Q-Learning vs SARSA

Describe the differences existing between the Q-Learning and SARSA algorithm

SARSA and Q-Learning are two algorithm used to o control using the model free method called temporal difference.

Q-learning is an example of off-policy learning which means that it learns a target-policy (denoted by π) while following a behavioral policy $\overline{\pi}$. It means that we are, for example, using old policies to learn a new policy, or we are learning a new policy from observing other agents.

Q-Learning Update Function:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\big(R_{t+1} + \gamma \max_{a' \in A} Q(S_{t+1}, a') - Q(S_t, A_t)\big)$$

π is greedy (there is a max) and $\overline{\pi}$ is ϵ-greedy.

SARSA is an example of on-policy learning, so it learns the optimal policy based on the actions performed following its own policy. It samples A_{t+1} from its own policy.

SARSA Update Function:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$

The two algorithms can perform differently in given situations, for example, in class, we have seen the cliff walking problem. Q-Learning learns an optimal policy along the edge of the cliff because the behavioral policy is ϵ-greedy; while SARSA learnt a safe non-optimal policy away from the edge. This means that if we adopt an ϵ-greedy behavioral policy for Q-Learning, and we use an ϵ-greedy policy for SARSA, we have that:

- If $\epsilon \neq 0$ SARSA performs better online
- if $\epsilon \to 0$ gradually both converge to the optimal policy.

UCB1 Algorithm

Describe the UCB1 algorithm. Is it a deterministic or a stochastic algorithm?

UCB1 is an algorithm to solve stochastic MAB problems through a frequentist approach.
In order to evaluate the expected reward of an arm, that we assume distributed as a Bernoulli, we do the following:

Evaluate for each arm

- its sample reward given its past rewards
- a bound

and then pick the arm whose $\hat{R} + B$ is the highest. Repeat the process.

In formulas:
For each time step t:

1. Compute $\hat{R}_t(a_i) = \frac{\sum_{i=1}^{t} r_{i,t} \mathbf{1}\{a_i = a_{i_t}\}}{N_t(a_i)} \; \forall a_i$
2. Compute $B_t(a_i) = \sqrt{\frac{2 \log t}{N_t(a_i)}} \; \forall a_i$
3. Play arm $a_{it} = \arg\max_{a_i \in A} \left(\hat{R}_t(a_i) + B_t(a_i) \right)$

Upperbound

Theorem:

At finite time T, the expected total regret of the UCB1 algorithm applied to a stochastic MAB problem is

$$L_t \leq 8 \log T \sum_{i | \Delta_i > 0} \frac{1}{\Delta_i} + \left(1 + \frac{\pi^2}{3}\right) \sum_{i | \Delta_i > 0} \Delta_i$$

where $\Delta_i = R^* - R(a_i)$, and R^* is the reward obtained by performing the best action.

the first term is our expected loss, and the second is our risk.

Since the choice of the arm to be pulled is deterministic, it's a deterministic algorithm.