Temporal Difference Learning

Ashwini Venkatesh - av28895
ashuven63@utexas.edu

Introduction

This programming assignment looks into two temporal difference learning methods - SARSA and Q-Learning. SARSA is an on policy control method. Q-Learning is an off-policy control method.

Approach

The environment chosen is a cliff when the agent must learn to navigate from the start goal to the end goal without falling off the cliff. This is a standard undiscounted, episodic task, with start and goal states, and the usual actions causing movement up, down, right, and left. Reward is 1 on all transitions except those into the region marked “The Cliff.” Stepping into this region incurs a reward of 100 and sends the agent instantly back to the start [1].

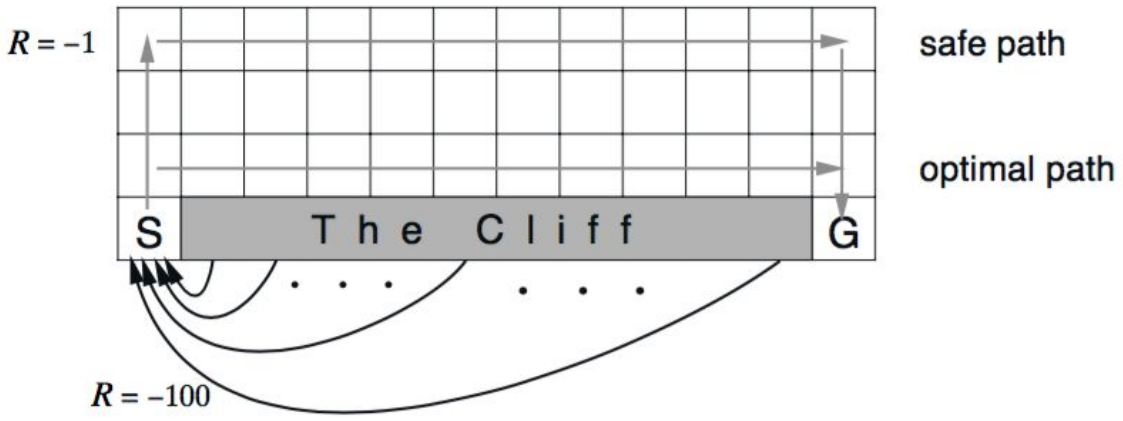

Figure 1 : Environment used in the task

The algorithm implemented for SARSA is as follows:

Sarsa: An on-policy TD control algorithm

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Choose A from S using policy derived from Q (e.g., ϵ-greedy)
  Repeat (for each step of episode):
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma Q(S',A') - Q(S,A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Figure 2 : Sarsa Algorithm

The algorithm implemented for Q-Learning is as follows:

Q-learning: An off-policy TD control algorithm

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S,A) \leftarrow Q(S,A) + \alpha\big[R + \gamma \max_a Q(S',a) - Q(S,A)\big]$
    $S \leftarrow S'$
  until S is terminal

Figure 3 : Q-Learning Algorithm

Experiments and Results

All the graphs were smoothed by taking an average using a window of size 10.

Multiple Epsilon values

Multiple epsilon values was tested to see how its value affects the optimal policy was found and the time taken for convergence. The optimal policies found in both cases were the same. Figure 4 shows the optimal policy obtained in case of SARSA. It learns a safer route and the optimal policy moves completely away from the cliff so that even in case of a random action not following the policy would not result in much loss. Figure 5 shows the policy learnt for Q-Learning. The policy learnt goes very close to the cliff. The rewards for ε = 0.2 is more spiky than ε = 0.1. This naturally because of higher exploration in the former case leading to a more irregular graph. Both seem to converge for same number of episodes. In both cases the number of steps taken per episode in SARSA is higher. This is also consistent with the optimal policy when the total number of steps is higher in SARSA compared to Q-Learning.

```
1  1  1  1  1  1  1  1  1  1  1  1
1  0  0  0  0  0  0  0  0  0  0  1
1  0  0  0  0  0  0  0  0  0  0  1
1  C  C  C  C  C  C  C  C  C  C  1
```

Figure 4 : Optimal Policy for Sarsa

```
0  0  0  0  0  0  0  0  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0
1  1  1  1  1  1  1  1  1  1  1  1
1  C  C  C  C  C  C  C  C  C  C  1
```

Figure 5 : Optimal Policy for Q-Learning

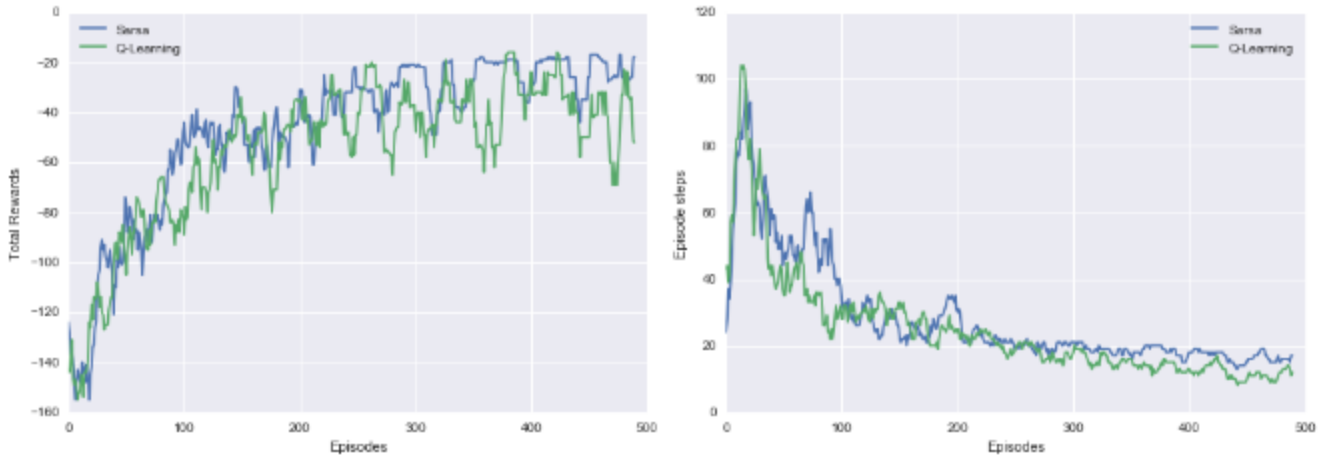

Figure 6 : Performance for ε = 0.1

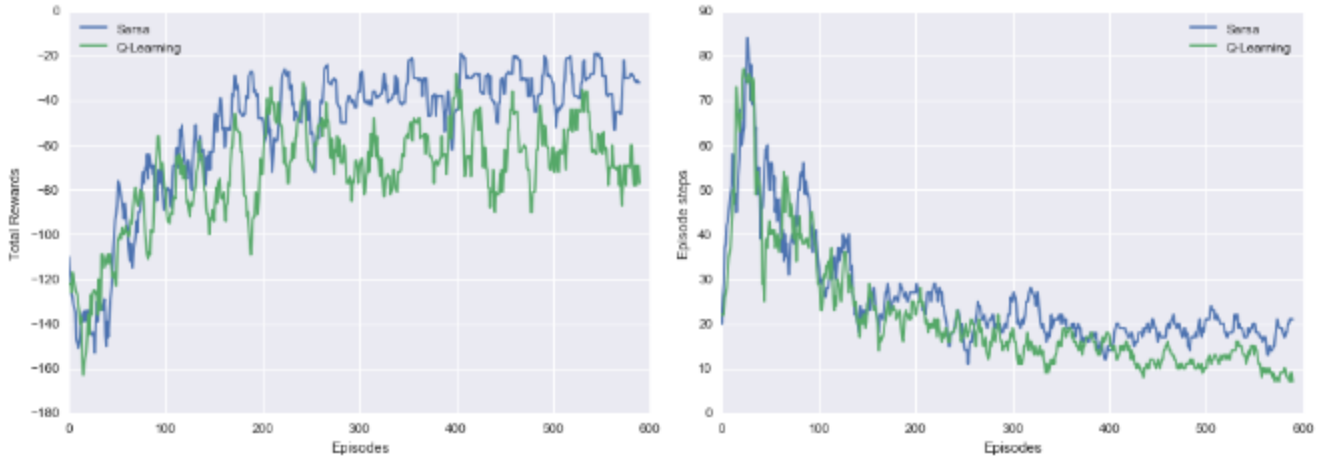

Figure 7 : Performance for ε = 0.2

Lesser Negative Reward Cliff States

The reward for falling into the cliff was changed from -100 to -20. This does not make any difference in the policy learnt by Q-Learning. However SARSA now learns a less safer route. However if you decrease the reward further the optimal policy learnt is weird as shown in Figure 9. This is probably because falling into the cliff gives better reward and ends the episode rather than exploring and reaching the goal state. Also the difference in the rewards for both the methods reduces as seen in Figure 10.

0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	1
1	C	C	C	C	C	C	C	C	C	C	1

Figure 8 : New policy for SARSA for reward = -20

0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
1	1	C	C	C	C	C	C	C	C	C	0

Figure 9 : New policy for SARSA for reward = -10

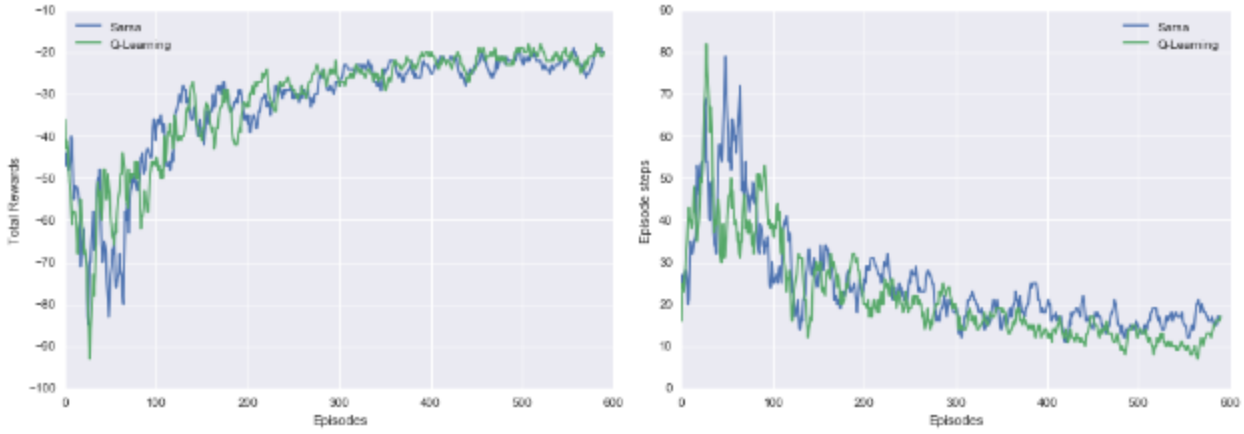

Figure 10 : Performance for new reward

Different Cliff States

The cliff states were increased as shown in Figure 11 and 12. This changed the policy as expected and pushed the policy learnt by Q-Learning a row higher. In terms of the rewards obtained in both cases seem to be similar since the policy learnt is almost similar.

1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	1
1	C	C	C	C	C	C	C	C	C	C	1
1	C	C	C	C	C	C	C	C	C	C	1

Figure 11 : Optimal Policy for Sarsa with new cliff states

0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1
1	C	C	C	C	C	C	C	C	C	C	1
1	C	C	C	C	C	C	C	C	C	C	1

Figure 12 : Optimal Policy for Q-Learning with new cliff states

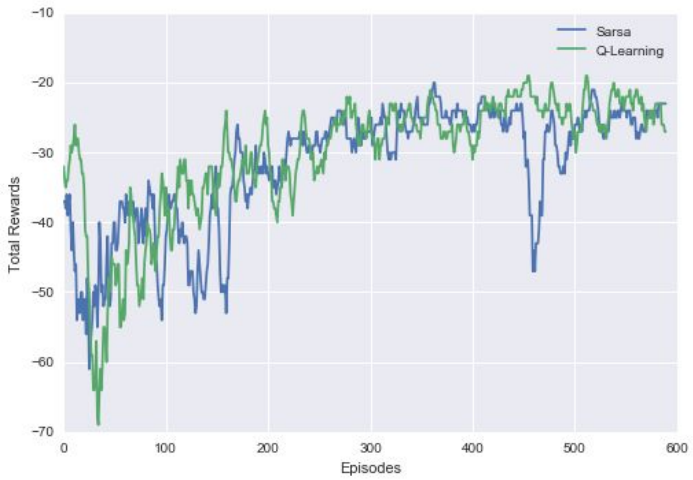

Figure 13: Performance comparison

Code

Please find the code here.

References

[1] Reinforcement Learning:An Introduction, Second Edition, Richard S. Sutton and Andrew G. Barto