CS6700 : Reinforcement Learning
Written Assignment #1

Intro to RL, Bandits, DP — Deadline: 23 Feb 2020, 11:55 pm

Name: Pranav Aurangabadkar — **Roll number:** ED18S007

- This is an individual assignment. Collaborations and discussions are strictly prohibited.
- Be precise with your explanations. Unnecessary verbosity will be penalized.
- Check the Moodle discussion forums regularly for updates regarding the assignment.
- Type your solutions in the provided LaTeXtemplate file.
- **Please start early.**

1. (2 marks) You have come across Median Elimination as an algorithm to get $(\epsilon, \delta)-$PAC bounds on the best arm in a bandit problem. At every round, half of the arms are removed by removing arms with return estimates below the median of all estimates. How would this work if we removed only one-fourth of the worst estimated arms instead? Attempt a derivation of the new sample complexity.

Solution: Median Elimination Algorithm one fourth of the worst estimated arms:
1. Set S=A
2. $\epsilon_1 = \epsilon/16, \delta_1 = \delta/2, l = 1$
3. Sample each arm $(2/{\epsilon_l}^2)\, ln\left(\frac{7}{3\delta_l}\right)$ times and let $\hat{p}_a^l$ denote its empirical value.
4. Find the first quartile of $\hat{p}_a^l$ and denote it by $\hat{q}_a^l$.
5. $S_(t+1) = S_t\left\{a : \hat{p}_a^l < \hat{q}_a^l\right\}$
6. If $|S_l| = 1$ then output S_l. Else $\epsilon_{l+1} = 15\epsilon_l/16, \delta_{l+1} = \delta_l/2, l = l+1$ Go to 3.

In each round one fourth of the arms are eliminated so total it takes $\log_{\frac{4}{3}} n$ rounds so that we are left with one arm.
The new sample arm complexity is

$$\sum_{l=1}^{\ln_{\frac{4}{3}} n} (2n_l/{\epsilon_l}^2) \ln\left(\frac{7}{3\delta_l}\right) = 2\sum_{l=1}^{\ln_{\frac{4}{3}} n} \left(\frac{\left(\frac{n}{\left[\frac{4}{3}\right]^{l-1}}\right) \ln \frac{2^l 7}{3\delta}}{\left[\left(\frac{15}{16}\right)^{l-1} \frac{\epsilon}{16}\right]^2} \right)$$

$$\sum_{l=1}^{\ln_{\frac{4}{3}} n} (2n_l/{\epsilon_l}^2) \ln\left(\frac{7}{3\delta_l}\right) = 512\sum_{l=1}^{\ln_{\frac{4}{3}} n} n\left(\frac{64}{75}\right)^{l-1} \left[\frac{\ln\left(\frac{1}{3\delta}\right)}{\epsilon^2} + \frac{7}{\epsilon^2} + \frac{l\ln 2}{\epsilon^2}\right]$$

$$\sum_{l=1}^{\ln_{\frac{4}{3}} n} (2n_l/{\epsilon_l}^2) \ln\left(\frac{7}{3\delta_l}\right) = 512\left(\frac{n\ln\frac{1}{\delta}}{\epsilon^2}\right)\sum_{l=1}^{\ln_{\frac{4}{3}} n} \left(\frac{64}{75}\right)^{l-1} [lC' + C] \leq O\left(\frac{n\ln\frac{1}{\delta}}{\epsilon^2}\right)$$

Thus sample complexity is bounded by $O\left(\frac{n\ln\frac{1}{\delta}}{\epsilon^2}\right)$.

2. (3 marks) Consider a bandit problem in which you know the set of expected payoffs for pulling various arms, but you do not know which arm maps to which expected payoff. For example, consider a 5 arm bandit problem and you know that the arms 1 through 5 have payoffs 3.1, 2.3, 4.6, 1.2, 0.9, but not necessarily in that order. Can you design a regret minimizing algorithm that will achieve better bounds than UCB? What makes you believe that it is possible? What parts of the analysis of UCB will you modify to achieve better bounds?

Solution: In UCB initially all arms are selected and at each step we select an arm j which maximizes $\left(Q(j) + \sqrt{\frac{2lnn}{n_j}}\right)$, where the first term is the estimate of the arm, the second term in the expression is the uncertainty in the estimate. As we don't know the mapping between arms and expected payoffs i.e q*(a) a better regret minimizing algorithm is UCB improved where action is chosen according to
$\left(Q(j) + \sqrt{\frac{2lnn}{n_j}}\Delta j\right)$
where $\Delta j = q*(a*) - q*(a_i)$ is also in the maximization part.
Suppose the uncertainty in UCB falls below $0.5(q*(a*) - q*(a_i)) = 0.5(4.6 - 3.1) = 0.75$ then the action with highest estimate of Q will be the optimum action. We can be sure of this as arm with worst estimate will be $< 3.1 + 0.75 = 3.85$ and the optimum arm here 4.6 will be $> 4.6 - 0.75 = 3.85$. This can be ensured if each arm is pulled n times minimizing regret.

3. (3 marks) Suppose you face a 2-armed bandit task whose true action values change randomly from time step to time step. Specifically, suppose that, for any time step, the true values of actions 1 and 2 are respectively 0.1 and 0.2 with probability 0.5 (case A), and 0.9 and 0.8 with probability 0.5 (case B).

   (a) (1 mark) If you are not able to tell which case you face at any step, what is the best expectation of success you can achieve and how should you behave to achieve it?

Solution: The goal of any reinforcement agent is to maximize expected reward. If we are not told which case we face at any step we can assume we face half of the times each case. As the cases are unknown the optimum action to choose can be picked randomly in this case the expectation of playing each case is
$E[A] = 0.5(0.1) + 0.5(0.9) = 0.5$
$E[B] = 0.5(0.2) + 0.5(0.8) = 0.5$
The total expected reward over many cases is
$(0.5)E[A] + (0.5)E[B] = 0.5$
UCB with incremental Q value update can be used.

(b) (2 marks) Now suppose that on each step you are told whether you are facing case A or case B (although you still don't know the true action values). This is an associative search task. What is the best expectation of success you can achieve in this task, and how should you behave to achieve it?

Solution: Here two separate Q value estimates can be updated independently if we are told which case we are facing. The best expectation of success in this associative search task is select action 2 in case A and select action 1 in case B. The expected reward is
$0.5\,(0.2) + 0.5\,(0.9) = 0.55$
The expected reward increases if we know which case agent is facing.

4. (5 marks) Many tic-tac-toe positions appear different but are really the same because of symmetries.

(a) (2 marks) How might we amend the learning process described above to take advantage of this? In what ways would this change improve the learning process?

Solution: In tic-tac-toe the actions are either 'X' or 'O' whereas the rewards can be defined as +1 for winning -1 for losing and 0 in case of draw. However due to 4 axis of symmetry the state space can be reduced to a more compact representation thus reducing overall space-action space(memory).

(b) (1 mark) Suppose the opponent did not take advantage of symmetries. In that case, should we? Is it true, then, that symmetrically equivalent positions should necessarily have the same value?

Solution: It is important for the agent to get the V shape in order to win. Due to symmetry if the player plays correctly initially except at a corner position the agent is unable to exploit it and will perform poorly. Thus symmetrically equivalent positions should not have the same value.

(c) (2 marks) Suppose, instead of playing against a random opponent, the reinforcement learning algorithm described above played against itself, with both sides learning. What do you think would happen in this case? Would it learn a different policy for selecting moves?

Solution: Here playing against itself can be categorized in two cases as follows. Case 1− There is only one agent playing. In this case the agent will learn to win and learning optimal policy.

Case 2− There are two agents(same algorithm) both playing against each other. When both the sides are learning that is if the first agent gets a reward of +1 i.e. wins the second agent gets a reward of -1 i.e. loses. Both the agents will always try to maximize expected reward. Over a large number of episodes learning happens and eventually all the games played will result in a Draw situation (same moves for all episodes) that is zero rewards for both the agents as equilibrium is reached.

5. (1 mark) Ego-centric representations are based on an agent's current position in the world. In a sense the agent says, I don't care where I am, but I am only worried about the position of the objects in the world relative to me. You could think of the agent as being at the origin always. Comment on the suitability (advantages and disadvantages) of using an ego-centric representation in RL.

Solution: Advantages of ego-centric representation in RL-
The agent works better in dyanamic environment.
Quick learning is adapted by the agent in egocentric representation of RL.
Disadvantages of ego-centric representation in RL-
If the immediate environment states is similar for different actions then distinguishing is difficult for the agent.
If the environment is stationary computation cost increases.

6. (2 marks) Consider a general MDP with a discount factor of γ. For this case assume that the horizon is infinite. Let π be a policy and V^π be the corresponding value function. Now suppose we have a new MDP where the only difference is that all rewards have a constant k added to them. Derive the new value function V^π_{new} in terms of V^π, c and γ.

Solution: $V^\pi(s) = E_\pi\{R_t|s_t = s\}$
$V^\pi(s) = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s\right\}$
For each reward a constant c is added to it.
$\hat{r}_{t+k+1} = r_{t+k+1} + c$
Thus new value function can be written as
$V^\pi_{new} = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k \hat{r}_{t+k+1}|s_t = s\right\}$
$V^\pi_{new} = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s\right\} + E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k c|s_t = s\right\}$
$V^\pi_{new} = V^\pi(s) + c\sum_{k=0}^{\infty}\gamma^k$
$V^\pi_{new} = V^\pi(s) + \frac{c}{1-\gamma}$

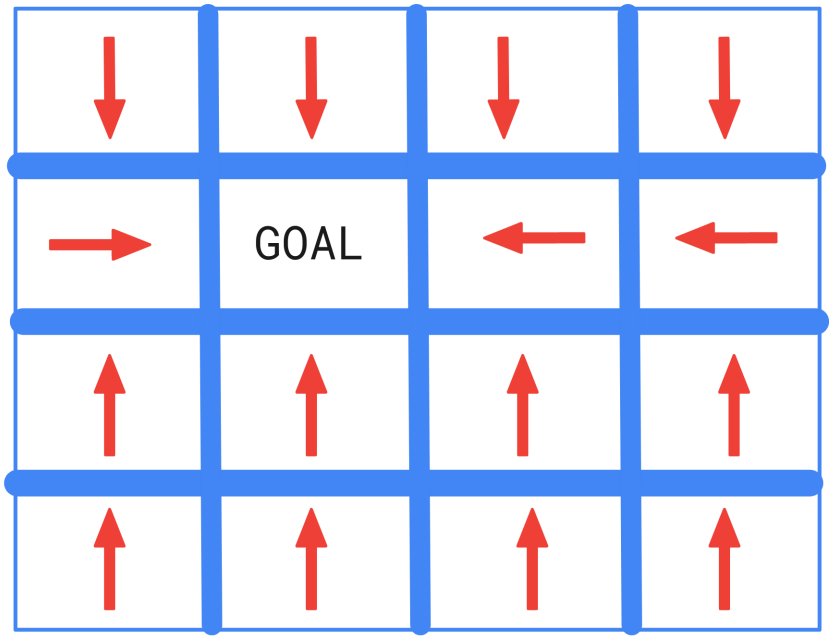
7. (4 marks) An ϵ-soft policy for a MDP with state set $\mathcal{S}$ and action set $\mathcal{A}$ is any policy that satisfies

$$\forall a \in \mathcal{A}, \forall s \in \mathcal{S} : \pi(a|s) \geq \frac{\epsilon}{|\mathcal{A}|}$$

Design a stochastic gridworld where a deterministic policy will produce the same trajectories as a ϵ-soft policy in a deterministic gridworld. In other words, for every trajectory under the same policy, the probability of seeing it in each of the worlds is the same. By the same policy I mean that in the stochastic gridworld, you have a deterministic policy and in the deterministic gridworld, you use the same policy, except for ϵ fraction of the actions, which you choose uniformly randomly.

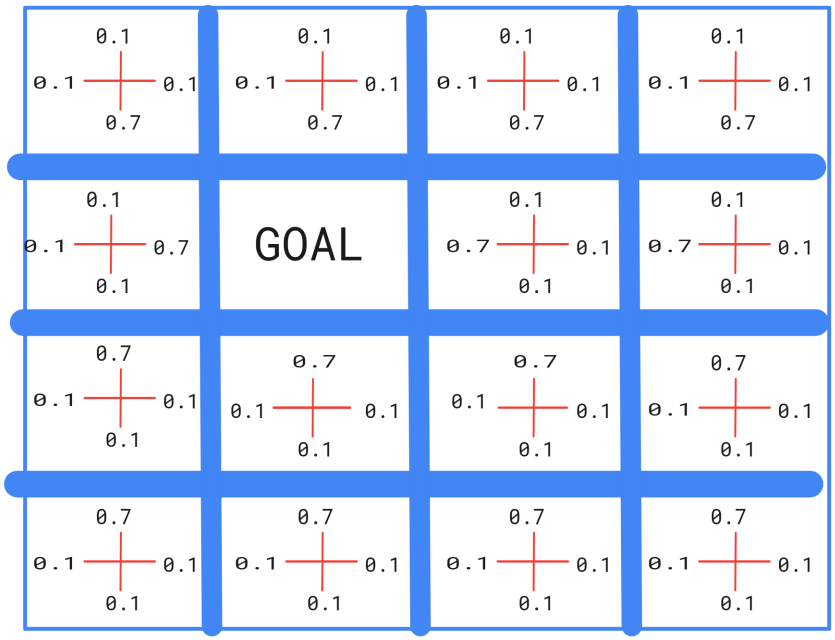
(a) (2 marks) Give the complete specification of the world.

Solution: Let us first begin with simple deterministic gridworld and deterministic policy as shown in the figure below.



In a deterministic policy every state has a specific action as shown in the figure. Now to specify a ϵ-soft policy for a simple deterministic gridworld the actions in each state will have a probablility of $(1-\epsilon+\frac{\epsilon}{numactions})$ and other actions will have an equal probability of $\frac{\epsilon}{numactions}$. Here numactions is 4 as left, right, up and down with deterministic action with probability $(1-0.75\epsilon)$ and uniformly randomly otherwise. Let us set $\epsilon = 0.4$ here thus soft policy with probability 0.7

other action with probability 0.1. Now we can construct from above a stochastic gridworld as shown in the figure below



This in turn can be the stochastic gridworld needed as asked above. Here if the deterministic policy is used and trajectory is generated we will get the same as the one in deterministic gridworld with ϵ-soft policy.

(b) (2 marks) Will SARSA on the two worlds converge to the same policy? Justify.

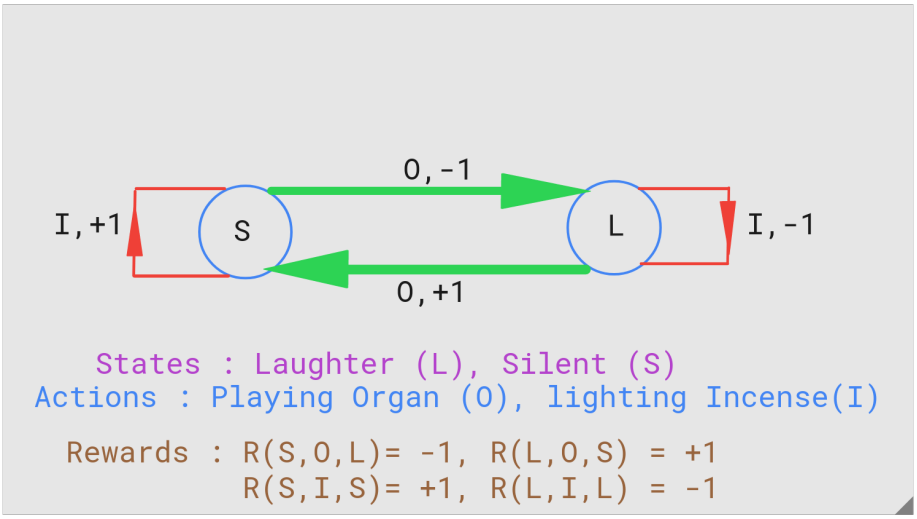
Solution: In SARSA starting with equiprobable random policy in stochastic gridworld after few episodes the the Q-table is updated and an optimum policy is reached. This may be different from the deterministic policy as either there will be many optimum policy giving the same expected reward or to begin with the chosen deterministic policy won't be optimum. However SARSA guarantees to converge to an optimum policy.

8. (7 marks) You receive the following letter:
Dear Friend, Some time ago, I bought this old house, but found it to be haunted by ghostly sardonic laughter. As a result it is hardly habitable. There is hope, however, for by actual testing I have found that this haunting is subject to certain laws, obscure but infallible, and that the laughter can be affected by my playing the organ or burning

incense. In each minute, the laughter occurs or not, it shows no degree. What it will do during the ensuing minute depends, in the following exact way, on what has been happening during the preceding minute: Whenever there is laughter, it will continue in the succeeding minute unless I play the organ, in which case it will stop. But continuing to play the organ does not keep the house quiet. I notice, however, that whenever I burn incense when the house is quiet and do not play the organ it remains quiet for the next minute. At this minute of writing, the laughter is going on. Please tell me what manipulations of incense and organ I should make to get that house quiet, and to keep it so.
Sincerely,
At Wits End

(a) (3 marks) Formulate this problem as an MDP (for the sake of uniformity, formulate it as a continuing discounted problem, with $\gamma = 0.9$. Let the reward be +1 on any transition into the silent state, and -1 on any transition into the laughing state.) Explicitly give the state set, action sets, state transition, and reward function.

Solution: The following figure shows states space, action space and rewards for different transitions



The state space S : Laughter (L), Silent (S)
The action space A : Playing Organ (O), Lighting Incense(I)
The state transition is specified as (next state, action, current state) and they are :
(Silent, Playing Organ, Laughter), (Laughter, Playing Organ, Silent),
(Silent, Lighting Incense, Silent), (Laughter, Lighting Incense, Laughter).
The rewards function is specified as R(next state, action, current state) and is defined as for various transitions:

R(S,O,L) = -1,R(L,O,S) = +1,R(S,I,S) = +1,R(L,I,L) = -1 .

(b) (2 marks) Starting with simple policy of **always** burning incense, and not playing organ, perform a couple of policy iterations.

Solution: Simple policy of always burning incense and not playing organ.
$\pi\left(playingorgan|laughter\right) = \pi\left(O|L\right) = 0$
$\pi\left(lightingincense|laughter\right) = \pi\left(I|L\right) = 1$
$\pi\left(playingorgan|silent\right) = \pi\left(O|S\right) = 0$
$\pi\left(lightingincense|silent\right) = \pi\left(I|S\right) = 1$
Initialize $V(L) = V(S) = 0$
Assume $\gamma = 0.9$
Policy evaluation :
$\Delta = 0$
$V(L) = \pi\left(O|L\right)\left[r + \gamma V(L)\right] + \pi\left(I|L\right)\left[r + \gamma V(L)\right] = 0 * \left[1 + 0.9 * 0\right] + 1 * \left[-1 + 0.9 * 0\right] = -1$
$V(S) = \pi\left(O|S\right)\left[r + \gamma V(S)\right] + \pi\left(I|S\right)\left[r + \gamma V(S)\right] = 0 * \left[-1 + 0.9 * 0\right] + 1 * \left[1 + 0.9 * 0\right] = +1$
$\Delta = max(\Delta, |V - V_s|) = 1$
$V(L) = \pi\left(O|L\right)\left[r + \gamma V(L)\right] + \pi\left(I|L\right)\left[r + \gamma V(L)\right] = 0 * \left[1 + 0.9 * (-1)\right] + 1 * \left[-1 + 0.9 * (-1)\right] = -1.9$
$V(S) = \pi\left(O|S\right)\left[r + \gamma V(S)\right] + \pi\left(I|S\right)\left[r + \gamma V(S)\right] = 0 * \left[-1 + 0.9 * 1\right] + 1 * \left[1 + 0.9 * 1\right] = +1.9$
$\Delta = max(\Delta, |V - V_s|) = 0.9$
$V(L) = \pi\left(O|L\right)\left[r + \gamma V(L)\right] + \pi\left(I|L\right)\left[r + \gamma V(L)\right] = 0 * \left[1 + 0.9 * (-1.9)\right] + 1 * \left[-1 + 0.9 * (-1.9)\right] = -2.71$
$V(S) = \pi\left(O|S\right)\left[r + \gamma V(S)\right] + \pi\left(I|S\right)\left[r + \gamma V(S)\right] = 0 * \left[-1 + 0.9 * 1.9\right] + 1 * \left[1 + 0.9 * 1.9\right] = +2.71$
$\Delta = max(\Delta, |V - V_s|) = 0.81$
$V(L) = \pi\left(O|L\right)\left[r + \gamma V(L)\right] + \pi\left(I|L\right)\left[r + \gamma V(L)\right] = 0 * \left[1 + 0.9 * (-2.71)\right] + 1 * \left[-1 + 0.9 * (-2.71)\right] = -3.44$
$V(S) = \pi\left(O|S\right)\left[r + \gamma V(S)\right] + \pi\left(I|S\right)\left[r + \gamma V(S)\right] = 0 * \left[-1 + 0.9 * 2.71\right] + 1 * \left[1 + 0.9 * 2.71\right] = +3.44$
$\Delta = max(\Delta, |V - V_s|) = 0.729$
$V(L) = \pi\left(O|L\right)\left[r + \gamma V(L)\right] + \pi\left(I|L\right)\left[r + \gamma V(L)\right] = 0 * \left[1 + 0.9 * (-3.44)\right] + 1 * \left[-1 + 0.9 * (-3.44)\right] = -4.096$
$V(S) = \pi\left(O|S\right)\left[r + \gamma V(S)\right] + \pi\left(I|S\right)\left[r + \gamma V(S)\right] = 0 * \left[-1 + 0.9 * 3.44\right] + 1 * \left[1 + 0.9 * 3.44\right] = +4.096$
$\Delta = max(\Delta, |V - V_s|) = 0.656$
Policy Improvement:
For state S:

$\pi(S) = argmax_a(r + \gamma V(S') = argmax(1 + 4.096, -1 - 4.096)$
Thus action in state S is lighting Incense.
For state L:
$\pi(L) = argmax_a(r + \gamma V(L') = argmax(-1 - 4.096, +1 + 4.096)$
Thus action in state L is Playing Organ.
New improved policy is:
$\pi\left(playingorgan|laughter\right) = \pi\left(O|L\right) = 1$
$\pi\left(lightingincense|laughter\right) = \pi\left(I|L\right) = 0$
$\pi\left(playingorgan|silent\right) = \pi\left(O|S\right) = 0$
$\pi\left(lightingincense|silent\right) = \pi\left(I|S\right) = 1$
Now policy iteration.
$V(L) = \pi\left(O|L\right)\left[r + \gamma V(L)\right] + \pi\left(I|L\right)\left[r + \gamma V(L)\right] = 1 * \left[1 + 0.9 * (-4.096)\right] + 0 * \left[-1 + 0.9 * (-4.096)\right] = -2.6864$
$V(S) = \pi\left(O|S\right)\left[r + \gamma V(S)\right] + \pi\left(I|S\right)\left[r + \gamma V(S)\right] = 0 * \left[-1 + 0.9 * 4.096\right] + 1 * \left[1 + 0.9 * 4.096\right] = +4.6856$
$\Delta = max(\Delta, |V - V_s|) = 0.5905$
Policy Improvement:
For state S:
$\pi(S) = argmax_a(r + \gamma V(S') = argmax(1 + 4.6856, -1 - 2.6864)$
For state L:
$\pi(L) = argmax_a(r + \gamma V(L') = argmax(-1 - 2.6864, +1 + 4.6856)$
The policy is an optimal policy as no improvement is observed.
$\pi\left(playingorgan|laughter\right) = \pi\left(O|L\right) = 1$
$\pi\left(lightingincense|laughter\right) = \pi\left(I|L\right) = 0$
$\pi\left(playingorgan|silent\right) = \pi\left(O|S\right) = 0$
$\pi\left(lightingincense|silent\right) = \pi\left(I|S\right) = 1$

(c) (2 marks) Finally, what is your advice to "At Wits End"?

Solution: If the rooms state is in laughter play the organ now the state transitions to silent. Here stop the organ and light the incense and never play the organ again so as to keep the room state in silent.

9. (4 marks) Consider the task of controlling a system when the control actions are delayed. The control agent takes an action on observing the state at time t. The action is applied to the system at time $t + \tau$. The agent receives a reward at each time step.

(a) (2 marks)What is an appropriate notion of return for this task?

Solution: The state is observed at time t and action is applied at time $t + \tau$ and the agent receives the reward at each time step
As the state transition is delayed we get same reward for same τ states. Thus the return is :
$G_t = \tau * R(s_t, a_{t-\tau-1}, s_{t-\tau-1}) + \sum_{k=0}^{\infty} \gamma^k R(t + \tau + k + 1)$

(b) (2 marks) Give the TD(0) backup equation for estimating the value function of a given policy.

Solution: TD(0) backup equation is :
$V(S_t) \leftarrow V(S_t) + \alpha [R_{t+\tau+1} + \gamma V(S_{t+\tau+1}) - V(S_t)]$