# Reinforcement Learning Basics

## 1 Basic Reinforcement Learning Terms

Here are some fundamental terms in reinforcement learning (RL) and their meanings:

- **Agent:** The learner or decision-maker in the environment. The agent takes actions based on the current state of the environment and learns from the rewards or punishments received to maximize the total cumulative reward over time.
- **Environment:** The external system or world in which the agent operates. The environment provides the agent with states and rewards, and reacts to the agent's actions by transitioning to new states.
- **State:** A representation of the current situation of the environment. It contains all the necessary information required by the agent to decide on an action. For example, in a chess game, the state could be the positions of all pieces on the board.
- **Action:** A decision or move made by the agent. Depending on the current state, the agent selects an action from a set of possible actions. For instance, in a robotic arm, the action could be moving the arm up or down.
- **Reward:** A scalar feedback signal received after taking an action in a particular state. The reward indicates the immediate benefit or cost of the action taken. The goal of the agent is to maximize the cumulative reward over time.
- **Policy ($\pi$):** A policy is a strategy used by the agent to decide which action to take in each state. It can be deterministic ($\pi(s) = a$) or stochastic ($\pi(a|s) =$ probability of taking action $a$ in state $s$).
- **Value Function ($V$):** The value function estimates the expected cumulative reward of a state, considering future rewards. It helps the agent determine the overall desirability of a state.
- **Action-Value Function ($Q$):** The action-value function, or $Q$-function, estimates the expected cumulative reward of taking a particular action in

a particular state and following a specific policy thereafter. $Q$-values help the agent in selecting the optimal actions.

- **Discount Factor ($\gamma$):** A parameter between 0 and 1 that determines the importance of future rewards. A discount factor close to 1 values future rewards almost as much as immediate rewards, while a factor close to 0 prioritizes immediate rewards.
- **Exploration vs. Exploitation:**
  - **Exploration:** The agent tries out new actions to discover their effects and improve its understanding of the environment.
  - **Exploitation:** The agent uses its current knowledge to take the best-known action to maximize rewards. Balancing exploration and exploitation is crucial for effective learning.
- **Episode:** A sequence of states, actions, and rewards, ending when a terminal state (goal or failure) is reached. An episode represents a complete run from the initial state to the terminal state.
- **Trajectory:** A path or sequence of states and actions taken by the agent from the beginning to the end of an episode.
- **Markov Decision Process (MDP):** A mathematical framework for modeling decision-making, representing the environment in terms of states, actions, rewards, and transition probabilities. An MDP assumes that the future state depends only on the current state and action, not on past states (Markov property).
- **Temporal Difference (TD) Learning:** A class of model-free learning methods that update value estimates based on the difference (temporal difference) between successive estimates. TD learning combines aspects of Monte Carlo methods and dynamic programming.
- **Learning Rate ($\alpha$):** A parameter that determines the extent to which new information overrides old information. It controls the speed of learning in algorithms like Q-learning.

These terms form the foundation of understanding reinforcement learning.

# 2 Q-Learning Basics

Q-learning is a popular reinforcement learning algorithm used for training agents to learn optimal policies in Markov Decision Processes (MDPs). Below are the basic concepts of Q-learning.

## 2.1 Q-Value or Action-Value ($Q$)

The $Q$-value, denoted as $Q(s, a)$, represents the expected cumulative reward the agent will receive after taking action $a$ in state $s$, and then following the optimal policy thereafter. It quantifies the value of a specific action in a specific state.

## 2.2 Q-Table

A table (or matrix) that stores the $Q$-values for every state-action pair. For each state $s$ and action $a$, the table contains a value $Q(s, a)$. The table is updated iteratively as the agent interacts with the environment.

## 2.3 Bellman Equation

Q-learning uses the Bellman equation to update $Q$-values:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right] \tag{1}$$

where:

- $s$: Current state
- $a$: Action taken in state $s$
- $r$: Reward received after taking action $a$
- $s'$: Next state after taking action $a$
- $\alpha$: Learning rate (controls how much new information overrides old information)
- $\gamma$: Discount factor (how much future rewards are valued compared to immediate rewards)
- $\max_{a'} Q(s', a')$: Maximum $Q$-value for the next state $s'$ over all possible actions $a'$

## 2.4 Q-Learning Update Rule

The update rule is applied every time the agent takes an action and receives a reward. It works by adjusting the $Q$-value of the current state-action pair based on the observed reward and the estimated optimal future $Q$-value.

## 2.5 Exploration vs. Exploitation

- **Exploration**: The agent tries out different actions to discover their effects and to improve its $Q$-value estimates.
- **Exploitation**: The agent selects the action with the highest $Q$-value for the current state, leveraging its learned knowledge to maximize rewards.

The $\epsilon$-greedy strategy is commonly used to balance exploration and exploitation. The agent chooses a random action with probability $\epsilon$ (exploration), and the best-known action with probability $1 - \epsilon$ (exploitation).

## 2.6 Algorithm Steps

1. **Initialize $Q$-values**: Start with a $Q$-table initialized to zeros (or random values).
2. **Repeat for each episode**:
   (a) Initialize the starting state $s$.
   (b) Repeat for each step in the episode until the terminal state is reached:
      i. Choose an action $a$ in state $s$ using the $\epsilon$-greedy policy.
      ii. Take the action $a$, observe reward $r$, and next state $s'$.
      iii. Update the $Q$-value using the formula:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right] \tag{2}$$

      iv. Set the new state $s$ to $s'$.
   (c) End the episode when a terminal state is reached.

## 2.7 Convergence

Q-learning is guaranteed to converge to the optimal policy, assuming all state-action pairs are explored infinitely often and the learning rate is decayed appropriately over time.

## 2.8 Advantages

- **Model-Free**: Q-learning does not require a model of the environment; it learns directly from interactions.
- **Off-Policy**: It learns the optimal policy independently of the agent's actions. This means the agent can learn the optimal policy even when not following it, which is useful for experimenting and exploration.

## 2.9 Limitations

- **Scalability**: The $Q$-table grows exponentially with the number of states and actions, making it difficult to use in large or continuous state spaces.
- **Exploration**: Choosing the right exploration strategy and balancing it with exploitation can be challenging.

### 2.10 Conclusion

Q-learning is a foundational algorithm in reinforcement learning that provides a method for agents to learn optimal actions in uncertain environments. It serves as a basis for many more advanced RL algorithms.

## References

[1] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*, 2nd Edition, MIT Press, 2018.