Class 6

Shikhar Saxena

January 24, 2023

Contents

Stochastic Optimization: 1-period MDP

Problem P1:

$$\min_{\pi: S \to \mathcal{A}} E[c(S, \pi(S), W)] \tag{1}$$

Here we are optimizing over all mappings (Functional Optimization Problem).

Define $Q(s, a) := E[c(s, a, W)|S = s]$.

Problem P2:

$$\min_{a \in \mathcal{A}} Q(s, a) \text{ and } \pi^*(s) = arg \min_{a \in \mathcal{A}} Q(s, a) \tag{2}$$

Here we are optimizing over actions (over a chosen state) (Collection of Parameter Optimization).

The solution in (2) is definitely a solution for (1) but the converse might not be true.

Proof. **Proof for converse might not be true:**

An intuition for this is that P2 might have a solution for a state (that is improbable $P(s) = 0$). If we solve through P1, it might map to any action (because this state is improbable) but through P2, we might get a fixed mapping for this state.

Proof for other side:

Let π^* minimize P2.

$$\begin{aligned} E[c(S,\pi(S),W)] &= E[E[c(S,\pi(S),W)|S]] \\ &\geq E[E[c(S,\pi^*(S),W)|S]] \\ &= E[c(S,\pi^*(S),W)] \end{aligned}$$

□

Dynamic Programming

Deterministic Dynamic Programming

The environment is governed by what is called a *deterministic plant equation*. Essentially, deterministic transitions and root.

Shortest Path from Root Node (R) to Leaf Node example

- State space $\mathcal{S}$
  - ⋆ all nodes
- Action space $\mathcal{A}$
  - ⋆ {leftnode, rightnode}
- $f : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ (Plant equation)
  - ⋆ Denotes next state. $s_{t+1} = f(s_t, a_t)$
- Edge cost incurred $c(s, a)$
- Objective: $V(R)$

$$V(R) = \min_{a \in \mathcal{A}} \{c(R, a) + V(s')\} \text{ where } s' = f(R, a)$$

These are called **Bellman-Type Equation**: Solve recursively backwards.

Notation

- Discrete set of times $t = 0, 1, \ldots, T$
- Notation from the shortest path example
- Countable spaces in most cases (unless specified)
- s_0 denote starting state
- $r(s_t, a_t)$ or $c(s, a)$
- $r_T(s_T)$: Reward for terminating in s_T at time T
- $\pi = (\pi_t : 0, 1, \ldots, T-1)$
  - ⋆ Policy Vector over all timesteps (No action at the last timestep).
  - ⋆ Specifies action $\pi_t \in \mathcal{A}$ to be taken at time t.

    - ⋆ π is a function of state (can also depend on the timestep tho; like we have used in this notation).
    - ⋆ $s_{t+1} = f(s_t, \pi_t)$
- Cumulative Reward:

$$V^\pi(s_0) = r(s_0, \pi_0) + r(s_1, \pi_1) + \cdots + r_T(s_T)$$

Definition of a Dynamic Program

$$V(s_0) \quad := \max_{\pi \in \Pi} V^\pi(s_0) \quad := \sum_{t=0}^{T-1} r(s_t, \pi_t) + r_T(s_T)$$

- Except for easy problems, it is difficult to get a closed form solution for this

Let $\pi_t := (\pi_t, \dots, \pi_{T-1})$

Define $V_t^{\pi_t}(s_t) = \sum_{u=t}^{T-1} r(s_u, \pi_u) + r_T(s_T)$

Then, $V_T^{\pi_T}(s) = r_T(s)$ and $V_0^\pi(s) = V(s_0)$

Bellman Optimality Equation

$$V_t(s) = \max_{a \in \mathcal{A}} \{r(s,a) + V_{t+1}(s')\} \text{ where } s' = f(s,a)$$