Exercises for week 5
Reinforcement Learning: Q-value and SARSA

Exercise 1: SARSA algorithm

In the lecture, we introduced the SARSA (state-action-reward-state-action) algorithm, which (for discount factor $\gamma = 1$) is defined by the update rule

$$\Delta Q(s,a) = \eta \left[r - \big(Q(s,a) - Q(s',a') \big) \right] , \tag{1}$$

where s' and a' are the state and action subsequent to s and a. In this exercise, we apply a greedy policy, i.e., at each time step, the action chosen is the one with maximal expected reward, i.e.,

$$a^*_t = \arg\max_a Q_t(s,a) \, . \tag{2}$$

If the available actions have the same Q-value, we take both actions with probability 0.5.

Consider a rat navigating in a 1-armed maze (=linear track). The rat is initially placed at the upper end of the maze (state s), with a food reward at the other end. This can be modeled as a one-dimensional sequence of states with a unique reward ($r = 1$) as the goal is reached. For each state, the possible actions are going up or going down (Figure 1). When the goal is reached, the trial is over, and the rat is picked up by the experimentalist and placed back in the initial position s and the exploration starts again.

a. Suppose we discretize the linear track by 6 states, $s_1, \ldots, s_6$ where s_1 is the initial state and s_6 is the goal state. Initialize all the Q-values at zero. How do the Q-values develop as the rat walks down the maze in the first trial?

b. Calculate the Q-values after 3 complete trials. How many Q-values are non-zero? How many trials do we need so that information about the reward has arrived in the state just 'below' the starting state?

c. What happens to the learning speed if the number of states increases from 6 to 12? How many Q-values are non-zero after 3 trials? How many trial do we need so that information about the reward has arrived in the state just 'below' the starting state?

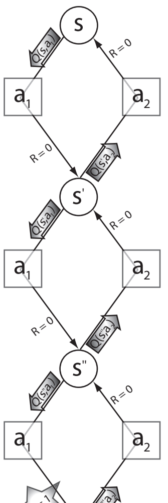


Figure 1: A linear maze.

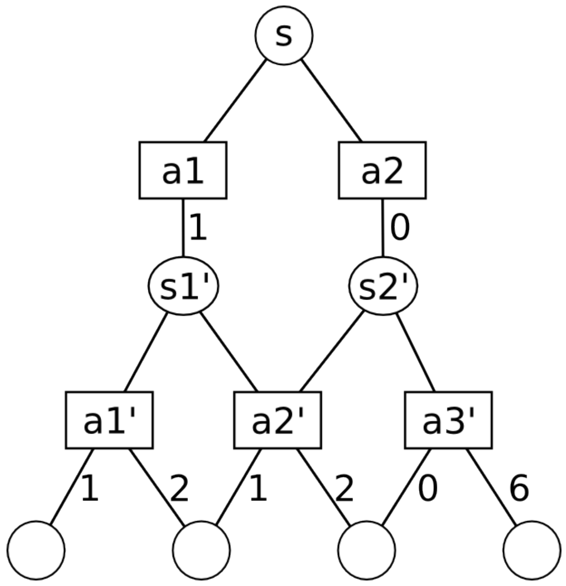


Figure 2: A tree–like environment.

Exercise 2: Bellman equation

Use the Bellman equation to calculate $Q(s, a1)$ and $Q(s, a2)$ for the environment shown in Figure 2. Consider two different policies:

- Total exploration: All actions are chosen with equal probability.
- Greedy exploitation: The agent always chooses the best action.

Note that the rewards/next states are stochastic for the actions $a1'$, $a2'$ and $a3'$. Assume that the probabilities for the outcome of these actions are all equal, and the discount factor γ is 1.

Exercise 3: Expected SARSA and a new variant of TD learning

We have seen the algorithm 'Expected Sarsa' and 'TD-algorithm in the narrow sense'. Suppose you invent a new algorithm that keeps tables of both Q-values and V-values. To do so consider the following:

a. Rewrite the updated step of 'Expected Sarsa' using both Q-values and V-values in the update rule.

b. Write down the full pseudo-algorithm.

   Hint: (i) keep track of the sequence of update steps of Q-values and V-values in your new algorithm. (ii) At which point in the sequence of online steps through the graph can you update $Q(s, a)$ and how far do you have to look backward?

c. Sketch the 'back-up-diagram' of your algorithm and compare it to that of SARSA. What are the costs and benefits of your new algorithm compared to SARSA?

Exercise 4: Monte Carlo versus expected SARSA[3]

We compare two algorithms: The Batch-Monte-Carlo algorithm (around slide 47), and the Online-Expected-SARSA (around slide 19). The aim of the exercise is to understand why bootstrap algorithms (i.e., TD algorithms) perform, under some conditions, more efficiently than a naive estimation of Q-Values via Monte-Carlo algorithms. We set the discount factor to $\gamma = 1$ and run 5 episodes in a given environment:

5 episodes, first action is always a1.

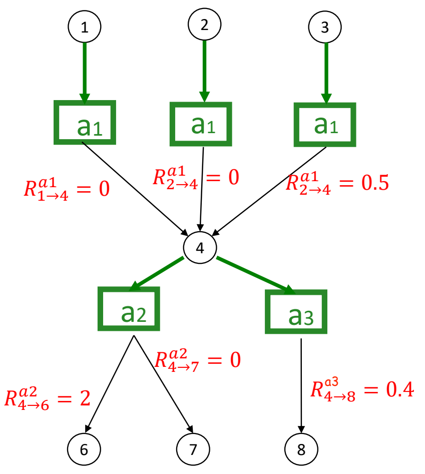
Episode 1: States 1-4-7 with action a2, Return=0

Episode 2: States 1-4-8 with action a3, Return=0.4

Episode 3: States 2-4-6 with action a2, Return=2

Episode 4: States 2-4-8 with action a3, Return=0.4

Episode 5: States 3-4-7 with action a2, Return=0.5



Each episode starts in one of the states 1 or 2 or 3 with action a_1. In state 4 there is a choice between actions a_2 and a_3 which are taken with equal probability $\pi = 0.5$. The transition sequence and total

[3]The result of Exercise 4 will be used in the second lecture of this week.

return for each of the 5 episodes is given in the figure above. All rewards are deterministic and only depend on the transition (s, a, s').

a. Calculate the Q-values in state 1, 2, 3, and 4 using Batch-Monte-Carlo (i.e., average total returns from each starting state).

b. Calculate the Q-values in state 1, 2, 3, and 4 using Online-Expected SARSA. For a given Q-value $Q(s, a)$, use $\eta_1 = 1$ the FIRST TIME you update this value and $\eta \in [0, 0.5]$ for all later update steps.

   Hint: You can neglect terms of order η^2.

c. You can choose the initial state for episode 6. Which initial state looks best in case a (Batch-Monte-Carlo)? Which initial state looks best in case b (Online-Expected SARSA)?

   Where does the difference come from? Which solution is closer to the exact Q-values under the $\pi = 0.5$ policy in state 4? Why?

d. What happens qualitatively if the order of episodes 1 and 2 were switched? What would be the value of $Q(s = 1, a_1)$ in this case? How would the other Q-values change?

Exercise 5: Computer exercises: Environment 1 (part 1)

Download the Jupyter notebook of the 1st computer exercise, setup your Python environment, and complete ‘Exercise 0: One step horizon’.