

Reinforcement Learning

Jesús Eduardo Hermosilla Díaz

I. MARKOV DECISION PROCESS

A **Markov Process** is a memoryless random process with the Markov property. Formally is a tuple $\langle S, P \rangle$ where:

- S is a (finite) set of states
- P is a **state transition probability** matrix
 $P_{ss'} = \mathbb{P}[S_{t+1} = s' | S_t = s]$

State transition matrix defines transition probabilities from all states s to all successor states s'

$$\mathbf{P} = \begin{matrix} & \begin{matrix} s'_0 & s'_1 & s'_2 \end{matrix} \\ \begin{matrix} s_0 \\ s_1 \\ s_2 \end{matrix} & \left\| \begin{matrix} P_{11} & \dots & P_{1n} \\ \vdots & & \vdots \\ P_{n1} & \dots & P_{nn} \end{matrix} \right\| \end{matrix}$$

A **Markov Reward Process** (MRP) is a Markov chain with values. Formally is a tuple $\langle S, P, R, \gamma \rangle$ where:

- S is a finite set of states
- P is a state transition probability matrix $P_{ss'}$
- R is a reward function $R_s = \mathbb{E}[R_{t+1} | S_t = s]$
- γ is a discount factor $\gamma \in [0, 1]$

The **discount factor** determines the present value of future rewards.

The **State value function** of an MRP is the expected return starting from state s

$$V(s) = \mathbb{E}[G_t | S_t = s]$$

$$V(s) = \mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) | S_t = s]$$

The **Bellman equation** expresses a relationship between the value of a state and the values of its successor states.

$$V(s) = R_s + \gamma \sum_{s' \in S} P_{ss'} V(s')$$

Using matrices:

$$V = R + \gamma PV$$

Solved directly:

$$V = (I - \gamma P)^{-1} R$$

A **Markov decision process** (MDP) is a Markov reward process with decisions. It is an environment in which all states are Markov. Formally is a tuple $\langle S, A, P, R, \gamma \rangle$ where:

- S is a finite set of states
- A is a finite set of actions
- $P_{ss'}^a$ is a state transition probability matrix
- R is a reward function, $R_s^a = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$
- γ is a discount factor $\gamma \in [0, 1]$

A **Policy** is a mapping from states to probabilities of selecting each possible action i.e, is a distribution over actions given states.

$$\pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$$

The **state-value function** of an MDP is the expected return starting from state s , and then following policy π .

$$V_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s]$$

The **action-value function** is the expected return starting from state s , taking action a , and then following policy π .

$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

There is always at least one policy that is better than or equal to all other policies. This is an **optimal policy**. All they share the same state-value function, called the **optimal state-value function**. It specifies the best possible performance in the MDP, which is “solved” when we know it.

$$V_*(s) = \max_\pi V_\pi(s)$$

Optimal policies also share the same **optimal action-value function**:

$$Q_*(s, a) = \max_\pi Q_\pi(s, a)$$

Optimal policy from optimal state-value function:

$$Q_*(s, a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_*(s')$$

Bellman Optimality equation

$$V_*(s) = \max_a R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a V_*(s')$$

$$Q_*(s, a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \max_{a'} Q_*(s', a')$$

II. PLANNING BY DYNAMIC PROGRAMMING

Optimal value function

Optimal policy

Bellman expectation equation

Bellman optimality equation

(iterative) policy equation

Bellman expectation backup $v_1 \rightarrow v_2 \rightarrow v_3 \dots$

Synchronous backup

Greedy policy with respect to V_k

Policy iteration

Diagram

Termination condition for policy evaluation

Value iteration

Prediction

Control

synchronous dynamic programming
asynchronous dynamic programming
in-place value iteration
prioritized sweeping and Bellman error
real-time dynamic programming

REFERENCES

- [1] David Silver. (2015). Lectures on Reinforcement Learning.
- [2] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.