AI 3000 / CS5500 : Reinforcement Learning
Exam № 1 : Solutions

Due Date : 23/10/2021, 3.00 PM

Easwar Subramanian, IIT Hyderabad 23/10/2021

Problem 1A : Markov Reward Process

A fair coin is tossed repeatedly and independently. By formulating a suitable Markov reward process and using Bellman equations, find the expected number of tosses required for the pattern HTH to appear. (8 Points)

Problem 1A : Solution

Call HTH our target. Consider a chain that starts from a state called nothing (denote by $\emptyset$) and is eventually absorbed at HTH. If we first toss H then we move to state H because this is the first letter of our target. If we toss a T then we move back to $\emptyset$ having expended 1 unit of time. Being in state H we either move to a new state HT if we bring T and we are 1 step closer to the target or, if we bring H, we move back to H: we have expended 1 unit of time, but the new H can be the beginning of a target. When in state HT we either move to HTH and we are done or, if T occurs then we move to $\emptyset$. The transition diagram looks like below.

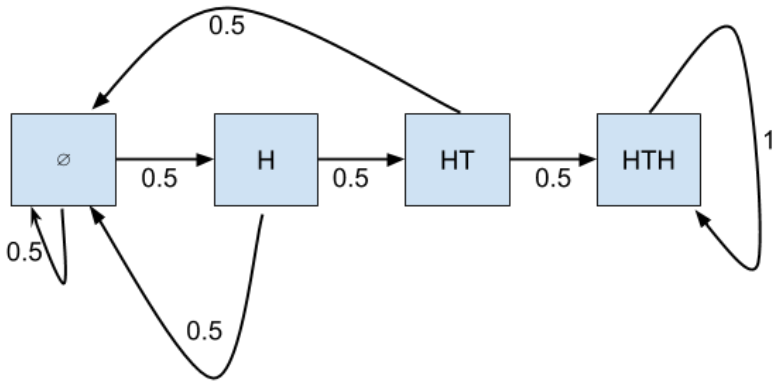


Figure 1: Suitable Markov Reward Process

Now we can write down the states of the MRP $< \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma >$ as follows.

- The set of states $\mathcal{S} = \{\emptyset, H, HT, HTH\}$

- The transition matrix $\mathcal{P}$ is given by,

$$\begin{array}{c} \\ \emptyset \\ H \\ HT \\ HTH \end{array} \begin{array}{c} \begin{array}{cccc} \emptyset & H & HT & HTH \end{array} \\ \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 \\ 0.5 & 0 & 0 & 0.5 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

- The absorbing state is HTH and this MRP is very similar to the snake and ladder problem discussed in the class. So, every time we toss a coin, we get a reward of -1 and when we reach the absorbing state we get a reward of 0. So, $\mathcal{R}(s) = -1 \quad \text{for} \quad s \in \{\phi, H, HT\}$ and $\mathcal{R}(HTH) = 0$.

- The discount factor $\gamma = 1$.

The Bellman evaluation equation for an MRP is given by $V = (I - \gamma\mathcal{P})^{-1}\mathcal{R}$ which when solved for $V(s)$ would give the "expected number" of coin tosses required to reach state HTH from any other state s of the MRP. The matrix $(I - \gamma P)$ becomes invertible if we set $V(s) = 0$ for $s = HTH$. One may find the inverse of the matrix $(I - \gamma\mathcal{P}_{3\times3})$ and multiply with $\mathcal{R}_{3\times1}$ to compute the expected coin tosses from any given state of the MRP. Specifically, we are interested from state ϕ. Upon solving one can find that the expected number of coin tosses from state $\emptyset$ to reach HTH is 10.

Problem 1B : Markov Reward Process

A Markov chain with state space $\mathcal{S} = \{1, 2, 3\}$ has the transition probability given by

$$\mathcal{P} = \begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1 \end{pmatrix}.$$

Is there a terminal state for the above Markov chain ? If so, what is the terminal state and why is it a terminal state ? What are the expected times to reach the terminal state starting from any other state of the Markov process ? (8 Points)

Problem 1B : Solution

State 3 is clearly an absorbing state as any transition from state 3 would end up in state 3 itself. There are no other absorbing states.

One approach is to model a suitable Markov reward process (MRP) with the $\mathcal{P}$ as given as above and $R = [1, 1, 0]^T$. The Bellman evaluation equation for an MRP is given by $V = (I - \gamma\mathcal{P})^{-1}\mathcal{R}$ which when solved for $V(s)$ would give the "expected number" of times required to reach state 3 from any other state of the MRP. The matrix $(I - \gamma P)$ becomes invertible if we set $V(s) = 0$ for state 3 even when we set $\gamma = 1$. One may find the inverse of the matrix $(I - \gamma\mathcal{P}_{2\times2})$

and multiply with $\mathcal{R}_{2\times 1}$ to compute the expected time taken from other states of the MRP to reach state 3. The expected time to reach state 3 from state 1 and 2 are 2.5 and 2 respectively.

Another approach to find the expected time to reach state 3 from any other state is through solving the system of equations as below. Let $\Psi(i)$ denote the expected time to reach state 3 from any other state $i \in \{1, 2, 3\}$. Then,

$$\begin{aligned}
\Psi(3) &= 0 \\
\Psi(2) &= 1 + \frac{1}{2}\Psi(2) + \frac{1}{2}\Psi(3) \\
\Psi(1) &= 1 + \frac{1}{3}\Psi(1) + \frac{1}{3}\Psi(2) + \frac{1}{3}\Psi(3)
\end{aligned}$$

Upon solving, we find, $\Psi(3) = 0; \quad \Psi(2) = 2; \quad \Psi(1) = 2.5$

.

Problem 1C : Markov Reward Process

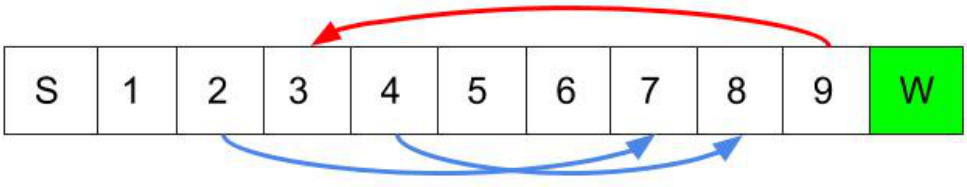
A fair coin is tossed repeatedly and independently. By formulating a suitable Markov reward process and using Bellman equations, find the expected number of tosses required for the pattern TTT to appear. (8 Points)

Problem 1C : Solution

The solution is similar to problem 1A. The answer is 14. (Pls. check)

Problem 1D and 1E : Markov Reward Process

Consider the following snake and ladders game as depicted in the figure below.



- Initial state is S and a fair four sided die is used to decide the next state at each time
- Player must land exactly on state W to win
- Die throws that take you further than state W leave the state unchanged

(a) Identify the states, transition matrix of this Markov process (1 points)

(b) Construct a suitable reward function, discount factor and use the Bellman equation for the Markov reward process to compute how long does it take "on average" (the expected number of die throws) to reach the state W from any other state (7 points)

Problem 1D and 1E : Solution

(a) The states of the Markov process is given by, $\mathcal{S} = \{S, 1, 3, 5, 6, 7, 8, W\}$. Positions 2 and 4 of the grid are same as positions 7 and 8 respectively.

The transition matrix is given by,

$$P = \begin{pmatrix} 0 & 0.25 & 0.25 & 0 & 0 & 0.25 & 0.25 & 0 \\ 0 & 0 & 0.25 & 0.25 & 0 & 0.25 & 0.25 & 0 \\ 0 & 0 & 0 & 0.25 & 0.25 & 0.25 & 0.25 & 0 \\ 0 & 0.25 & 0.25 & 0 & 0 & 0.25 & 0.25 & 0 \\ 0 & 0 & 0.25 & 0 & 0 & 0.25 & 0.25 & 0.25 \\ 0 & 0 & 0.25 & 0 & 0 & 0.25 & 0.25 & 0.25 \\ 0 & 0 & 0.25 & 0 & 0 & 0 & 0.5 & 0.25 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

(b) The state W is an absorbing state since if the Markov process reach the state W there are no further state transitions possible apart from staying at state W.

(c) Following are the suitable reward functions and discount factor γ.

- The suitable discount factor is $\gamma = 1$ as we are estimating "average" number of steps to reach W.
- The reward for any state could be $R(s) = -1$ for $s \neq W$ and $R(s) = 0$ for $s = W$. Then $V(s) = 0$ for $s = W$.
- The Bellman evaluation equation for an MRP given by $V = (I - \gamma P)^{-1} R$ which when solved for $V(s)$ would give the "average number" of die throws required to reach state W from state s. The matrix $(I - \gamma P)$ becomes invertible if we set $V(s) = 0$ for $s = W$. One may find the inverse of the matrix $(I - \gamma P_{7\times 7})$ to compute the average die throws from other seven states. Upon solving, the vector $V(s)$ is given by,

$$V(s) = \{7.0833, 7, 6.6667, 6.6667, 5.3333, 5.3333, 5.3333\}$$

Problem 2A : Bellman Equations and Dynamic Programming

(a) Consider an MDP $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ where the reward function has the structure

$$\mathcal{R}(s, a) = \mathcal{R}_1(s, a) + \mathcal{R}_2(s, a).$$

Suppose we are given optimal polices π_1^* and π_2^*, corresponding to reward fuctions $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively. Explain whether it is possible to combine these optimal policies in a simple manner to formulate an optimal policy π^* corresponding to the composite reward function $\mathcal{R}$. (4 Points)

(b) Let $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ be an MDP with fintite state and action space. We further assume that the reward function $\mathcal{R}$ to be a deterministic function of current state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$. Let f and g be two arbitraty action value functions mapping a state-action pair of the MDP to a real number, i.e. $f, g : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Let $\mathcal{L}$ denote the Bellman optimality operator (for the action value function) given by,

$$(\mathcal{L}f)(s,a) = \mathcal{R}(s,a) + \gamma P(s,a), V_f(s)$$

where $V_f(s) = \max_a f(\cdot, , a)$. Prove that,

$$\|\mathcal{L}f - \mathcal{L}g\|_\infty \leq \gamma \|f - g\|_\infty$$

(6 Points)

[**Note :** The Bellman optimality operator defined above is for action value functions and is different from the one that was defined in the lectures which is for value functions.Think of $V_f(s)$ as a transformation operator that turns a vector $f \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ into a vector of length $|\mathcal{S}|$. The max norm of an action value function f is defined as $\|f\|_\infty = \max_s \max_a |f(s,a)|$]

Problem 2A : Solution

1. Combining optimal policies is not straightforward as it involves taking care of the max operator which is a non-linear operator. Hence, optimal policies of the two MDPs cannot be combined in a straightforward fashion.

2. From definiiton, we have that,

$$\|\mathcal{L}f - \mathcal{L}g\|_\infty \leq \gamma \big\|PV_f - PV_g\big\|_\infty \leq \gamma \big\|V_f - V_g\big\|_\infty \leq \gamma \|f - g\|_\infty$$

where the first inequality uses that each element of $P(V_f - V_g)$ is a convex average of $(V_f - V_g)$. The last inequality follows from that fact that for each $s \in \mathcal{S}$, $|V_f(s) - V_g(s)| \leq \max_a |f(s,a) - g(s,a)|$. The easiest way to see this is let $V_f(s) > V_g(s)$ and let a_0 be the greedy action for f at s. Then,

$$|V_f(s) - V_g(s)| = f(s,a_0) - \max_a g(s,a) \leq f(s,a_0) - g(s,a_0) \leq \max_a |f(s,a) - g(s,a)|$$

Problem 2B : Bellman Equations and Dynamic Programming

(a) Consider an MDP $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ where the reward function has the structure

$$\mathcal{R}(s,a) = \mathcal{R}_1(s,a) + \mathcal{R}_2(s,a).$$

Suppose we are given action value functions Q_1^π and Q_2^π, for a given policy π, corresponding to reward fuctions $\mathcal{R}_1$ and $\mathcal{R}_2$, respectively. Explain whether it is possible to combine these action value functions in a simple manner to compute the action value function Q^π corresponding to the composite reward function $\mathcal{R}$. (4 Points)

(b) Let $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ be an MDP with fintite state and action space. We further assume that the reward function $\mathcal{R}$ to be a deterministic function of current state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$. Let f be a arbitraty action value function mapping a state-action pair of the MDP to a real number, i.e. $f, : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Prove that

$$\left\|V^* - V^{\pi_f}\right\|_\infty \leq \frac{2\|f - Q^*\|_\infty}{1-\gamma}$$

with π_f being the greedy policy with respect to f. (6 Points)

[**Note :** The max norm of an action value function f is defined as $\|f\|_\infty = \max_s \max_a |f(s,a)|$]

Caution!. The sub-questions of question 2 is asked in combinations in different exam sets

Problem 2B : Solution

1. Yes, it is possible to combine the two action value functions of the MDP into a single action value function for the composite MDP since the combination involve only expectation operator and it is linear in nature.

2. Fix a state $s \in \mathcal{S}$ and $a = \pi_f(s)$. Then,

$$\begin{aligned}
V^*(s) - V^{\pi_f}(s) &= Q^*(s, \pi^*(s)) - Q^{\pi_f}(s,a) \\
&= Q^*(s, \pi^*(s)) - Q^*(s,a) + Q^*(s,a) - Q^{\pi_f}(s,a) \\
&= Q^*(s, \pi^*(s)) - Q^*(s,a) + \gamma \mathbb{E}_{s' \sim P(s'|s,a)} \left[V^*(s') - V^{\pi_f}(s')\right] \\
&\leq Q^*(s, \pi^*(s)) - f(s, \pi^*(s)) + f(s,a) - Q^*(s,a) + \gamma \mathbb{E}_{s' \sim P(s'|s,a)} \left[V^*(s') - V^{\pi_f}(s')\right] \\
&\leq 2\|f - Q^*\|_\infty + \gamma \left\|V^* - V^{\pi_f}\right\|_\infty
\end{aligned}$$

Problem 3 : Monte Carlo Methods

Consider a Markov process with 2 states $\mathcal{S} = \{S, A\}$ with transition probablities as shown in the table below, where $p \in (0,1)$ is a non-zero probablity. To generate a MRP from this Markov chain, assume that the rewards for being in states S and A are 1 and 0, respectively. In addition, let the discount factor of the MRP be $\gamma = 1$.

	S	A
S	$1-p$	p
A	0	1

(a) Provide a generic form for a typical trajectory starting at state S. (1 Point)

(b) Estimate $V(S)$ using first visit MC. (2 Points)

(c) Estimate $V(S)$ using every visit MC. (2 Points)

(d) What is true value of $V(S)$? (3 Points)

(e) Explain if the every vist MC estimate is biased. (2 Points)

(f) In general for a MRP, comment on the convergence properties of the first visit MC and every visit MC algorithms (2 Points)

Problem 3 : Solution

(a) All trajectories starting in state S look like $SSS\cdots A$ where there are l occurences of S.

(b) First time estimate of $V(S)$ would be l.

(c) There are l every time estimates of $V(S)$ for every trajectory and their sum is $l(l+1)/2$. So, everytime estimate per trajectory is given by $(l+1)/2$

(d) The true value of $V(S)$ is $1/p$ as it is the average length of the trajectory.

(e) Yes. Expected value of the everytime estimate $((1/p)+1)/2$ which is erroneous by a factor of 2, if p were small

(f) FVMC convergence rely just on law of large numbers; whereas EVMC convergence may need to more sophisticated care since the samples for mean calculation are independent.

Problem 4 : Problem Forumlation

Consider a *SUBWAY* outlet in your locality. Customers arrive to the store at times governed by an unknown probablity distribution. The outlet sells sandwiches with a certain type of bread (choice of 4 types) and filling (choice of 5 types). If a customer cannot get the desired sandwich, he/she is not going to visit the store again. Ingredients need to be discarded every 3 days after purchase. The store owner wants to figure out a policy for buying ingredients in such a way to maximize his long-time profit using reiforcement learning. To this end, we will formulate the problem as a MDP. You are free to make other assumptions regarding the probem setting. Please enumerate your assumptions while answering the questions below.

(a) Suggest a suitable state and action space for the MDP. (5 Points)

(b) Devise an appropriate reward function for the MDP (3 Points)

(c) Would you use discounted or undiscounted setting in your MDP formulation ? Justify your answer. (3 Points)

(d) Would you use dynamic progamming or reinforcement learning to solve the the problem ? Explain with reasons. (3 Point)

(e) Between MC and TD methods, which would you use for learning ? Why ? (3 Point)

(f) Is function approximation required to solve this problem ? Why or why not ? (3 Point)

Problem 4 : Solution

We will leave out the first two questions as they are design choices. There is not one right answer. For example, the state space could be 4 variable to denote quantity of 4 types of bread and 5 other variables to denote volume of 5 types of filling. The reward function could indicate profit (or revenue) maximization. Exact details of the formula are needed. Depending on the choice of state, action and reward design, one can answer other questios as follows.

(c) Since the store owner is interested in maximizing long-time profit, the horizon is infinite horizon (or long finite time). So, it is good to use discounted setting

(d) DP is used in model based setting. It is difficult to get transition and reward function formulated accurately. Hence, model free RL approaches are to be used

(e) Since the time horizon is long (or even infinite), TD methods are to preferred.

(f) Depending on the state and action space formulation the choice of whether to use FA methods or not is to be decided. For example, if either state or action space is huge / continuous, then FA methods are to be used.

Problem 5A : Miscellaneous Questions

(a) Do we require MDP formalism to solve RL problems ? Explain. (1 Point)

(b) What could be the possible range for choosing the discount factor of an MDP in solving episodic tasks ? Justify your answer (2 Points)

(c) For a given MDP, will value and policy iteration converge to the same optimal value function V^* ? Justify your answer. (3 Points)

(d) The policy improvement step in the dynamic programming set up involves using the greedy operator to arrive at the next policy, while in the model free setting, ε-greedy improvement is used to arrive at the next policy. Why is this so ? (3 Points)

(e) For the policy evaluation problem, what are the advantages of using Monte-Carlo over DP methods ? (2 Points)

(f) What is the computational complexity of evaluating the value function of an MRP using the iterative Bellman updates ? (3 Points)

(g) Consider an MDP with three states $\{s_1, s_2, s_3\}$ and three actions $\{a_1, a_2, a_3\}$ with discount factor $\gamma = 0.5$. There is no noise in the environment and therefore all actions result in intended state transitions. The reward for transitioning into a state s_i is i. For example, if any action $a_k, k\{1, 2, 3\}$ pushes the agent into state s_3, then the reward is 3. We consider a Q-learning agent that uses the ϵ-greedy strategy. When the Q-values from a particular state are same for more than one action, the agent breaks ties by choosing the action a_k with lowest k. Let us initialize the Q table to zeros for all state-action pairs and let the learning rate be set to $\alpha = 0.7$. For $\epsilon \neq 0$, could the Q-learning agent generate the following trajectory given by

$$(s_1, a_1, 1, s_1, a_2, 2, s_2)$$

If yes, reason out, which of the two action is greedy and which of it is random ? (6 Points)

Problem 5A : Solutions

(a) MDP formalism is not required to solve RL problems. RL problems involve sequential decision making and MDP formalism is just one way to get to solve them and in several real world problems Markovian assumption may not hold at all.

(b) In general discount factor γ lies in the interval $[0, 1]$ but for epsiodic tasks normally $\gamma = 1$ is choosen as all epsidoes terminate after some time T and therefore there is no need to discount.

(c) Yes. For a given MDP, value and policy iteration will converge to same optimal value function V^* since all optimal policies achieve same optimal value function

(d) Exploration is not needed in the DP setting as the environment is well defined through transistion functions. Whereas in the model free world, one needs to explore to navigate various portions of state / action space and hence exploration is recommended.

(e) MC algorithms are model free, unbiasaed and work in non-Markovian setting as well. It doesn't require full back up to evaluate the value of the state.

(f) $O(S^2)$ per iteration

(g) Clearly, it is possible. For the first experience tuple $(s_1, a_1, 1, s_1)$, we start at $s1$ and select action a_1, which returns the reward as 1 and stay at the same state. The initial Q for all state-action is zeros. Since that ties are broken by choosing a_i with the smallest index i, the greedy action is thus a_1. Also, the random action can also pick the greedy action. For the second experience tuple $(s_1, a_2, 2, s_2)$, we start at s_2 and select action a_2, which returns the reward 2 and goes to state s_2. Now, we've updated the $Q(s_1; a_1)$ based on the first experience tuple, and $Q(s_1; a1) = 0.7$. Thus the optimal action for s_1 is a_1 now. The action a_2 could be taken only when the action is randomly selected.

Problem 5B : Miscellaneous Questions

(a) How many deterministic policies are possible when the MDP has finite state and action space ? (1 Point)

(b) What could be the possible range for choosing the discount factor of an MDP in solving continual tasks ? Justify your answer (2 Points)

(c) For a given MDP, will value and policy iteration converge to the same optimal policy π^* ? Justify your answer. (3 Points)

(d) Among model free control algorithms, why is Q-learning off-policy and SARSA on-policy ? (3 Points)

(e) For the policy evaluation problem, what are the advantages of using Monte Carlo over TD methods ? (2 Points)

(f) What is the computational complexity of evaluating the value function of an MRP using the matrix inversion method ? (3 Points)

(g) In the TD(λ) algorithm, we use λ returns as the target. The λ return target is given by,

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

where $G_t^{(n)}$ is the n-step return defined as,

$$G_t^{(n)} = r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1}r_{t+n} + \gamma^n V(s_{t+n}).$$

The parameter λ is used to determine the weights corresponding to each of the n-step returns in the λ-return target. We know that the weights decay exponentially with n. Therefore, in the G_t^λ sequence, after some terms, the weights of subsequent terms would have fallen by more than half as compared to the weight of the first term. Let $\eta(\lambda)$ denote the time by which the weighting sequence would have fallen to half of its initial value. Derive an expression that relates the parameter λ to $\eta(\lambda)$. Use the expression derived to compute the value of λ for which the weights would drop to half after 3 step returns. (6 Points)

Problem 5B : Solutions

(a) $|\mathcal{A}|^{|\mathcal{S}|}$

(b) $\gamma \in [0, 1)1$. Need to take care of infinite sum that can occur in continual tasks and also needed for convergence properties

(c) No, need not. Policy iteration can converge to a different optimal policy

(d) Both behaviour and target policy is same in SARSA which is ε-greedy; Target policy is greedy and behaviour policy is ε-greedy in Q-learning.

(e) Non-Markovian, unbiased estimate of value function

(f) $O(\mathcal{S}^3)$

(g) The λ-return is defined as

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_t^{(n)}$$

Defining $\tau = \eta(\lambda)$ to be the value of n such that

$$\lambda^\tau = \frac{1}{2} \implies \tau \approx \frac{\ln(\frac{1}{2})}{\ln(\lambda)}$$

Thus given λ, the half life τ is given by this expression. For example if $\lambda \approx \frac{1}{\sqrt{2}}$, then we compute that $\tau = 3$ so we are looking that many steps ahead before our weighting drops by one half.

(Pls. Check this is correct)

Problem 5C : Miscellaneous Questions

(a) Does a MDP admit at least one optimal deterministic policy ? Explain (2 Points)

(b) Between Monte Carlo and TD methods, which method can be used in online learning ? Explain with reasons ? (2 Points)

(c) Explain, why does value iteration converge ? (3 Points)

(d) What are the reasons for having a discount factor in defining a MDP ? (2 Points)

(e) For the policy evaluation problem, what are the advantages of using TD over Monte Carlo methods ? (2 Points)

(f) Why does TD methods for policy evaluation yields an biased estimate of the true value of the policy ? (3 Points)

(g) Consider the following experiment in evolution. We are interested in the evolution of a gene by name **mTOR**. The gene occurs in two variants T and S and plays an important role in determining the height of an individual. Each individual has a pair of this gene, either TT (tall) or TS (medium height) or SS (short). In the case where the individual is of medium height, the order of gene pairing is irrelevant (TS or ST is same). In evolution, an offspring inherits a pair of gene, one variant each from his/her biological parents, with equal probability. Thus if one partner is tall (TT) and other partner is medium (TS), the offspring has $1/2$ probability of being tall or $1/2$ probability of being medium.

We shall start with an individual of a arbitrary but fixed trait (TT or TS or SS). The other partner is of medium height (ST or TS). The offspring of such partners again finds a medium

height partner. This pattern repeats for a number of generations, wherein the resultant offspring always has a medium height partner.

(1) Write out the states and transition probabilities of this Markov process. (2 Point)

(2) Suppose we start with a medium height individual (as the first partner). What are the probabilities that any offspring belonging to first, second or third generation would be tall, medium or short in height ? (3 Points)

(3) What would be the answer to the previous question for any n-generations into future ? (1 Point)

[**Trivia and Disclaimer** : Actually, the height of an individual is not a single gene trait. But features like 'attached earlobes', 'short big toe' or 'PTC tasting' are examples of single gene traits. The purpose of this question is to test the understanding on Markov chains and not to drive home any biological concepts !!]

Problem 5B : Solutions

(a) Yes, MDP admit at least one optimal deterministic policy. The deterministic optimal policy is derived from greedy version of Q^*.

(b) TD methods, as in TD methods we can work with partial sequences unlike the in MC method, where we need full roll-outs

(c) Bellman operator (both evaluation and optimality) are contractions under max norm and then one can use Banach fixed point theorem.

(d) Convergence of Bellman updates, finite reward sum and variance reduction are some reasons to have discount factor

(e) TD methods use bootstrapping and hence they yield a biased estimate of the value function

(f) (a) The gene pairing of an individual, forms the states of the Markov process. Specifically, we let $S = \{TT, TS, SS\}$.

From the description of the problem one can write out the transition matrix P of the Markov process as follows,

$$P = \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0.25 & 0.5 & 0.25 \\ 0 & 0.5 & 0.5 \end{pmatrix} = 2^{-1} \begin{pmatrix} 1 & 1 & 0 \\ 1/2 & 1 & 1/2 \\ 0 & 1 & 1 \end{pmatrix}$$

(b) The elements from the second row of the matrix P^n will give us the probabilities for a medium height individual to give tall, medium or short off-springs in $n-1$-th generation in

this experiment, respectively (reading this row from left to right). Specifically, for 2nd, 3rd and 4th generation, we would have,

$$P^2 = 2^{-2}\begin{pmatrix} 1.5 & 2 & 0 \\ 1 & 2 & 1 \\ 0.5 & 2 & 1.5 \end{pmatrix}$$

$$P^3 = 2^{-3}\begin{pmatrix} 2.5 & 4 & 1.5 \\ 2 & 4 & 2 \\ 1.5 & 4 & 0.2.5 \end{pmatrix}$$

$$P^4 = 2^{-4}\begin{pmatrix} 4.5 & 8 & 3.5 \\ 4 & 8 & 4 \\ 3.5 & 8 & 0.4.5 \end{pmatrix}$$

so that, the probabilities for a medium height individual to give tall, medium or short offsprings is given by, P(tall) = 0.25, P(medium) = 0.5, P(short) = 0.25.

(c) Actually, the probabilities are same for any n-th generation. As an exercise, one can show that,

$$P^n = 2^{-n}\begin{pmatrix} \frac{3}{2} + (2^{(n-2)} - 1) & 2^{(n-1)} & \frac{1}{2} + (2^{(n-2)} - 1) \\ 2^{(n-2)} & 2^{(n-1)} & 2^{(n-2)} \\ \frac{1}{2} + (2^{(n-2)} - 1) & 2^{(n-1)} & 0.\frac{3}{2} + (2^{(n-2)} - 1) \end{pmatrix}$$

Problem 5D : Miscellaneous Questions

(a) What is the algorithm that results, if In the TD(λ) algorithm, we set $\lambda = 1$? (1 Point)

(b) What are the possible reasons to study TD(λ) over TD(0) method ? (2 Points)

(c) Given a MDP, does scaling of rewards using a positive scale factor, change the optimal policy ? (3 Points)

(d) In off policy evaluation, would it be beneficial to have the behaviour policy be deterministic and the target policy be stochastic ? (2 Points)

(e) Under what conditions, does temporal methods for policy evaluation converge to true value of the policy π ? Explain intitutively, the reasoning behind those conditions. (3 Points)

(f) Why does MC methods for policy evaluation yields an unbiased estimate of the true value of the policy ? (2 Points)

(g) Let $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ be an MDP with fintite state and action space. We further assume that the reward function $\mathcal{R}$ is non-negative for all state-action pairs. In addition, suppose that for every state $s \in \mathcal{S}$, there is some action a_s such that $P(s'|s, a_s) \geq p$ for some $p \in [0, 1]$. We intend to find optimal value function V^* using value iteration. Initialize $V_s(0) = 0$ for all

states of the MDP and let $V_t(s)$ denote the value of state s after t iterations. Prove that for all states s and $t \geq 0$, $V_{t+1}(s) \geq p\gamma V_t(s)$. (4 Points)

(h) Explain if it is possible to parallelize the value iteration algorithm (3 Points)

Problem 5D : Solutions

(a) MC update

(b) Better bias-variance tradeoff

(c) No. Explanation is required using expanding expectation of discounted rewards with and without scaling of rewards.

(d) Not much benefits. In fact, it can be counter productive. For example, it is useful for Behaviour policy to be random, as it will be simple to generate, can navigate many parts of state-action space.

(e) States and actions are finite; they are infintely visited often and Robbins Monroe condition on the learning rate is satisfied.

(f) MC methods are basically sample mean. Hence, they are unbiased

(g) From the value iteration algorithm, we have,

$$\begin{aligned} V_{t+1}(s) &= \max_a \left[R(s,a) + \gamma \sum_s^{'} P(s'|s,a) V_t(s') \right] \\ &\geq \left[R(s,a_s) + \gamma \sum_s^{'} P(s'|s,a_s) V_t(s') \right] \\ &\geq \left[R(s,a_s) + \gamma \rho V_t(s') \right] \geq \gamma \rho V_t(s') \end{aligned} \tag{1}$$

Third line uses the reward function is non-neqative and initilization is 0 and last inequaliity uses non-negative rewards

(h) Since V_{t+1} depends only on V_t, it is easy parallelize the back up operation for all states of the MDP

Problem 5E : Miscellaneous Questions

(a) What is the algorithm that results, if In the TD(λ) algorithm, we set $\lambda = 0$? (1 Point)

(b) Given a MDP, does scaling the discount factor using a scale factor $\kappa \in (0, 1)$, change the optimal policy ? Explain. (3 Points)

(c) Consider the grid world navigation problem. Following are two choices for reward function. One reward formulation assigns a reward of +5 at the goal state and -1 elsewhere. The other reward function is identical to the first one but assigns an additional reward of +2 to states that are two squares away from the goal state. Which reward formulation is suitable for the task to make the RL agent navigate the agent from a start state in the grid to the goal state in as few a steps as possible ? Explain. (3 Points)

(d) What are the benefits of off-policy learning ? (2 Points)

(e) Under what conditions, does MC methods for estimating action value function work effectively ? Why are those conditions required ? (2 Points)

(f) The goal of a RL agent is to choose a policy π such that it maxmizes the expected sum of total discounted rewards. What are the factors that are averaged in calculating the expectation of the sum of total discounted rewards ? (2 Points)

(g) Suppose if all rewards of an MDP are non-negative and are upper-bounded by $R_{\max}$, what is the lower and upper bound for the discounted sum of rewards ? (2 Points)

(h) We are helping a robot learn a policy π which is helpful in performing a real world application. To this end, we formulate an MDP $M =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$. Unfortunately, we only have access to a simulation software to generate samples and not from the robot directly. We know that the simulation software uses a transition model $\tilde{P}(s'|s,a)$ different from the real transition model given by $P(s'|s,a)$. Provide an expression for the update rule for learning Q^π using samples generated by the simulator. (3 Points)

(i) Continuing with set up from previous question; now the scenario is that we need to train n robots simulataneously. Each robot has its own transition model (assume other elements of the MDP are same). Is it possible to make all robots simultaneously learn about policy π by using samples generated by the simulator ? If yes, would the learnt action value function Q^π be identical for all robots ? (2 Points)

Problem 5E : Solutions

(a) TD(0)

(b) Yes, it does change the optimal policy. A classic example, is the exercise given by in Assignment 1 (Effect of Noise and Discounting).

(c) The first reward formulation. Since, in the second reward formulation the agent can get carried away by the reward of +2 and keep looping around that state which is not task we want the agent to learn

(d) Learn by observing other agents; Re-use previous experience generated from earlier policies; Learn about optimal policy while following exploratory policy; Learn about multiple policies while following one policy

(e) GLIE; all state action pairs are visited infintely often (a requirement in model free method) and infinite exploration in a way guarantees that

(f) Rewards obtained across various roll outs (including state-visits) and action taken in those states and possible next states

(g) Lower bound is 0; Upper bound is $\frac{R_max}{1-\gamma}$ where R_max is the maximum reward possible. (Check it out)

(h) Off-policy learning will come to aid. Use of importance sampling in the Q^π update rule is the key

(i) Yes, possible. Again using IS. The learnt Q^π would be different as iS ratios would be different for different robots.