Christopher Tsai

CS 349 HW #6 Free Response Questions

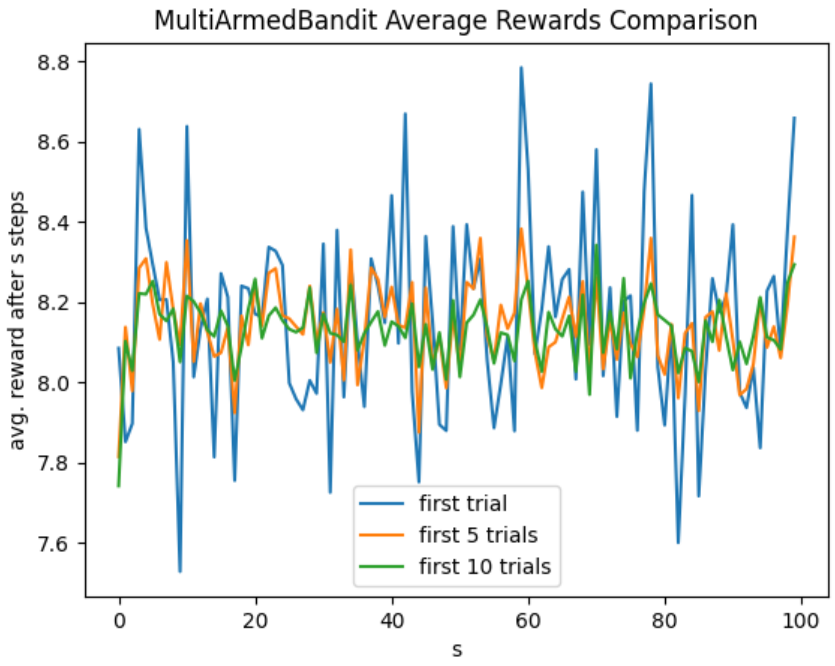
1a:

States are all the possible scenarios of x's and o's being filled out in all of the 9 spots. There are 9 actions, one for filling an x or an o (depending on the player's assignment) in each of the 9 spots. Current state affects decision-making because not all actions are feasible at each state, since a player can't mark a spot twice.

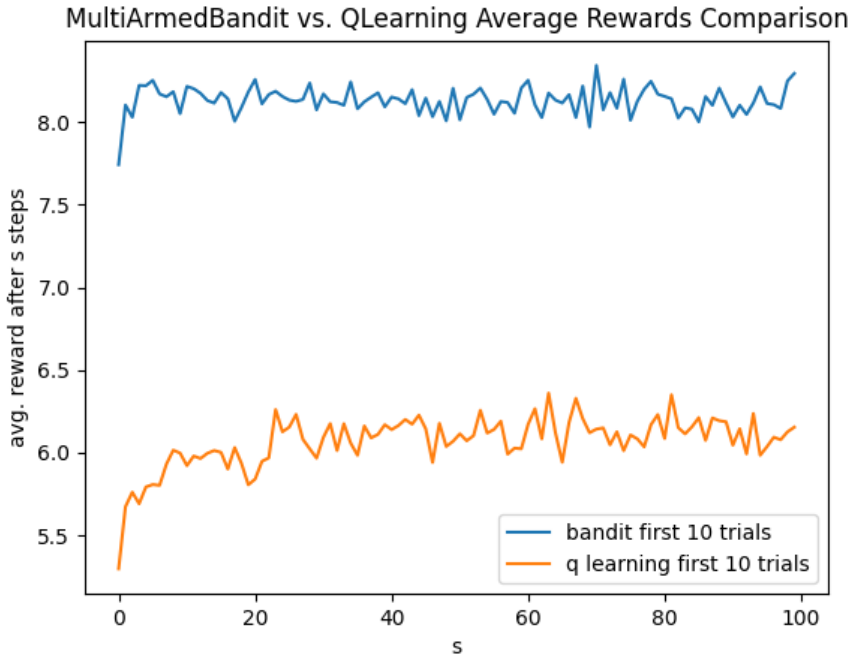
1b:

My reward function returns 1 if the agent wins the game, -1 if the opponent wins the game, and 0.5 if the game is a tie. It returns -0.1 otherwise, so that the agent prioritizes winning is the least number of turns. This would happen after the opponent plays. This is just a general design and would require tuning. The -0.1 might be too harsh, in which case I would replace it with a 0.

2a:



2b:



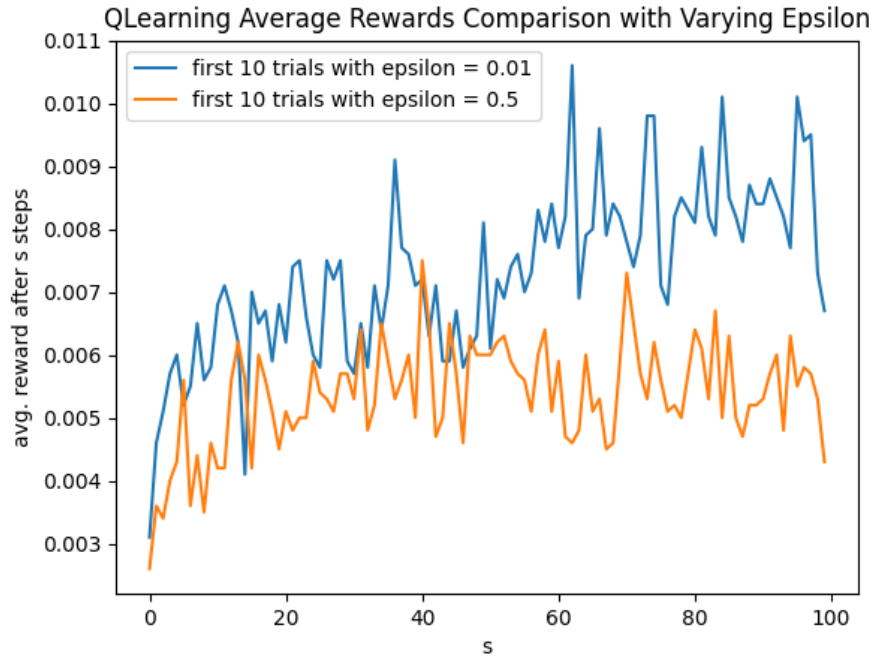
2c:

As the number of trials increased in question 2a's plot, the jaggedness of the line decreased. This is especially apparent when trials are increased from 1 to 5 but the change from 5 to 10 is also clearly visible. It is important to average the same learner over multiple trials in order to keep this jaggedness low and ensure we are viewing accurate representations of the rewards our learner acquired over time. This helps us when comparing multiple learner types.

2d:

The reward obtained by the Bandit learner is consistently higher than the reward obtained by the Q-learning learner. The difference in reward is about 2 throughout the s values. The jaggedness of both rewards seems low enough for us to observe an accurate comparison.

3a:



3b:

The epsilon value of 0.01 produced the single highest reward value in the graph. It produced consistently higher reward values at most s values (at least for these iterations). What this says is that a higher exploitation-to-exploration ratio produced better results in this case.

4a:

On-policy means learning while already knowing the target policy and off-policy means exploring different policies separate to the target policy and learning through that. On-policy is like having an adult tell a baby what to do and off-policy is like having the baby explore multiple things and learn what the right things to do are through exploring.

4b:

If we want to learn the shortest path, we should use an on-policy algorithm whose target policies specifically avoid going too far near the cliff. This is because using an off-policy algorithm can very likely lead to falling off the cliff due to the exploratory nature of off-policy.

5a:

Adding a 10% change of the robot suddenly failing effectively inflates the punishment (negative reward) of moving a single step. To account for this change, we increase this punishment in the MDP to, let's say, -10. We could also increase the reward of picking trash and depositing trash by a similar proportion (to 100 and 1000, for example).

5ba:

State-values transition matrix	To: 0	To: 1	To: 2
From: 0	0	0	1
From: 1	0	0	1
From: 2	0	0	0

State-action-values transition matrix	Action: A	Action: B
From: 0	1	0
From: 1	1	1
From: 2	0	0

5bb:

There are three equal optimal policies:

- To go from state 0 directly to state 2 through action A.
- To go from state 0 to state 1 through action B and then from state 1 to state 2 through action A.
- To go from state 0 to state 1 through action B and then from state 1 to state 2 through action B.

5bc:

State-values transition matrix	To: 0	To: 1	To: 2

From: 0	0	0	1
From: 1	0	0	0.95
From: 2	0	0	0

State-action-values transition matrix	**Action: A**	**Action: B**
From: 0	1	0
From: 1	0.95	0.95
From: 2	0	0

Now the optimal policy is strictly to go from state 0 to state 2, with no stops in between. Discount factor penalizes larger number of steps so it makes sense that it would make us prefer the shorter route in this case.