RL Assignment 3

Question 1 -

Pseudo Code for monte Carlo ES -

Prerequisites :-
T states ranging from 0 to (T-1)
An episode is generated from $S_0 A_0$ following -

$S_0 A_0 \; R_1 S_2 A_2 \; \dots \; S_{T-1} A_{T-1} R_{T-1}$
Returns $(S_t, A_t) \leftarrow []$ (Empty list)
Pseudocode :-
For each episode i.e, from $t = T-1, T-2, \dots 0$.

$G_t \leftarrow \gamma G_t + R_{t+1}$
returns
Append G to Returns (S_t, A_t)
$Q(S_t, A_t) \leftarrow$ average (Returns (S_t, A_t))
$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$

Since, $\frac{1}{T} \sum_{t=0}^{T-1} G_t$.

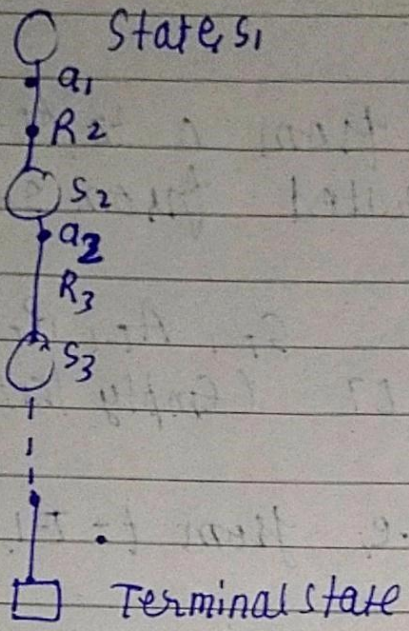
So, rewritten as - $\frac{1}{T}\left[G_{t-1} + \sum_{t=0}^{T-2} G_t\right] = \frac{1}{T}\left[G_{T-1} + \frac{1}{T-1}(T-1)\sum_{t=0}^{T-2} G_t\right]$

$= \frac{1}{T}\left[G_{T-1} + (T-1)\, Q(S_{t-1}, A_{t-1})\right]$

Recursively, $Q(S_t, A_t) = Q(S_{t-1}, A_{t-1}) + \frac{1}{T}\left[G_t - Q(S_{t-1}, A_{t-1})\right]$

Question 2 -

Backup Diagram for Monte-Carlo -



It includes entire episode, only one choice can be made at each state. It doesn't estimates on the basis of other estimates, & estimates are independent.

Equation given as -

$$V(s) = \frac{\sum_{t \in J(s)} \cdot P_{t:T(t)-1} \; G_t \cdot}{\sum_{t \in J(s)} P_{t:T(t)-1}}$$

for $Q(s,a)$, it will be -

$$Q(s,a) = \frac{\sum_{t \in J(s,a)} P_{t:T(t)-1} \; G_t}{\sum_{t \in J(s,a)} \cdot P_{t:T(t)-1}}$$

where, $P_{t:T(t)-1}$ is important sampling ratio
G_t is expected return at time t.

Question 3

Equation given as -

$$V(s) = \frac{\sum_{t \in T(s)} P_{t:T(t)-1} \, G_t}{\sum_{t \in T(s)} P_{t:T(t)-1}}$$

for $Q(s,a)$, it will be -

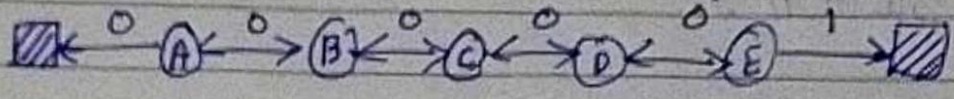
$$Q(s,a) = \frac{\sum_{t \in T(s,a)} P_{t:T(t)-1} \, G_t}{\sum_{t \in T(s,a)} P_{t:T(t)-1}}$$

where, $P_{t:T(t)-1}$ is important sampling ratio
G_t is expected return at time t.

Question 5:-
TD learning would be better than MC learning in the case where we are moved to a new building since it is just a change in our initial route & some of the state encountered during general episode will be same.

For Example :- On our drive home many of states are the same once we enter the highway & value function estimates for these states obtained when we worked in original building should be very close to what we will compute starting from our new building. Starting with a very good initial guess should result in faster convergence.
This will happen in our original learning task if our initial guess at value function is very close to that of true value function.

Question 6.3



Since, $\alpha = 0.1$, & $\gamma = 1$.
Then,
TD expression for state value function will be -

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

$$= V(S_t) + 0.1(r_{t+1} + V(S_{t+1}) - V(S_t))$$

Initial value function begin with constant value So, in first update, there is no change in value function.
If we terminate on left the reward is zero, so state A will be updated.

$$V(A) = V(A) + 0.1(r_{t+1} + V(A_{t+1}) - V(A))$$
$$= V(A) + 0.1(0 + 0 - V(A))$$
$$= 0.5 + 0.1(-V(A))$$
$$= 0.45.$$

Thus, we go to left of random walk, then we decrease the state value function by 0.05.

Question 6.4

The Right figure (graph) in Example 6.2 is dependent on value of α.
Since, when α will be small, the convergence of Monte Carlo & Temporal Difference is satisfied. Small value of α is better than large value in long run, And so, learning will be more & finally reach to lowest error.
Therefore, we cannot conclude which algorithm will perform better if wider range of α is used.

Question : 6.5
It is given that α is large enough, so, large value of α causes more change in state-value function for each time-stamp.
Thus, TD(0) is heavily dependent on specific returns. So, initially graph of RMS error goes down because learning rate is high & then up.
At smaller value of α, learning will take longer & it is less sensitive to specific random step as compare to large value of α. While on other hand, if initialization of initial value to state generates linear relations about updates during transition, there may chance, it over estimates result around terminal state.

Question - 8

No, it will not take same selection because Q-learning will update Q-function value first, As Q-learning is off-policy algorithm.

In Q-learning, the next action is to perform to selected in next iteration derived from q-function while SARSA is on-policy algorithm, it chooses a', s' & then update Q-value.

Q-learning Algorithm :-
Initialise Q-table with ones
for each episode
    Initialise s
    for each step of Episode
    Choose a from s using greedy policy
    Take action a, observe r, s'.
    $Q[s,a] \leftarrow Q[s,a] + \alpha [r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
    $s \leftarrow s'$
until s is terminal

SARSA Algorithm :-
Initialise Q-table with ones
for each episode
    Initialise state, s
    Choose action from state using policy derived from Q (greedy approach)
    For each step of episode

Take action, observe r, s'.
Choose a' from s' & update Q-table
$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma(Q(s',a') - Q(s,a))$
$s \leftarrow s'$
$a \leftarrow a'$
until s is terminal.