Solutions to Reinforcement Learning by Sutton Chapter 6

Yifan Wang

Aug 2019

Exercise 6.1

By the setup, $\delta_t \doteq R_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)$.
And the update of TD is $V_{t+1}(S) = V_t(S) + \alpha[R_{t+1} + \gamma V_t(S') - V_t(S)]$.
With these new definitions in mind we continue as follows:

$$\begin{aligned}
G_t - V_t(S_t) &= R_{t+1} + \gamma G_{t+1} - V_t(S_t) + \gamma V_t(S_{t+1}) - \gamma V_t(S_{t+1}) \\
&= \delta_t + \gamma(G_{t+1} - V_t(S_{t+1})) \\
&= \delta_t + \gamma(G_{t+1} - V_{t+1}(S_{t+1})) + \gamma u_{t+1} \\
&(\text{for } u_{t+1} = \alpha[R_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)]) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2(G_{t+2} - V_{t+2}(S_{t+2})) + \gamma u_{t+1} + \gamma^2 u_{t+2} \\
&= \sum_{k=t}^{T-1} \left[\gamma^{k-t}\delta_k + \gamma^{k-t+1}u_{k+1}\right]
\end{aligned}$$

Thus, the additional amount that must be added is $\sum_{k=t}^{T-1} -\gamma^{k-t+1}u_{k+1}$
■

Exercise 6.2

The scenario book mentioned is very efficient for TD because states of the highway part are not changed, and we can still have a solid belief of its state value. Utilizing TD will accelerate our adjustment of our new states. Each updated state will then accelerate the one prior to it. Monte Carlo, on the other hand, has to go to the end and

bring average improvement of all states, especially those high way parts, which do not need too much adjustment at all but will still be fluctuated by the Monte Carlo method.

Another situation where TD is much more efficient is when we are very difficult to make to the terminal state. Even if under horizon method in which we do not consider state value that decayed too small, the original Monte Carlo is still not efficient, or even practical.

Finally, TD can treat experimental problem well as the 6.2 will mention.

■

Exercise 6.3

It tells us that the first episode ends at state A. We do not have any information about how the middle transition is though. Because those rewards are 0 and γ is 1 and because they are all initialized as 0.5, we can only expect the state value change be nonzero at state A or E. It is changed by the exact value $0.5 * (-\alpha)$ and thus equals to -0.05.

■

Exercise 6.4

No. Sufficiently small $alpha$ is the convergence requirement for both TD and Monte Carlo. Thus, one could say small α in TD and Monte Carlo is always better than large ones in the long run and is reaching lowest error it can reach in the limit. As a result, the shown α has already touched the limit of each method's performance. So we can conclude there is no magic fixed α that could make significant improvement. On the other hand, one could expect a changing α

or changing weight for different states will be handy, and it will be covered in much later chapters.

■

Exercise 6.5

First, α is not sufficiently small enough. Second, the initialization brings all same initial value to the state and it may not be so ideal. I cannot give exact proof but those initialization brings a linear-ish relations about the updates during the transition, may result the over-estimate around the terminal state.

■

Exercise 6.6

One way is using computation power, like DP to accurately calculate the result. Since we do not have policy, it will only result a long but precise state evaluation.

Another approach, however, lies in the Mathematics. Consider State E, if we start at E, what is the probability of going into the left end $P_E(L)$and what is the probability of going into the right end$P_E(R)$?

$$\begin{aligned} P_E(R) &= 1 - P_E(L) \\ &= 1 - P_E(D) * P_D(L) \\ &= 1 - P_E(D) * [P_D(C) * P_C(L) + P_D(E) * P_E(L)] \end{aligned}$$

But, we know $P_C(L) = 0.5$ due to symmetry. We also have knowledge about these: $P_E(D) = 0.5$, $P_D(C) = 0.5$, plugin we have:

$$P_E(R) = 1 - 0.5 * [0.5 * 0.5 + 0.5 * P_E(L)]$$

Thus:

$$P_E(L) = 0.5 * [0.5 * 0.5 + 0.5 * P_E(L)]$$

We have $P_E(L) = \frac{1}{6}$.
Similarly, one can easily hacked the rest.

I think the book uses this method because it is simple and mathematically accurate.

■

Exercise 6.7

See page 110 update part.Replace G with V with appropriate suffix t will do. ■

Exercise 6.8

It is almost trivial because the only thing we changed is notation. Changing G to $Q*$ and V to Q with appropppriate parameter is sufficient.

$$\begin{aligned} G(S_t, A_t) - Q(S_t, A_t) &= R_{t+1} + \gamma G(S_{t+1}, A_{t+1}) - Q(S_t, A_t) + \gamma Q(S_{t+1}, A_{t+1}) - \gamma Q(S_{t+1}, A_{t+1}) \\ &= \delta_t + \gamma[G(S_{t+1}, A_{t+1}) - Q(S_{t+1}, A_{t+1})] \\ &= \delta_t + \gamma\delta_{t+1} + \gamma^2[G(S_{t+2}, A_{t+2}) - Q(S_{t+2}, A_{t+2})] \\ &= \sum_{k=t}^{T-1} \gamma^{k-t}\delta_k \end{aligned}$$

■

Exercise 6.9

I will skip the programming exercise for now until I have a collaboration.

■

Exercise 6.10

I will skip the programming exercise for now until I have a collaboration.

■

Exercise 6.11

It is not straightforward. Usually off-policy will include policy weighting or, as later we will see, a tree method to consider expectation. However, without these, Q-learning is still an off-policy method because it does NOT take the current policy into consideration at all. ANY policy will only choose the A from S, and Q-learning will always evaluate the states using greed method, e.g, maximize the very next movement. Since it does not necessarily update the policy and never care about it when evaluating, we call it off-policy method. ■

Exercise 6.12

They seem to be the same method in general. But it will be different in occasional situation. Consider $S = S'$ during some steps. During update, SARSA will use the previous greedy choice A' made from S', and advance to $Q(S', A')$. But Q-learning will use the updated Q to re-choose greedy A^*. Because A' may not be the same action as the new greedy choice A^* that will maximize Q(S',A*), these two methods are different.

■

Exercise 6.13

It is trivial but very tedious to write out. Add additional action selection part and then change all argmax with A'. Thus I will skip this part. ■

Exercise 6.14

We can focus the afterstates about parking numbers at each day's morning. It will be the whole result of yesterday's actions. It will decrease the computation because we have many different ways of arranging cars but we will end up next morning the same number of cars at each locations. ■