Names: Venkat Mohit Sornapudi, Laréb Fatima Ahmad *Date*: 23-12-2021

Student ID's: 2721697, 2737338

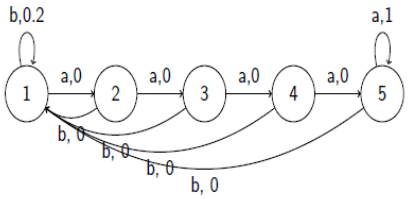
Dynamic Programming and Reinforcement Learning

Assignment - 4

All the referenced code snippets, and can be found in Appendix section.

Environment

The given problem is a 5 states MDP with a discount factor of 0.9 that starts from state 0, as shown below:



At any given state, if action 0 is taken a transition is taken to state 0. If this transition happens from state 0 it results in a reward of 0.2, else 0. Similarly, action 1 always takes to the last state of the chain. If this transition happens from the last state it results in a reward of 1, else 0. This is implemented by function named as *take_step* as shown in code snippet 1.

The assignment also discusses 10 states MDP which follows the above rules.

Part 1

In this part, we find the optimal Q values by using Q-learning.

Method:

The Q values corresponding to every combination of state and actions pairs are first initialized as zeros. These values are updated in several iterations. We loop the updation on given maximum number of episodes and steps.

For every episode, we restart the system at the state 0 and run the given number of steps. In every step we take, possible transitions according to the defined policy. Here we consider epsilon-greedy policy. This ensures that we take *epsilon* x 100% times a random action and take the optimal action (obtained by taking argmax on Q values at the current state). Using *take_step* function, we take this action in the environment and get *new_state* and reward obtained. These rewards are aggregated over each episode to study the statistics of the progress. Then, *max_future_q* and *current_q* values are derived as maximum possible Q at *new_state* and Q at the current state respectively. *new_q* is then obtained by using the following expression:

*new_q = (1 - LEARNING_RATE) * current_q + LEARNING_RATE'* (reward + DISCOUNT* max_future_q)*

We then update Q value at current state and taken action as *new_q*. Lastly, we replace current state by *new_state* and loop this updation process for maximum steps number of times in each episode.

For each episode, we decay the epsilon value if it is in the range of *START_DECAYING_EPISODE* and *END_DECAYING_EPISODE*. This decay is done linearly in the from *INITIAL_EPSILON* to zero.

Results:

Optimal Q values found:

```
[[ 6.561   6.1049]
 [ 7.29    5.9049]
 [ 8.1     5.9049]
 [ 9.      5.9049]
 [10.      5.9049]]
```

(a) 5 states MDP

```
[[ 3.87420489  3.6867844 ]
 [ 4.3046721   3.4867844 ]
 [ 4.782969    3.4867844 ]
 [ 5.31441     3.4867844 ]
 [ 5.9049      3.4867844 ]
 [ 6.561       3.4867844 ]
 [ 7.29        3.4867844 ]
 [ 8.1         3.4867844 ]
 [ 9.          3.4867844 ]
 [10.          3.4867844 ]]
```

(b)10 states MDP