SHAGUN UPPAL
2016088

Date: ....

Reinforcement Learning
Homework 3

① Exercise 5.4

Keeping record of the returns (G) for every state-action pair (s, a) :-

We average out the returns for the pair (s, a) in order to update the state-action value function. We initialise Q(s, a) randomly for all s & a.

→ For example,

$$Q_{n+1}(s, a) = \frac{1}{n}\left(\sum_{k=1}^{n} G_k\right)$$

for a first-visit Monte Carlo.

→ So,

$$Q_{n+1}(s, a) = \frac{1}{n}\left(\sum_{k=1}^{n-1} G_k + G_n\right)$$

$$= \frac{1}{n}\left(\frac{(n-1)}{(n-1)}\sum_{k=1}^{n-1} G_k + G_n\right)$$

$$= \frac{1}{n}(n-1)\, Q_n(s, a) + \frac{G_n}{n}$$

$$\boxed{\therefore\ Q_{n+1}(s, a) = Q_n(s, a) + \frac{1}{n}\left(G_n - Q_n(s, a)\right)} \quad -①$$

∴ We can say that (from ①) in order to update a state-action value function, we need its previous value, the new return value encountered at every step along with the number of times that pair (s, a) has been encountered.

- Pseudo Code for the same :-

Initialization:

$\pi(s) \in A(s)$ for all $s \in S$ with an arbitrary value

$Q(s, a) \in \mathbb{R}$ for all $s \in S$, $a \in A(s)$ with an arbitrary value

Loop forever :-

- For each episode

  Choose $S_0 \in S$, $A_0 \in A(S_0)$ randomly such that all pairs have probability of occurrence > 0.

  Using S_0, A_0; generate an episode from S_0, A_0 following π : $S_0, A_0, R_1, S_1, A_1, R_2 \dots, S_{T-1}, A_{T-1}, R_T$.

  $G = 0$

  Loop for every step of episode backwards s.t.

  $t = T-1, T-2, \dots, 0$:-

  $G = R_{t+1} + \gamma G$

  NumReturns (s, a) += 1

  Unless (s, a) appears in $S_0, A_0, \dots, S_{t-1}, A_{t-1}$

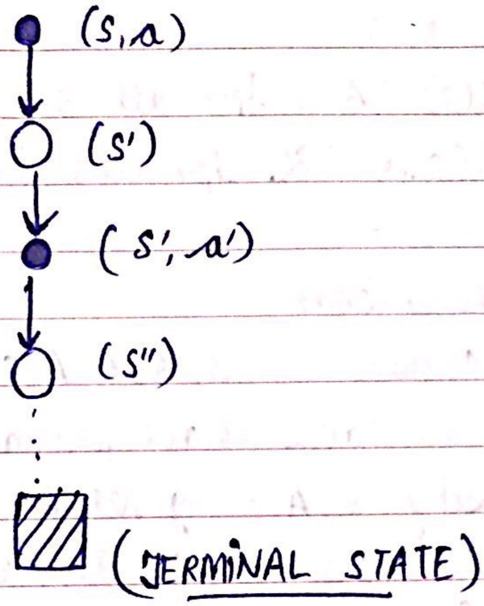
  $$Q(s,a) = Q(s,a) + \frac{1}{\text{NumReturns}}(G - Q(s,a))$$

  $$\pi(s) = \underset{a}{\mathrm{argmax}}\, Q(s, a)$$

⇒ This pseudocode represents the Monte Carlo exploring starts algorithm as per the equivalent update rule for both [as shown previously].

② Backup diagram for monte carlo estimation of q_π:-

(q_π)



● : state-action pair node

O : state node

as its represents q_π, we start with the state-action pair, followed by the next state, follows by state-action & state nodes respectively as per the generated episode for Monte Carlo.

③ $$V(s) = \frac{\sum_{t \in J(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in J(s)} \rho_{t:T(t)-1}}$$

Now, say we track all the time steps at which (s, a) state-action pair is encountered as the set $J(s, a)$

So, We get:

$$\boxed{Q(s, a) = \frac{\sum_{t \in J(s,a)} \rho_{t+1:T(t)-1} G_t}{\sum_{t \in J(s,a)} \rho_{t+1:T(t)-1}}}$$

Date: ....

such that $T(t)$ represents the first time of termination after time t and G_t represents the return from time step $t+1$ until $T(t)$.

Hence, $G_{t \in (S,A)}$ corresponds to the return for state-action pair (s, a).

Also, $\rho_{t+1:T(t)-1}$

$$= \prod_{i=t+1}^{T-1} \frac{\pi(A_i|S_i)}{b(A_i|S_i)}$$

represents the corresponding importance-sampling ratios for target policy π and behaviour policy b.

⑤ Exercise 6.2

As we choose a new start state, only the initial route changes but still some states are the same as the original problem.

Assuming that we have a lot of experience to drive home from work and because bootstrapping occurs in TD, TD will eventually perform better as the state values of the common states can be reused as they would be close to their true values as per the original problem. This would help in a faster convergence due to the initialisation from values closer to the true ones instead of being arbitrary due to bootstrapping.

Also, this would be the similar case in ~~which~~ the original situation where initial state-value estimate is closer to the true values.

⑥ Exercise 6.3

Given $\gamma = 1$ and $\alpha = 0.1$,

$V(S_t) += \alpha\,(R_{t+1} + V(S_{t+1}) - V(S_t))$

Now, if $S_{t+1} \neq$ terminal state :-

reward will be zero.

So, $V(S_t) += \alpha\,(0 + V(S_{t+1}) - V(S_t))$

Now, since we initialised $V(s) = 0.5 \ \forall\ s \in S\ \&\ s \neq$

$V(S_t) = V(S_t) \quad \forall\ s \in S\{s \to$ non terminal$\}$ terminal state :-

i.e. 0.5 & $V(S_{terminal}) = 0$

Now, in the first episode, we get a reward 0 if we end up in the terminal state on the left (A).

So, $V(A) = V(A) + 0.1\,[0 + 0 - V(A)]$

$V(A) = V(A) - 0.1\,V(A)$ $\boxed{= 0.9\,V(A)}$

$\boxed{\therefore\ V(A) = 0.9 \times 0.5 = 0.45}$

Eventually, only the state A is weighted by 0.9, hence its value decreases by 0.05.

Exercise 6.4

α represents the weightage given to the reward received at each time step. Bigger the α, more the weightage to the reward at each t.

Now, if we use a wider range of α for diff. algorithms when visualising RMSE v/s episodes.

→ As a smaller value of α leads to slower learning and at the same time smaller and smoother RMSE due to lesser oscillations, any other fixed alpha would not lead to a better performance for both the algorithms as the considered alphas already have quite small values.

Date: ....

Exercise 6·5

No. It should not be a function of the initial values & hence should not depend on Initialisation. The update eqn. for TD works in a similar manner as the gradient-descent update, taking a step in the target direction. Once, we reach convergence to the optimal values, if α would be bigger, there would be more oscillations around the optimal value converged.

(8) Exercise 6·12

Even if we make the action picking procedure greedy Q-Learning and SARSA would not be the same. This is because in Q-Learning, it first modifies the state-action value function & then chooses the action as per updated state-value function whereas for SARSA, we first choose the action as per the previous state-value function and then update it. Both scenarios can certainly lead to different simulations.