.3:

Equation 4.3, 4.4 for $q_\pi(s,a)$ is:

$$\begin{aligned} q_\pi(s,a) &= \mathbb{E}[R_{t+1} + \gamma v_\pi(s') \mid S_t = s, A_t = a] \quad (9) \\ &= \mathbb{E}[R_{t+1} + \gamma \mathbb{E}_\pi[q_\pi(s',a')] \mid S_t = s, A_t = a, S_{t+1} = s'] \quad (10) \\ &= r + \gamma \sum_{s',r} p(s',r \mid s,a) \sum_{a'} \pi(a' \mid s') q_\pi(s',a') \quad (11) \end{aligned}$$

So we can have the approximation for $q_\pi(s,a)$ as:

$$q_{k+1}(s,a) = r + \gamma \sum_{s',r} p(s',r \mid s,a) \sum_{a'} \pi(a' \mid s') q_k(s',a') \quad (12)$$

Exercise 4.4:

First of all, if a policy will have non-zero probablities for all the actions given any state, then we will not have this issue. The negative infinity value only happens when certain states consist of a subset S', with no transition between the remaining states S-S'. To avoid this issue we can introduce some of the approaches explained in Chapter 2, such as an exploration rate ϵ.

To be more specific, in the algorithm while update $V(s)$, we can update it to:

$$V(s) \leftarrow \begin{cases} \sum_a \pi(a|s) \sum_{s',r} p(s',r \mid s,a)[r+\gamma V(s')], & \text{with probability } 1-\epsilon \\ \sum_a \pi'(a|s) \sum_{s',r} p(s',r \mid s,a)[r+\gamma V(s')], & \text{with probability } \epsilon \end{cases}$$

Here π' is any non-zero policy, such as random equiprobable policy.

Exercise 4.5:

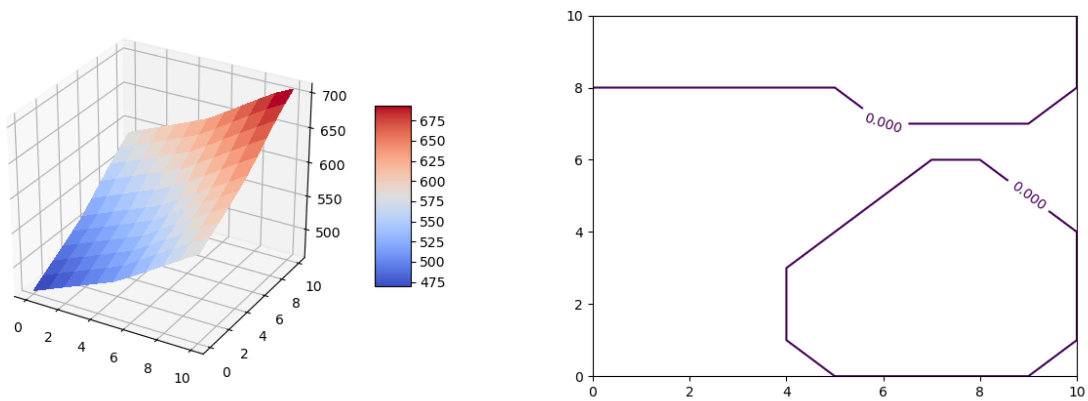

Figure 1: Value and Policy for Exercise 4.5 solution with 10 maximum cars per location, and 5 maximum free parking

Python code attached. I cut the maximum number of cars to be 10 and the parking limit to be 5. Figure 1 is the optimal value and policy for the problem.

Exercise 4.6:

1. Initialization:
$Q(s,a) \in \mathbb{R}$ and $\pi(s) \in A(s)$ arbitrarily for all $s \in S$ and $a \in A(s)$

2. Policy Evaluation:
repeat
  $\Delta \leftarrow 0$
  for Each $s \in S$: **do**
    for Each $a \in A(s)$: **do**
      $q \leftarrow Q(s,a)$
      $Q(s,a) \leftarrow r(s,a) + \gamma \sum_{s',r} p(s',r|s,a) \sum_{\pi(s')} \pi(s')Q(s',\pi(s'))$
      $\Delta \leftarrow \max(\Delta, |q - Q(s,a)|)$
until $\Delta < \theta$ (a small positive number)

3. Policy Improvement:
policy-stable $\leftarrow$ *true*
for each $s \in S$: **do**
  old-action $\leftarrow \pi(s)$
  $\pi(s) \leftarrow \arg\max_a Q(s,a)$
  if *old-action* $\neq \pi(s)$ **then**
    policy-stable $\leftarrow$ *false*
if *policy-stable* **then**
  stop and return $Q \approx q_*$
else
  go to 2

Exercise 4.7:

For step 3 in the algorithm, For each state instead of assigning a single value, we will need to assign probability for each state-action pair $\pi(s,a)$. For action that maximize $\sum_{s',r} p(s',r|s,a)[r+\gamma V(s')]$, $\pi(s,a) = 1-\epsilon+\dfrac{\epsilon}{|A(s)|}$. For all the other actions $\pi(s,a) = \dfrac{\epsilon}{|A(s)|}$. Everything else would stay the same, except that the old-action would be a array with all the probabilities under s, i.e. $\pi(s, ...)$

For step 2 in the step that update $V(s)$, we need to loop through all the actions under state s, and reset $V(s)$ to be 0 after $v \leftarrow V(s)$. Then for each action loop, we define $V(s) \leftarrow V(s)+\pi(s,a)\sum_{s',r} p(s',r|s,a)[r+\gamma V(s')]$. Everything else stays the same.

For step 1 $V(s)$ does not change. But we need to make $\pi(s)$ to be a matrix $\pi(s,a), a \in A(s)$. Also for initialization we must make sure $\pi(s,a) > \epsilon, \forall s, a$ and $\sum_a (s,a) = 1, \forall s \in S$

Exercise 4.8:

What caused this form is the fact that the value function for mid point $v(50)$ is slightly higer than the regression curve, due to the fact that the value $v(50)$ is deterministic of 0.4. Thus the other states will prefer to take actions to move to this state, e.g. at state 51 it will take action = 1, at state 25/75 it will take action = 25, etc.. Based on the same reason state 50 itself will prefer deterministic destination states, i.e. state 0 and 100. Thus it will bet all on the flip.

Exercise 4.9:

Python program attached. I was unable to reproduce the optimal policy shown in Figure 4.3 in the book for $p_h = 0.4$, But after debugging I believe that is due to the approximation. Figure 2 and Figure 3 show the final optimal value and policy for $p_h = 0.25$ and $p_h = 0.55$ respectively.

Exercise 4.10:

Obviously we have:

$$q_{k+1}(s,a) = \sum_{s',r} p(s',r \mid s,a)[r+\gamma \max_a q_k(s,a)] \tag{13}$$

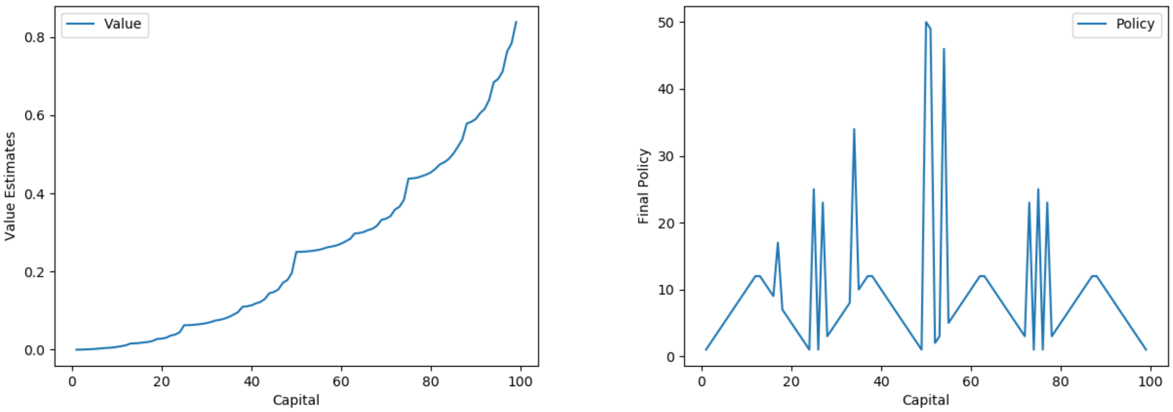

Figure 2: Final value and policy for Gambler with $p_h = 0.25$

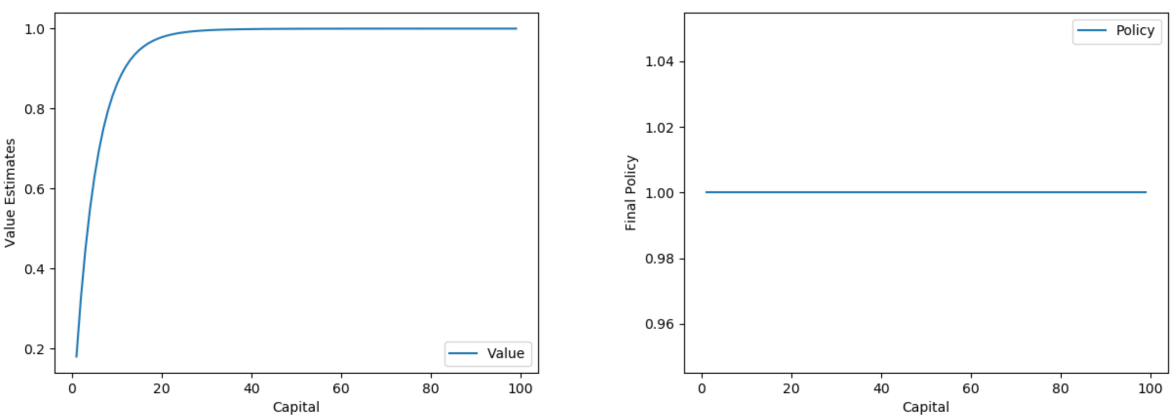

Figure 3: Final value and policy for Gambler with $p_h = 0.55$