Reinforcement Learning
Assignement - Value-Based RL Methods

In this assignment we will use RL techniques first to compute value functions and then to optimize the policy. We will use simple MDPs in order to understand more easily how RL works.

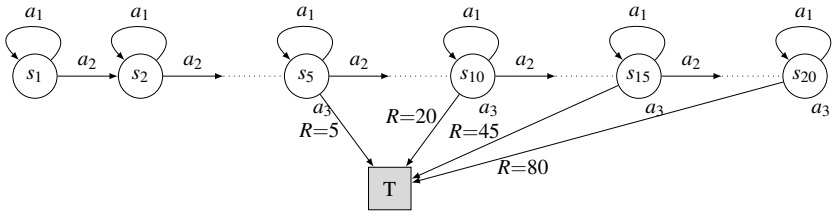

Figure 1: Cycles MDP. There are twenty states in a row and three actions: a_1 leaves the state unchanged; a_2 leads to the state to the right, except for s_{20}, in which a_2 does not exist; a_3 only exists in states $s_5, s_{10}, s_{15}, s_{20}$ and leads to s_1. The reward is always 0 except when taking action a_3.

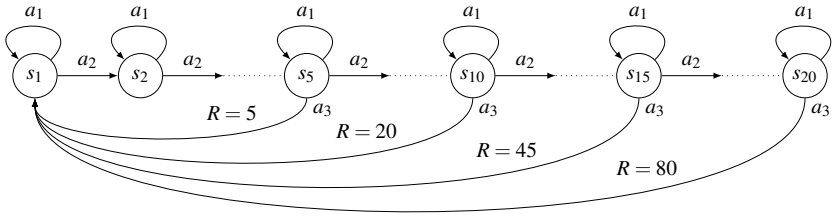

Figure 2: Cycles MDP. There are twenty states in a row and three actions: a_1 leaves the state unchanged; a_2 leads to the state to the right, except for s_{20}, in which a_2 does not exist; a_3 only exists in states $s_5, s_{10}, s_{15}, s_{20}$ and leads to s_1. The reward is always 0 except when taking action a_3.

1. **Cycles MDP**

   Consider the modification of the "Cycles" MDP from the first assignment depicted in Figure 1.

   Consider two deterministic policies: π_1 always selects a_2, except when in $s_5, s_{10}, s_{15}, s_{20}$, where it selects a_3; π_2 always selects a_2, except when in s_{20}, where it selects a_3. In the following, we will consider discount factor $\gamma = 0.99$.

   (a) Implement every-visit MC to compute $\hat{V}^{\pi_1}$, $\hat{V}^{\pi_2}$. Verify that you do obtain the correct solution.

   (b) Perform the same computations using TD(0)

   (c) Now apply TD(0) to the original cycles MDP depicted in Figure 2. Compare your solution to the exact one you computed in the previous assignment

   (d) **Optional** Perform the same computations of the two previous points using backward TD(λ) using eligibility traces with $\lambda = 0.9$

   (e) **Optional** Implement Monte-Carlo control with an ε-greedy policy and $\varepsilon = 0.1$

   (f) Implement SARSA and Q-learning using an ε-greedy policy and $\varepsilon = 0.1$

   (g) **Optional** Implement backward SARSA(λ) and Q(λ) with eligibility traces with $\lambda = 0.9$, using an ε-greedy policy and $\varepsilon = 0.1$

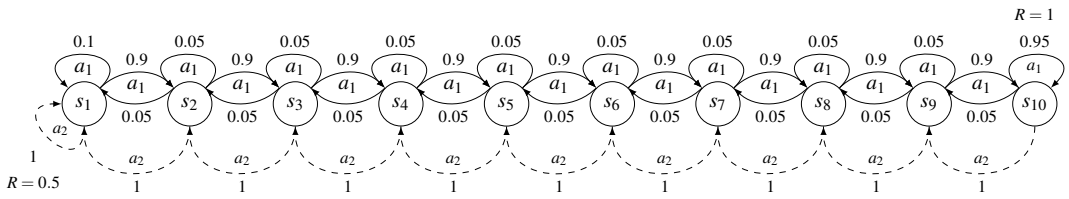

Figure 3: RiverSwim MDP. There are ten states in a row and two actions: a_1 leads to the state to the right with probability 0.9, to the same state with probability 0.05 and to the state to the left with probability 0.05; a_2 always leads to the state to the left. The reward is always 0, except $R(s_{10},a_1) = 1$ and $R(s_1,a_2) = 0.5$.

2. **River Swim**

   Consider the River Swim MDP depicted in Figure 3, with discount factor $\gamma \in \{0.8, 0.9, 0.99\}$.

   (a) Implement Q-learning to compute the optimal policy and value function. What do you observe?
   Hint: use an ε-greedy policy. You can explore several values of ε and observe how they impact on learning.

   (b) Implement SARSA to compute the optimal policy and value function. What do you observe?

   (c) **Optional** Implement backward Q(λ) with eligibility traces with $\lambda = 0.9$. What do you observe?

3. **LQR**: **Optional**

   Consider a very simple LQR problem, defined as

$$s_+ = \boldsymbol{A}s + \boldsymbol{B}a + w \qquad\qquad R(s,a) = s^\top \boldsymbol{Q}s + a^\top \boldsymbol{R}a,$$

   with $w \sim \mathcal{N}\left(0, 10^{-4}\right)$, $\boldsymbol{A} = 1.1$, $\boldsymbol{B} = 1$, $\boldsymbol{Q} = 0.5$, $\boldsymbol{R} = 1$, and discount factor $\gamma = 0.9$.

   (a) **Optional** Implement Q-learning to solve the LQR problem.