Exercise Sheet 12

Exercise 1

Derive the recursive formulation

$$Q(s,a) = \rho(s,a) + \gamma \max_{a' \in \mathcal{A}} Q(\delta(s,a), a').$$

Exercise 2

(a) Implement Q-Learning with for the Frozen Lake Environment of the Lecture with the following modifications:

1. Use the ϵ-greedy policy as described below for action selection.
2. Moves to a rock cell or outside of the grid bounce the agent back to its current frozen cell and are rewarded with -1.
3. Rewards are
   - -1 for a move to a save frozen cell,
   - -100 for a move to a thin ice cell,
   - $+100$ for a move to the exit cell.
4. The maximum number of steps within an episode is $T = 100$.

(b) Apply your implementation and run Q learning for a sufficiently large number of episodes. Plot the number of moves and the result in each episode. Results are

- $+1$ when the agent successfully found the exit
- ± 0 when the agent is still alive after T steps
- -1 when the agent has drowned.

(c) Periodically evaluate your agent's performance after every k-th episode. To do this, keep track of the number of moves and results as in part (b), for each possible initial state corresponding to a frozen cell. After each k-th episode, display the count of wins, draws, and losses, along with the average number of moves taken during wins.
Note: Make sure that the action selection during evaluation follows Equation (1), as described below in the section on the ϵ greedy policy.

(d) Compare Q-learning with and without the ϵ-greedy policy.

The ϵ-Greedy Policy

In the lecture, the agent selects the next action as follows:

$$\pi(s) = \text{argmax}_a \, q(s, a), \tag{1}$$

where the function $q(s, a)$ represents the estimated value (quality) of taking action a in state s. This action selection policy can be problematic because it always chooses the action with the highest estimated reward, leading to a lack of exploration of other actions that might actually have higher rewards but haven't been sufficiently explored yet.

The ϵ-greedy policy is an action selection strategy that alleviates the problems of choosing the next action according to Equation (1). It works as follows:

1. With probability $1 - \epsilon$, choose the action a according to Equation (1). This is known as **exploitation**.
2. With probability ϵ, choose an action uniformly at random from the set of all possible actions. This is known as **exploration**.

The idea behind the ϵ-greedy policy is to balance exploration and exploitation. Exploitation makes the best decision given current knowledge, while exploration gathers more knowledge at the risk of making a suboptimal decision.

The parameter ϵ controls the trade-off between exploration and exploitation. A high ϵ, as typically used in the initial phase of learning, encourages more exploration (trying out different actions to see their effects), while a low ϵ, as typically used in the final phase of learning, encourages more exploitation (choosing the actions that are known to yield high rewards). This balance allows the agent to learn an optimal policy over time.

Implement an exponential annealing schedule for the exploration rate ε. Begin with an initial value for ε. After each episode, reduce ε according to the following rule:

$$\varepsilon \leftarrow \delta \cdot \varepsilon, \tag{1}$$

where δ represents the decay rate. Continue to decrease ε until it reaches a specified minimum value ε_{min}. You can use the following schedule:

- $\varepsilon = 0.9$ as initial value
- $\delta = 0.95$ as decay rate
- $\varepsilon_{\text{min}} = 0.05$ as minimum value