AI Planning for Autonomy

Sample Solutions for Problem Set IX: MDPs and Reinforcement Learning

1. The difference between SARSA and Q-learning is that Q-learning is "off-policy" learning, while Sarsa is "on-policy" learning. Essentially, this means that SARSA chooses its action using the same policy used to choose the previous action, and then uses this difference to update its Q-function; while Q-learning simply chooses the next value based on the maximum Q-value.

   Q-learning is therefore 'optimistic', in that when it updates, it assumes that in the next state, the 'best' (greedy) action will be chosen, even it may be that in the next step, the policy, such as ϵ-greedy, will choose to explore an action other than the best.

   SARSA instead knows the action that it will execute next when it performs the update, so will learn on the action whether it is best or not.

2. For Q-learning, this is calculated as:

$$\begin{aligned} Q(S,P) &= Q(S,P) + 0.4 \cdot [r(S,P) + 0.9 \cdot \max_{a' \in A(M)} Q(M,a') - Q(S,P)] \\ &= -0.7 + 0.4 \cdot [(-1) + 0.9 \cdot (-0.4) - (-0.7)] \\ &= -0.7 + 0.4 \times (-0.66) \\ &= -0.964 \end{aligned}$$

3. For SARSA, this is calculated as:

$$\begin{aligned} Q(S,P) &= Q(S,P) + 0.4 \cdot [r(S,P) + 0.9 \cdot Q(M,\pi(M)) - Q(S,P)] \\ &= -0.7 + 0.4 \cdot [(-1) + 0.9 \cdot (-0.8) - (-0.7)] \\ &= -0.7 + 0.4 \times (-1.102) \\ &= -1.108 \end{aligned}$$

4. For 3-step SARSA is calculated as:

   $Q(s,a) = Q(s,a) + \alpha[G_t^n - Q(s,a)]$

   $G_t^n = r_t + \gamma \cdot t_{t+1} + \gamma^2 \cdot r_{t+2} + \cdots + \gamma^n \cdot Q(S_{t+n}, \pi(S_{t+n}))$

$$\begin{aligned} Q(S,P) &= Q(S,P) + 0.4 \cdot [G_S^3 - Q(S,P)] \\ &= -0.7 + 0.4 \cdot [-1.4716 - (-0.7)] \\ &= -0.7 + 0.4 \times (-0.7716) \\ &= -1.00864 \end{aligned}$$

   where

$$\begin{aligned} G_S^3 &= r(S,P) + 0.9 \cdot r(M,S) + 0.9^2 \cdot r(Scored,R) + 0.9^3 \cdot Q(M,P) \\ &= (-1) + 0.9 \cdot (-2) + 0.9^2 \cdot 2 + 0.9^3 \cdot (-0.4) \\ &= (-1) + (-1.8) + 1.62 + (-0.2916) \\ &= -1.4716 \end{aligned}$$

   And basically yes, it can converge much faster than 1-step.