Lecture 8: Markov Decision Processes

Lecturer: Steven Wu
Scribe: Kimberly Truong, Megan Li

8.1 Overview

We will start sequential decision making and reinforcement learning:

1. Elements of Markov Decision Processes (MDP) with a focus on finite horizons
2. Value functions: Bellman Equations/optimality
   Note: Bellman is sometimes synonymous with dynamic programming
3. Value/policy iteration

8.2 MDP Notation

Consider the following notation:

- State space S
- Action space A
- Reward function $R : S \times A \to \Delta([0, 1])$, or $r \sim R(s, a)$
- Transition operator $P : S \times A \to \Delta(S)$
- Initial state distribution $\mu_0 \in \Delta(S)$
  with initial state $s_0 \sim \mu_0$

Fact 1 *MDPs have the Markovian ("memorylessness") property where its future state(s) are independent of its history.*

8.3 Finite Horizon

Remark 2 *Discounted and Infinite Horizon problems also exist, but we will focus on problems with finite horizons in this class.*

Consider the following notation:

- Horizon H
- Trajectory $\tau = (s_0, a_0, r_0, s_1, ..., s_{H-1}, a_{H-1}, r_{H-1})$
- Policy π
- Objective

$$J(\pi) = \mathbb{E}\left[\sum_{h=0}^{H-1} r_h | S_0, a_{0:H-1} \sim \pi\right] \tag{8.1}$$

  There can be as many as $|A|^{|S|H}$ deterministic policies.

- Value function(s)
  - State-value function

$$V_h^\pi := \mathbb{E}\left[\sum_{h'=h}^{H-1} r_{h'} | s_h = s, a_{h:H-1} \sim \pi\right] \tag{8.2}$$

  - Action-value function

$$Q_h^\pi := \mathbb{E}\left[\sum_{h'=h}^{H-1} r_{h'} | s_h = s, a_h = a, a_{h+1:H-1} \sim \pi\right] \tag{8.3}$$

8.3.1 (Non-stationary) Markov Policy

In general, a policy $\pi : \mathcal{H} \to \Delta(A)$, where $\mathcal{H}$ represents all partial history.

Definition 1 *A* ***non-stationary Markov Policy*** *compresses all partial history into the current state and time step. π_h denotes a policy defined at time step h.*

$$\pi : S \times [H] \to \Delta(A)$$

8.3.2 Bellman Equations

For any policy π, we can derive the following *Bellman equations*:

$$V_h^\pi = \mathbb{E}[r_h + V_{h+1}^\pi(s_{h+1})|s_h = s, a_h \sim \pi] \tag{8.4}$$

$$Q_h^\pi(s,a) = \mathbb{E}[r_h + Q_{h+1}^\pi(s_{h+1}, a_{h+1})|s_h = s, a_h = a, a_{h+1} \sim \pi] \tag{8.5}$$

Remark 3 *Considering just the first time step $h = 1, \pi = \pi_0$ yields:*

$$J(\pi) = \mathbb{E}_{s_0 \sim \mu_0}[V^\pi(s_0)]$$

Theorem 4 *Define $V^* = (V_0^*, ..., V_H^*)$ recursively as*

$$V_H^*(s) = 0 \tag{8.6}$$

$$\forall (s,h) : V_h^*(s) = \max_a \mathbb{E}_{s' \sim P(s,a)} \left[r(s,a) + V_{h+1}^*(s') \right] \tag{8.7}$$

Then, $\sup_{\pi:\mathcal{H}\to\Delta(A)} J(\pi) = \mathbb{E}_{s_0\sim\mu}[V_0^*(s_0)]$*. Now define the policy $\pi^* := (\pi_0^*, ..., \pi_{H-1}^*)$ as*

$$\forall (s,h) : \pi_h^*(s) = \arg\max_a \{\mathbb{E}_{s'\sim P(s,a)}[r(s,a) + V_{h+1}^*(s')]\} \tag{8.8}$$

Because π^* achieves value V^* for all (s,h), it achieves optimality.

The equation (8.7) is called the Bellman optimality equation (for V). The recursive procedure (going from H to 0) defined by equations (8.6) and (8.7) is called *value iteration*.

Similarly, one can also derive the Bellman optimality equation and value iteration in terms of action value functions:

$$\forall (s,a) \in S \times A \qquad Q_H^*(s,a) = 0 \tag{8.9}$$

$$\forall (s,a,h) \in S \times A \times [H-1] \qquad Q_h^*(s,a) = \mathbb{E}\left[r(s,a) + \max_{a'} Q_{h+1}^*(s',a') \right] \tag{8.10}$$

The optimal policy can also be written as

$$\pi_h^*(s) = \arg\max_a Q_h^*(s,a)$$

8.3.3 Policy Iteration

Another algorithm for computing the optimal value function and policy is *policy iteration*. Start with $\pi^{(0)}$, and then for each increment of t, we compute $Q^{\pi^{(t-1)}}$ where $Q_H^{\pi^{(t-1)}} = 0$ and for all (s,a) pairs, $Q_h^{\pi^{(t-1)}}(s,a)$ is computed by (8.5). This is the *policy evaluation* step. It is followed by the *policy improvement* step, during which we greedily update the policy as

$$\pi_h^{(t)}(s) := \arg\max_a Q_h^{\pi^{(t-1)}}(s,a) \tag{8.11}$$

Remark 5 *This algorithm guarantees* local improvement*—that is, at every time step h and state s:*

$$\forall(s,h) : \mathbb{E}_{a\sim\pi^{(t+1)}(s)}[Q_h^{\pi^{(t)}}(s,a)] = \max_{a\in A} Q_h^{\pi^{(t)}}(s,a) \geq \mathbb{E}_{a\sim\pi^{(t)}}[Q_h^{\pi^{(t)}}(s,a)] \tag{8.12}$$

for all t.

Interestingly, by the seminal *Performance Difference Lemma*, we can ensure *global improvement*, provided that we can local improvement at every state s. In this context, the lemma can be stated as follows.

Lemma 6 (Performance Difference Lemma)

$$J(\pi^{(t+1)}) - J(\pi^{(t)}) = \mathbb{E}_{\tau\sim\pi^{(t+1)}}\left[\sum_{h=1}^{H-1} \mathbb{E}[Q^{\pi^{(t)}}(s_h, \pi_h^{(t+1)}(s_h))] - \mathbb{E}[Q^{\pi^{(t)}}(s_h, \pi_h^{(t)}(s_h))]\right] \tag{8.13}$$

The left-hand side of the equation is the "global improvement" in the total reward objective and the right-hand side is the sum over the expected local improvement over time steps.