

Reinforcement Learning

Marshal Sinaga - marshal.sinaga@aalto.fi

2024-09-11

This note aims to cover some materials on the reinforcement learning. The primary references are [Reinforcement Learning: An Introduction \(2nd edition\)](#) by Sutton & Barto and ELEC-E8125 by Joni Pajarinen.

1 Overview

- Reinforcement learning (RL) problem:
 - Denote that $\pi : \mathcal{O} \rightarrow \mathcal{A}$ is a policy that maps the observation to an action.
 - Determine a policy:

$$a = \pi(s) \tag{1}$$

- s.t. the expected cumulative return is maximum, i.e.,

$$\pi^* = \arg \max_{\pi} \mathbb{E}[G] \tag{2}$$

$$G = \sum_t r_t \tag{3}$$

- Markov decision process (MDP):
 - We have an environment observable $z = s$, defined by a Markov dynamics defined as:

$$p(s_{t+1} | s_t, a_t) \tag{4}$$

and a reward function

$$r_t = r(s_t, a_t) \tag{5}$$

- The solution is formulated as follows:

$$a_{1,\dots,T}^* = \arg \max_{a_1,\dots,a_T} \sum_{t=1}^T r_t \tag{6}$$

Represented as policy:

$$a = \pi(s) \tag{7}$$

- Connection between RL and MDP: RL is a MDP with unknown Markov dynamics $p(s_{t+1}|s_t, a_t)$, and unknown reward function r_t .
- Partially observable MDP (POMDP):
 - The environment is not directly observable.
 - Following MDP, POMDP is governed by a Markov dynamics $p(s_{t+1}|s_t, a_t)$ and reward function $r_t = r(s_t, a_t)$. In addition, we have an observation model $p(z_{t+1}|s_{t+1}, a_t)$.

2 Solving discrete MDP

- Markov property: future is independent of past conditioned on the present, i.e.,

$$p(s_{t+1}|s_t) = p(s_{t+1}|s_1, \dots, s_t) \quad (8)$$

- Markov process: a random process that generates a state sequences \mathcal{S} , following the Markov property. Markov process is defined as a tuple (\mathcal{S}, T) , where $T : \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$ denotes the state transition function.
- Markov reward process: defined by a tuple $(\mathcal{S}, T, r, \gamma)$:
 - \mathcal{S}, T follows Markov process
 - $r : \mathcal{S} \rightarrow \mathcal{R}$ denotes the reward function
 - $\gamma \in [0, 1]$ denotes the discount factor
 - Accumulate reward in H horizon step (can be infinite):

$$G_t = \sum_{k=0}^H \gamma^k r_{t+k} \quad (9)$$

- State value function:

$$V(s) = \mathbb{E}[G_t | s_t = s] \quad (10)$$

$$= \mathbb{E}[r_t + \gamma V(s_{t+1}) | s_t = s] \quad (11)$$

- MDP: defined by a tuple $(\mathcal{S}, \mathcal{A}, T, R, \gamma)$
 - \mathcal{S}, γ follows Markov reward process
 - \mathcal{A} denotes set of actions
 - $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$
 - $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$ denotes the reward function
 - Goal: Find the policy $\pi(s)$ that maximizes $V(s)$
- Policy:
 - Deterministic: $\pi(s) : \mathcal{S} \rightarrow \mathcal{A}$
 - Stochastic: $\pi(a|s) \rightarrow [0, 1]$, i.e., distribution over actions.

- MDP value function:

$$V_\pi(s) = \mathbb{E}_\pi[G_t | s_t = s] \quad (12)$$

$$= \mathbb{E}_\pi[r_t + \gamma V_\pi(s_{t+1}) | s_t = s] \quad (13)$$

$$= r(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s') V_\pi(s') \quad (14)$$

- Action-value function:

$$Q_\pi(s, a) = \mathbb{E}_\pi[r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}) | s_t = s, a_t = a] \quad (15)$$

$$= r(s, a) + \gamma \sum_{s'} T(s, a, s') Q_\pi(s', \pi(s')) \quad (16)$$

- Optimal value function:

$$V^*(s) = \max_\pi V_\pi(s) \quad (17)$$

$$Q^*(s, a) = \max_\pi Q_\pi(s, a) \quad (18)$$

- Optimal policy:

$$\pi^*(s) = \operatorname{argmax}_a \mathbb{E}_{s'}[r(s, a) + \gamma V^*(s')] \quad (19)$$

$$= \operatorname{argmax}_a (r(s, a) + \gamma \sum_{s'} T(s, a, s') V^*(s')) \quad (20)$$

- Iterative policy evaluation

- Problem: Evaluate the value of policy π
- Solution: Iterate Bellman expectation backs-up:

$$V_1 \rightarrow \dots \rightarrow V_\pi$$

- Apply synchronous back-ups:

- * For all s , update $V_{k+1}(s)$ from $V_k(s')$
- * Repeat

$$V_{k+1}(s) = r(s, \pi(s)) + \gamma \sum_{s'} T(s, \pi(s), s') V_k(s') \quad (21)$$

$$= \sum_a \pi(a|s) (r(s, a) + \gamma \sum_{s'} T(s, a, s') V_k(s')) \quad (22)$$

- Time complexity of value iteration:

- Complexity $\mathcal{O}(|\mathcal{A}||\mathcal{S}|^2)$ per iteration.
- Complexity when applied to action-value function: $\mathcal{O}(|\mathcal{A}|^2|\mathcal{S}|^2)$ per iteration.