(Q1) Calculating the mean part is inefficient as every time a return is appended, average is taken over all the previous returns. This increases the amount of storage as all returns for all states action pairs need to be stored.

Maintain a count(St, At) variable and change the part of the code where average is taken.

Pseudocode

Unless the pair St, At appears in S0, A0 ... St-1, At-1:

$$Q(S_t, A_t) \leftarrow \frac{Q(S_t, A_t) \times count(S_t, A_t) + G}{count(S_t, A_t) + 1}$$

$$count(S_t, A_t) \leftarrow count(S_t, A_t) + 1$$

$$\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$$

$Q(S_t, A_t) \times count(S_t, A_t)$ → gives the sum of previous returns.

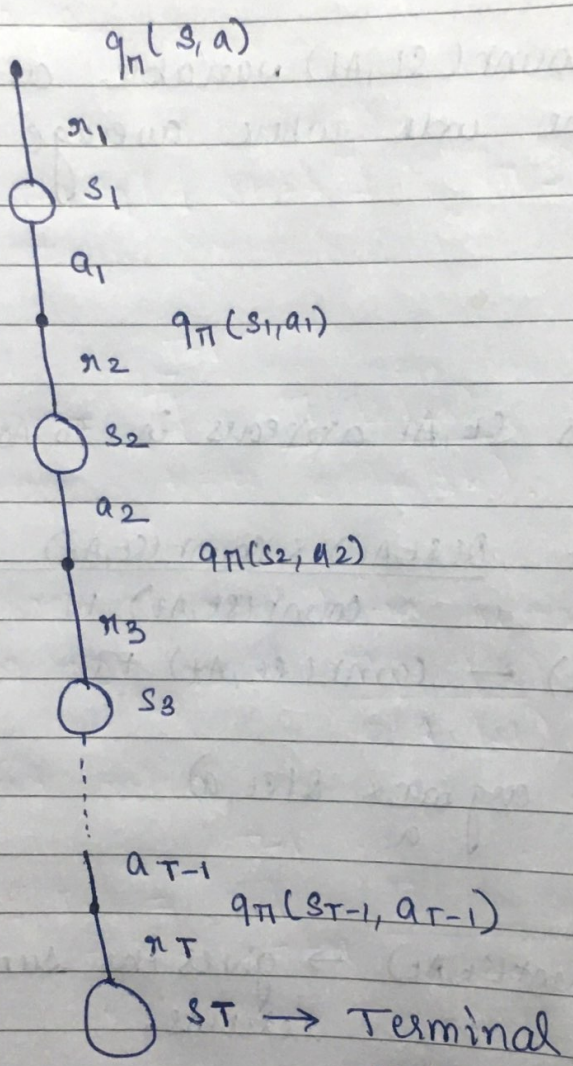
G is added

Now total no. of visit to this pair increases by 1, hence we divide by count(St, At) + 1.

(Q2) Backup diagram MC estimation of q_π.

• → represents action

○ → represents state



(Q3) $\mathcal{T}(s)$ → Time steps that were first visit to s in an episode

Introduce $\mathcal{T}(s,a)$ → Time steps that were first visit to s and taking action a in an episode.

$$E[\rho_{t:T-1} G_t \mid S_t = s] = V_\pi(s).$$

where

$$\rho_{t:T-1} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}.$$

since we already have initial actions, we do not need $\frac{\pi(A_t|S_t)}{b(A_t|S_t)}$, so our modified ratio,

$$\rho'_{t:T-1} = \prod_{k=t+1}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

$$E[\rho'_{t:T-1} G_t \mid S_t = s, A_t = a] = q_\pi(s,a).$$

$$q_\pi(s,a) = \frac{\sum_{t \in \mathcal{T}(s,a)} \rho'_{t:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s,a)} \rho'_{t:T-1}}$$

(Q5) ~~Suppose~~ As given in the hint, consider the scenario where you move to a new office but while returning home, you enter the same highway.

Since you had a lot of experience entering that highway, you know average state value i.e. the average time it takes you from highway to your home.

Thus when you need to estimate the time from your new office to home, using bootstrapping (idea behind TD), you can estimate the time accurately by observing uptill the highway and then update the time using the value from highway to home.

In case of monte carlo, this was not possible, as the whole episode was needed again to estimate the correct time.

Q6) 6.3

There are 2 possible cases after first episode, episode ends in left terminal state or right terminal state.

for all states B, C and D,

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

the S_{t+1} state also ~~now~~ has value 0.5 so $V(S_{t+1}) - V(S_t) = 0$, $R_t = 0$, therefore, no update takes place.

if episode ended in right terminal state,

$$V(E) \leftarrow V[E] + \alpha\,(R_{t+1} + V(S_T) - V(E))$$

$V(S_T) = 0$
$R_{t+1} = 1$
$\alpha = 0.1$

$$V[E] \rightarrow 0.5 + 0.1\,(1 + 0 - 0.5)$$

$$\rightarrow 0.5 + 0.1\,(0.5)$$
$$\rightarrow 0.55$$

if episode ended in left terminal state,

$$V(A) \leftarrow V(A) + \alpha (R+H + V(ST) - V(A))$$

$\alpha \rightarrow 0.1$

$R+H \rightarrow 0$

$V(ST) \rightarrow 0$

$$V(A) \leftarrow 0.5 + 0.1(0 + 0 - 0.5)$$

$$V(A) \leftarrow 0.5 - 0.05$$

$$V(A) \leftarrow 0.45$$

6.4

Q8) if action ~~select~~ selection is greedy, then Q-learning will be same as sarsa, as the behaviour and target policy are same.

In Q learning,

we choose action A from s using ϵ-greedy, observe R, s'. The updation of Q(s,A) takes place with the action ~~of~~ value Q(s',A') where A' is according to the greedy policy.

$$Q(s,A) \leftarrow Q(s,A) + \alpha [R + \gamma \max_a Q(s',a) - Q(s,A)]$$

$$s \leftarrow s'$$

At the next time step, when choosing Action from s' with greedy policy we will get action A' only. Thus behaviour and target remain the same.

In sarsa,

Action A is chosen from state s, s' and R are observed. Action A' is chosen from s' and then ~~pola~~ values updated,

$S \leftarrow S'$, $A \leftarrow A'$, so for the next time step we have to choose A'.

Thus if greedy policy is used, then Q learning is same as Sarsa.