Markov Decision Processes (MDPs)

Reinforcement Learning (RL) is a type of machine learning where an agent learns to make decisions by taking actions in an environment to achieve maximum cumulative reward. The agent interacts with the environment, receives feedback in terms of rewards or penalties, and learns to adjust its behavior to maximize the total reward over time.

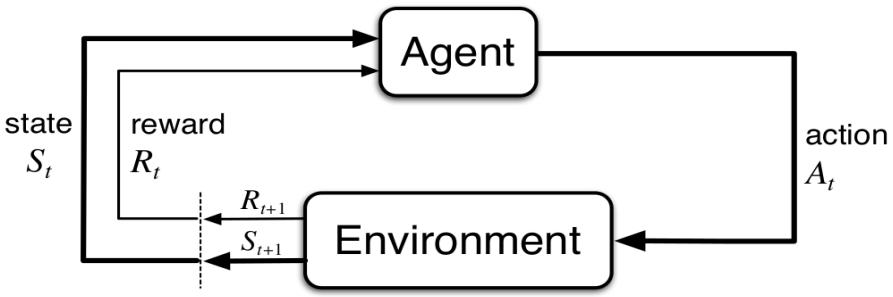

Figure 1: Agent-Environment Interaction

In RL, the environment is typically modeled as a Markov Decision Process (MDP). An MDP provides a mathematical framework to describe an environment in RL. It consists of a set of states S, a set of actions A, a transition model P that describes the probability of transitioning between states given an action, and a reward function R that assigns a scalar reward to each state-action pair. The agent's objective is to find a policy, which is a mapping from states to actions, that maximizes the expected cumulative reward over time, typically with a discount factor applied to future rewards.

The relationship between RL and MDP is fundamental. MDPs provide the formalism for modeling the decision-making problem that RL aims to solve, and RL algorithms are techniques for finding optimal or near-optimal policies in MDPs. The use of MDPs in RL allows for the application of various mathematical tools and algorithms to systematically explore, evaluate, and optimize the decision-making process of agents in various environments.

A *Markov Decision Process* (MDP) is a mathematical framework for modeling decision-making in situations where outcomes are partly random and partly under the control of a decision-maker [1, 2]. An MDP is defined by a tuple (S, A, P, R), where:

- S is a finite set of states,
- A is a finite set of actions,
- P is a state transition probability matrix,
- R is a reward function.

State Transition Probability Matrix (P)

For each state-action pair $(s, a) \in S \times A$, there is a probability distribution over the state space S, denoted by $P(s'|s, a)$. This denotes the probability of transitioning to state s' given that the current state is s and action a is taken.

$$P(s'|s, a) = \Pr(S_{t+1} = s'|S_t = s, A_t = a)$$

Reward Function (R)

The reward function R gives the immediate reward received after transitioning from state s to state s', and is denoted by $R(s, a, s')$ or sometimes simplified to $R(s, a)$ if the reward is only dependent on the current state and action.

$$R(s, a, s') = E[R_{t+1}|S_t = s, A_t = a, S_{t+1} = s']$$

Policies

A policy π defines the behavior of the agent, mapping from states to probabilities of selecting each possible action. If the agent is following policy π at time t, then $\pi(a|s)$ is the probability that $A_t = a$ if $S_t = s$.

$$\pi(a|s) = \Pr(A_t = a|S_t = s)$$

Value Functions

The value function of a state s under a policy π, denoted $V^\pi(s)$, is the expected return when starting in state s and following policy π thereafter.

$$V^\pi(s) = E_\pi[G_t|S_t = s]$$

where G_t is the return at time t, defined as:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

and γ is the discount factor, $0 \leq \gamma \leq 1$.

Bellman Equation

The Bellman equation expresses the relationship between the value of a state and the values of its successor states. For a given policy π, the Bellman equation for the value function V^π is:

$$V^\pi(s) = \sum_{a \in A} \pi(a|s) \left(R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^\pi(s') \right)$$

Optimal Value Function and Optimal Policy

The optimal value function $V^*(s)$ gives the maximum value function overall policies:

$$V^*(s) = \max_\pi V^\pi(s)$$

The optimal policy π^* is the policy that achieves the optimal value function:

$$\pi^* = \arg\max_\pi V^\pi(s)$$

The Bellman optimality equation for V^* is:

$$V^*(s) = \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^*(s') \right)$$

0.1 Assumptions for this problem

The environment here is a 7 gridworld domain. For this project, we assume that the environment is deterministic meaning that, being at a given state, taking an action will lead you to the next state with probability 1. The goal is for the agent to choose the optimal or shortest path to move from the start to the goal position given the Bellman optimality equation for V^*.

- The environment here is a 7 gridworld domain. Assume the third row from the top is all obstacles, except the rightmost cell. The initial state is the bottom left, and the goal is the top left. Our MDP is defined as follows:
  - The states are just the cells the agent is currently in
  - The actions are the cardinal directions and are deterministic
  - There is no discounting. The reward for reaching the goal is 20, and -1 for each action taken by the agent.
- We assume we are given the optimal value function for simplification

$$V^* = \begin{bmatrix} 20 & 19 & 18 & 17 & 16 & 15 & 14 \\ 19 & 18 & 17 & 16 & 15 & 14 & 13 \\ 0 & 0 & 0 & 0 & 0 & 0 & 12 \\ 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix},$$

  where the 0 values represent the obstacle.

We will run both the random and the greedy agents that terminate after 50 steps in the domain to see their behavior. We will plot a bar graph of the returns accumulated by the two agents, averaged over 20 runs each.

References

[1] Martin L Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[2] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press Cambridge, 2018.