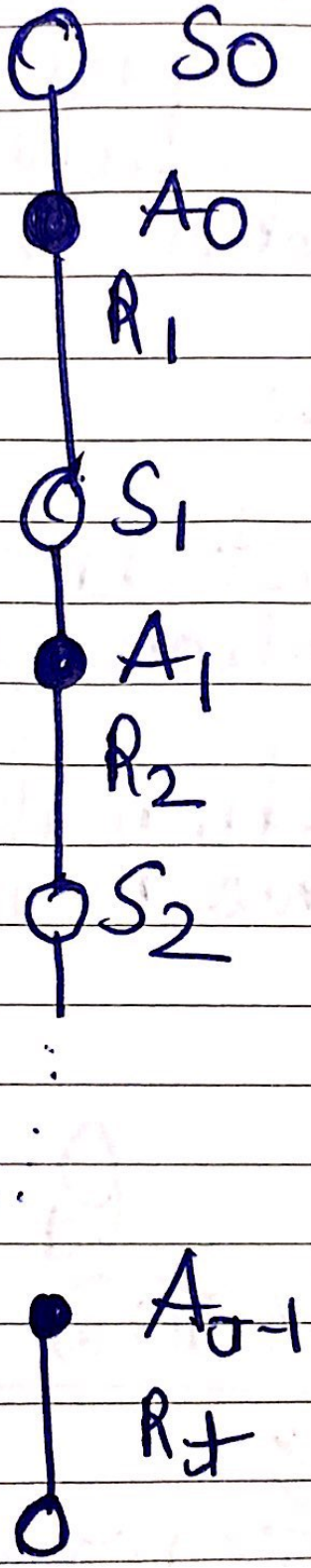
Date ..... / ..... / .....

Question 2:

The backup diagram for Monte Carlo will show all states till the end of the episode.



Date .....

Question 3

Following notation given in the TB,

$\mathcal{T}(s,a)$ ⇒ set of all time steps in which we arrived at state s and picked action.

~~$T(t)$ denote~~

$T(t)$ ⇒ first time of terminal following time t.

G_t ⇒ return after t up through $T(t)$.

So to

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

Date ...../...../.....

Question 5.

The given problem states that there is only some part of the path that is charged. And since we have a lot of experience driving in the path that is constant, we just have to learn from parking lot to highway.

TD algorithm will take advantage of this, ~~and, hence~~ so by using state values of the constant path. and here be better off initially.

In the original scenario, if our initial estimates are close to the true values, then also TD update would be better on average than Monte Carlo.

Date ..... /..... / .....

Question 8:

[redacted]

No, Q learning is not same as Sarsa.
At its base, Q-learning is an off-policy and Sarsa is on-policy.

Sarsa chooses a' and s' and then updates Q-function where Q-learning first updates Q-function, and next action is selected after. These may not be the same.

Date ...../...../.....

Question 1:

Update of state-action pairs happens as follows:

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

$$= \frac{1}{n}\left(\sum_{j=1}^{n-1} q_j + q_n\right)$$

$$= \frac{1}{n}\left((n-1)\times\left[\frac{1}{n-1}\sum_{j=1}^{n-1} q_j\right] + q_n\right)$$

$$Q_{n+1}(s,a) = \frac{1}{n}\left((n-1)\cdot Q_n(s,a) + q_n\right)$$

If we use this update rule, we just need to store the previous value for each state action pair instead of n rewards.

Date ...../...../.....

The pseudocode will be,

- initialize [redacted] array(S,a)

Loop Forever (each episode):
choose $S_0 \in S$, $A_0 \in A(S_0)$ randomly such that all pairs have non zero probability.

Generate an episode starting at S_0, A_0 and following π.

$G = 0$

Loop for each step, $t = T-1, T-2, \ldots, 0$:

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

array(S,a) $T = 1$

unless the pair S,a appears before.

$$Q(S,a) = Q(S,a) + \frac{1}{\text{array}(S,a)} \times (G - Q(S,a))$$

$$\pi(S) \in \arg\max_{a} Q(S,a)$$

Date ...../...../.....

We have only changed the update step in the code and rest everything remains the same. Here, both codes are equivalent.

Question 6:

6.3

Under the given conditions, update rule is

$$v(S_t) = v(S_t) + 0.1\left[R_{t+1} + v(S_{t+1}) - v(S_t)\right]$$

We initial $v(S_t) = 0.5 \quad \forall s \in S$,

Any transition from $S_t \Rightarrow S_{t+1}$ does not result in an update because,

$$v(S_t) = v(S_t) + 0.1[0 + 0.5 - 0.5)$$

Date ..... / ..... / .....

However, in the first episode we went from state A, to left terminal state with value is 0

$$v(A) = v(A) + 0.1[0 + 0 - v(A)]$$

$$= 0.9\, v(A)$$

$$v(A) = 0.45.$$

Here, only $v(A)$ got updated.

6.4

Yes, if we use large value of α, then TD will perform worse than monte carlo.

Because large value of α will lead to higher sensitivity towards a specific rewards in a time step. This might result in estimates that are way off.

6.5

If we increase the value of α greatly, the update is much larger at each timestep.

This makes the TD algorithm much more sensitive to the rewards received at that particular step.

What we notice in the graph is because of the randomly selected step in the random walk initially. The disturbance are because of high sensitivity. If we use a lower value of α, these disturbances are reduced.

Forget the mistake, remember the lesson.