Chapter 6

- Temporal-difference learning is a combination of Monte Carlo methods and dynamic programming methods.

- Nonstationary MC method:

(6.1) $$V(S_t) \leftarrow V(S_t) + \alpha\left[G_t - V(S_t)\right]$$

Simple TD(0) method:

(6.2) $$V(S_t) \leftarrow V(S_t) + \alpha\left[R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right]$$

MC $\Rightarrow G_t$

TD $\Rightarrow R_{t+1} + \gamma V(S_{t+1})$

- TD method bases update in part on an existing estimate, therefore it is a bootstrapping method.

(6.3) - $$V_\pi(s) = E_\pi\left[G_t \mid S_t = s\right]$$

$$= E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s\right]$$

$$= E_\pi\left[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2} \mid S_t = s\right]$$

(6.4) $$= E_\pi\left[R_{t+1} + \gamma V_\pi(S_{t+1}) \mid S_t = s\right]$$

- TD methods have an advantage over DP methods in that they do not require a model of the environment.

- TD methods have an advantage over MC methods because they can be implemented in an incremental fashion. Estimates using TD methods can be updated each time step instead of waiting until the end of an episode.

- TD methods usually converge faster than constant-α MC methods.

- Sarsa - On-Policy TD Control

(6.5) $$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\right]$$

- Q-Learning - off-Policy TD Control

(6.6) $$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right]$$

- Afterstates - An action is taken based on the current state, but the resulting next state that an action can be taken from depends on an action taken by an opponent (the environment).

Exercise 6.1

After moving to a new building, the old values learned for the TD method are still going to be accurate for the part of the journey that remains the same. Since the MC method is episodic, the outcome is based on the entire trip and not the individual legs of the drive home.

Exercise 6.2

The first episode ended with a transition $A \rightarrow$ Term. On the first episode, only a transition from $A \rightarrow$ Term or $E \rightarrow$ Term results in a change to the action-value estimate because the initial estimates are all the same and $R=0$ for all non-terminal transitions.

$V(S') = V(S)$ when $B \rightarrow A$:

$$V(S) \leftarrow 0.5 + 0.1\left[0 + 0.5 - 0.5\right] = 0.5$$

$V(S') = 0$ when $A \rightarrow$ Term:

$$V(S) \leftarrow 0.5 + 0.1\left[0 + 0 - 0.5\right] = 0.45$$

Exercise 6.3

TD works out better regardless of the step size parameter. For both TD and MC methods, higher values of α cause the error to converge faster, but it converges to a value that is less accurate. Lower values of α converge slower but give more accurate results. The TD method can converge more quickly and more accurately than the MC method.

Exercise 6.4

The first few episodes drive the initial estimates closer to the true value. For larger α, however, the estimates converge to values that are not as accurate. It does not always occur. If the initial values are closer to the true values, the error goes up and converges at a less than optimal value. If the initial values are wildly inaccurate, the error drops sharply and then flattens out.

Exercise 6.5

The true values for each state could be computed by:

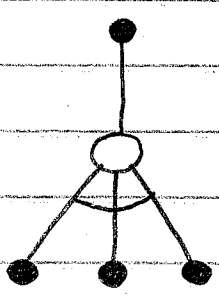
1. Solving the Bellman equations $V_\pi(s)$ for each state.

2. Using Monte Carlo methods with exploring starts.

My guess is that the values were computed by solving the Bellman equations because this would yield the exact values as fractions instead of a stochastic estimate that would be given using a Monte Carlo method.

Exercise 6.8

Q-learning



Sarsa

Exercise 6.9

Q-learning is off-policy because the update to the action-value is based on the action that has the highest value for the next state instead of the action that would be chosen by following the policy.

Exercise 6.10

The new method is an on-policy method because it chooses the action-value for the next state based on the action chosen by following the stochastic policy π.

The backup diagram would be the same as the backup diagram for Sarsa.

This method would work the same as Sarsa provided that $\pi(a|s)$ is the same ε-greedy policy used by the Sarsa method.

Exercise 6.11

Instead of computing state values based on the probability distribution $p(s', r | s, a)$ of pickups and dropoffs, an afterstate value function could be used based on the state of the parking lot (how many cars are on each lot) at the end of the day.

A reformulation in terms of afterstates would be more efficient and speed convergence because a given afterstate (at the end of the day) could be the result of several different beginning states (at the start of the day). For example, two different beginning states at the start of the day can have the same exact afterstate at the end of the day.