Table of Contents

1. Recap
2. Policies
3. Reward and Return
4. Value Functions
5. Bellman Equation
Recap: MDP

At each step $t$ the agent:

- Receives state $S_t$ / observation $O_t$ and reward $R_t$
- Executes action $A_t$

The environment:

- Receives action $A_t$
- Emits state $S_{t+1}$ / observation $O_{t+1}$ and reward $R_{t+1}$
Recap: MDP Definitions

Definition

Markov decision process (MDP) defined by the tuple $\langle s, a, O, P, r, \rho_0, \gamma \rangle$, where

- $s \in S$ denotes states, describing all possible configurations;
- $a \in A$ denotes actions;
- $P : S \times A \times S \rightarrow \mathbb{R}$ is the states transition probability distribution;
- $O$ is a set of observations;
- $r : S \rightarrow \mathbb{R}$ is the reward function;
- $\rho_0 : S \rightarrow [0, 1]$ is the distribution of the initial state $s_0$;
- $\gamma \in [0, 1]$ is a discount factor
Table of Contents

1 Recap

2 Policies

3 Reward and Return

4 Value Functions

5 Bellman Equation
A policy $\pi$ is a distribution over actions given states. It defines the agent’s behaviour. It can be either deterministic or stochastic:

- **Deterministic:** $\pi(s) = a$
- **Stochastic:** $\pi(a|s) = \mathbb{P}_\pi[A = a|S = s]$

- A policy fully defines the behaviour of an agent
- MDP policies depend on the current state (not the history)
Example

A Markov decision process (MDP) example:

- **Starve**
  - Transition probability: \( p = 0.1 \)
  - Reward: \( r = -10 \)

- **Don't Eat**
  - Transition probability: \( p = 0.9 \)
  - Reward: \( r = -1 \)

- **Hungry**
  - Transition probability: \( p = 0.1 \)
  - Reward: \( r = -1 \)

- **Eat**
  - Transition probability: \( p = 0.9 \)
  - Reward: \( r = 1 \)

- **Full**
  - Transition probability: \( p = 0.5 \)
  - Reward: \( r = 1 \)
  - Transition probability: \( p = 0.5 \)
  - Reward: \( r = -1 \)

Transition probabilities and rewards are represented for each state transition.
Notations: Transition Probability

\[ P_{s,s'}^a = P(S_{t+1} = s' | S_t = s, A_t = a) \]

\( P \) is the transition probability. If we start at state \( s \) and take action \( a \) and we end up in state \( s' \) with probability \( P_{ss'}^a \).
Notations: Transition Probability

\[ \mathcal{P}_{s,s'}^a = P(S_{t+1} = s'|S_t = s, A_t = a) \]

\( \mathcal{P} \) is the transition probability. If we start at state \( s \) and take action \( a \) and we end up in state \( s' \) with probability \( \mathcal{P}_{ss'}^a \).

\[ \mathcal{R}_{s,s'}^a = \mathbb{E}[R_{t+1}|S_t = s, S_{t+1} = s', A_t = a] \]

\( \mathcal{R}_{ss'}^a \) is another way of writing the expected (or mean) reward that we receive when starting in state \( s \), taking action \( a \), and moving into state \( s' \).
Example: Recycling Robot

| s     | a      | s'    | $p(s'|s,a)$ | $r(s,a,s')$ |
|-------|--------|-------|-------------|-------------|
| high  | search | high  | $\alpha$   | $r_{\text{search}}$ |
| high  | search | low   | $1-\alpha$ | $r_{\text{search}}$ |
| low   | search | high  | $1-\beta$  | $-3$        |
| low   | search | low   | $\beta$    | $r_{\text{search}}$ |
| high  | wait   | high  | 1           | $r_{\text{wait}}$ |
| high  | wait   | low   | 0           | -           |
| low   | wait   | high  | 0           | -           |
| low   | wait   | low   | 1           | $r_{\text{wait}}$ |
| low   | recharge | high | 1         | 0           |
| low   | recharge | low | 0         | -           |

The diagram shows a transition graph for the recycling robot, with states including 'high', 'low', 'search', and 'wait', and actions such as 'search' and 'recharge'. The transitions are labeled with probabilities and rewards.
Table of Contents

1 Recap
2 Policies
3 Reward and Return
4 Value Functions
5 Bellman Equation
RL agents learn to **maximize cumulative future reward** ($R$).
Cumulative reward:

\[ R_t = r_{t+1} + r_{t+2} + r_{t+3} + r_{t+4} + \cdots = \sum_{k=0}^{\infty} r_{t+k+1} \]
Cumulative reward:

\[ R_t = r_{t+1} + r_{t+2} + r_{t+3} + r_{t+4} + \cdots = \sum_{k=0}^{\infty} r_{t+k+1} \]

Discounted cumulative reward:

\[ R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \]

where \( 0 \leq \gamma \leq 1 \)
Value Functions

There are two types of value functions:

- state value function $V(s)$
There are two types of value functions:

- state value function \( V(s) \)
- action value function \( Q(s, a) \)
**State value function** $V(s)$

**Definition**

*State value function* describes the value of a state when following a policy. It is the expected return when starting from state $s$ acting according to our policy $\pi$:

$$V^\pi(s) = \mathbb{E}_\pi[R_t | S_t = s]$$
\( V^\pi(s) \) can also be interpreted, as the **expected cumulative future discounted reward**, where

- "Expected" refers to the "expected value"
- "Cumulative" refers to the summation
- "Future" refers to the fact that it’s an expected value of a future quantity with respect to the present quantity, i.e. \( s_t = s \).
- "Discounted" refers to the "gamma" factor, which is a way to adjust the importance of how much we value rewards at future time steps, i.e. starting from \( t + 1 \).
- "Reward" refers to the main quantity of interested, i.e. the reward received from the environment.
**Action value function**

**Definition**

*Action value function* tells us the value of taking an action $a$ in state $s$ when following a certain policy $\pi$. It is the expected return given the state and action under $\pi$:

$$Q^\pi(s, a) = \mathbb{E}_\pi[R_t | S_t = s, A_t = a]$$
Table of Contents

1 Recap
2 Policies
3 Reward and Return
4 Value Functions
5 Bellman Equation
Richard Bellman was an American applied mathematician who derived the following equations which allow us to start solving these MDPs. The Bellman equations are ubiquitous in RL and are necessary to understand how RL algorithms work.
Bellman equation for the state value function

\[ V^\pi(s) = \mathbb{E}_\pi[R_t | S_t = s] \]
Bellman equation for the state value function

\[ V^\pi(s) = \mathbb{E}_\pi[R_t | S_t = s] \]
\[ = \mathbb{E}_\pi[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots | S_t = s] \]
Bellman equation for the state value function

\[ V^\pi(s) = \mathbb{E}_\pi[R_t | S_t = s] \]

\[ = \mathbb{E}_\pi[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots | S_t = s] \]

\[ = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s] \]
Bellman equation for the state value function

\[ V^\pi(s) = \mathbb{E}_\pi[R_t | S_t = s] \]
\[ = \mathbb{E}_\pi[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots | S_t = s] \]
\[ = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s] \]
\[ = \mathbb{E}_\pi[r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | S_t = s] \]
Bellman equation for the state value function

The expectation here describes what we expect the return to be if we continue from state $s$ following policy $\pi$. The expectation can be written explicitly by summing over all possible actions and all possible returned states.

$$
\mathbb{E}_\pi[r_{t+1}|s_t = s] = \sum_a \pi(s, a) \sum_{s'} \mathcal{P}_{ss'}^a \mathcal{R}_{ss'}^a
$$

$$
\mathbb{E}_\pi[\gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2}|S_t = s] = \sum_a \pi(s, a) \sum_{s'} \mathcal{P}_{ss'}^a \gamma \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+2}|S_{t+1} = s']
$$
Bellman equation for the state value function

By distributing the expectation between these two parts, we can then manipulate our equation into the form:

\[
V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a \left[ R_{ss'}^a + \gamma E_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_{t+1} = s' \right] \right]
\]
By distributing the expectation between these two parts, we can then manipulate our equation into the form:

\[
V^\pi(s) = \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a \left[ R_{ss'}^a + \gamma \mathbb{E}_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+2|s_{t+1} = s'} \right] \right]
\]

\[
= \sum_a \pi(s, a) \sum_{s'} P_{ss'}^a \left[ R_{ss'}^a + \gamma V^\pi(s') \right]
\]
Bellman equations is that they let us express values of states as values of other states. This means that if we know the value of \( s_{t+1} \), we can very easily calculate the value of \( s_t \).

Bellman equations is a foundation for iterative approaches to solve reinforcement learning task.