CS 7.603
Reinforcement Learning

Tejas Bodas

Assistant Professor, IIIT Hyderabad

Sub-problems of main problem

- Recall $V^{\pi}(s_0) := \sum_{t=0}^{T-1} r(s_t, \pi_t) + r_T(s_T)$
- $V(s_0) := \max_{\pi} V^{\pi}(s_0)$ and $\pi^* := \underset{\pi}{argmax}\, V^{\pi}(s_0)$
- We now consider notations for sub-problems starting at time t.
- Let $\boldsymbol{\pi}_t := (\pi_t, \ldots, \pi_{T-1})$.
- Define $V_t^{\boldsymbol{\pi}_t}(s_t) := \sum_{u=t}^{T-1} r(s_u, \pi_u) + r_T(s_T)$;
- $V_T^{\boldsymbol{\pi}_T}(s) = r_T(s)$ and $V_0^{\pi}(s) = V^{\pi}(s)$.

Policy Evaluation: $V_t^{\boldsymbol{\pi}_t}(s_t) = r(s_t, \pi_t) + V_{t+1}^{\boldsymbol{\pi}_{t+1}}(s')$

- Now define $V_t(s_0) := \max_{\boldsymbol{\pi}_t} V_t^{\boldsymbol{\pi}_t}(s_0)$
- Bellman equations relate V_t with V_{t+1} which we use recursively to obtain $V_0 = V$.

Bellman Optimality Equation

Theorem

For $t = 0, \ldots, T-1$ and all $s \in \mathcal{S}$, the following is true:

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

where $s \in \mathcal{S}$ and $s' = f(s,a)$.

Proof:

- $V_t^{\pi_t}(s) = r(s, \pi_t) + V_{t+1}^{\pi_{t+1}}(s')$ and $V_t(s) := \max_{\boldsymbol{\pi}_t} V_t^{\pi_t}(s)$

$$\begin{aligned}
V_t(s) &= \max_{\boldsymbol{\pi}_t} \ \{r(s, \pi_t) + V_{t+1}^{\boldsymbol{\pi}_{t+1}}(s')\} \\
&= \max_{a} \max_{\{\boldsymbol{\pi}_t : \pi_t = a\}} \ \{r(s,a) + V_{t+1}^{\boldsymbol{\pi}_{t+1}}(s')\} \\
&= \max_{a} \ \{r(s,a) + \max_{\{\boldsymbol{\pi}_{t+1}\}} V_{t+1}^{\boldsymbol{\pi}_{t+1}}(s')\} \\
&= \max_{a \in \mathcal{A}} \{r(s,a) + V_{t+1}(s')\} \quad \square
\end{aligned}$$

Principle of Optimality

Theorem

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

- How do we get the optimal policy from this ?
- When in state s, we have $\pi_t^* = \underset{a \in \mathcal{A}}{argmax}\{r(s, a) + V_{t+1}(s')\}$

Principle of optimality: Subsolutions of an optimal solution of the problem are themesleves optimal solutions for their subproblems

Remarks

- We assumed that the plant equation f and rewards r are stationary. We can also allow for it to depend on time.
- We can have $s_{t+1} = f_t(s_t, a_t)$ and $r_t(s_t, a_t)$.
- We assumed the action space to be $\mathcal{A}$ for all states and time. The feasible actions could instead depend on current state and time.
- In this case, we denote the action space by $\mathcal{A}_{s_t}$ when in state s_t at time t.
- With any of the above cases, the theorems hold as is (with appropriate change in notation).
- We now provide the DP algorithm with these generalizations.

The DP algorithm

Start with

$$V_T(s_T) = r_T(s_T) \quad \text{for all possible } s_T \tag{1}$$

and for $t = T - 1 \ldots 0$, set

$$V_t(s_t) = \max_{a \in \mathcal{A}_{s_t}} \{r_t(s_t, a) + V_{t+1}(f_t(s_t, a)\} \quad \text{for all } s_t. \tag{2}$$

Now construct optimal policy $\pi^* = (\pi_0^*, \ldots, \pi_{T-1}^*)$ as follows by sequentially going forward for $t = 0, \ldots, T - 1$ to set

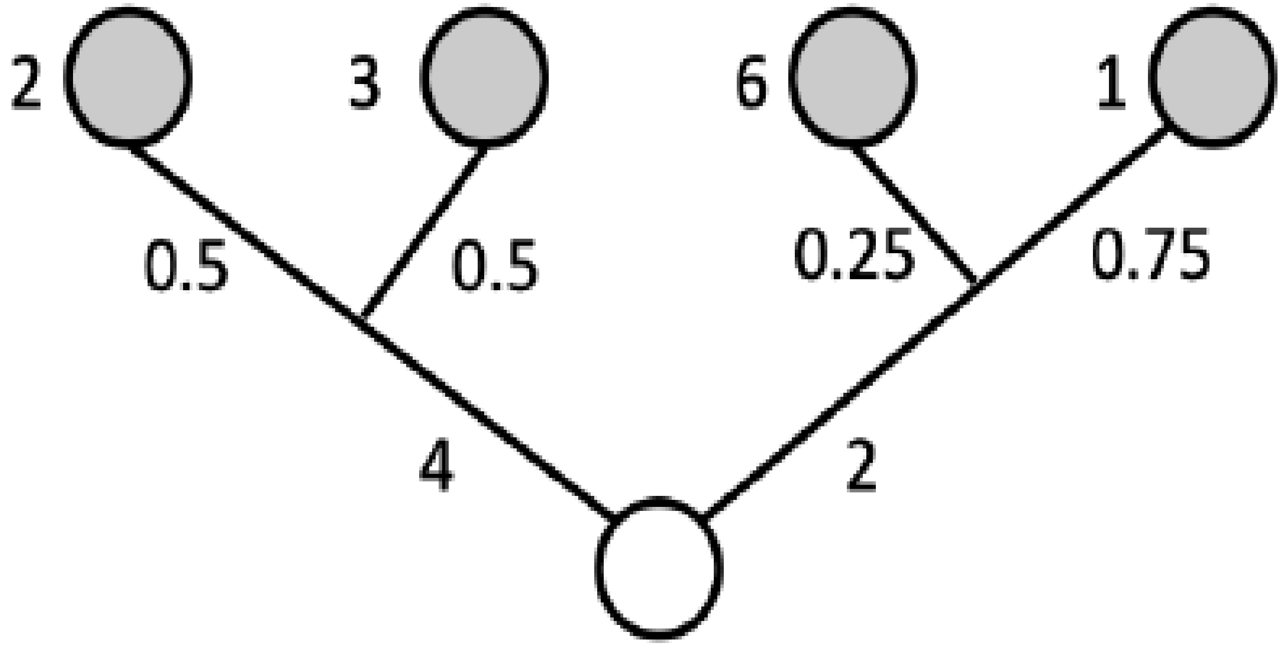
$$\pi_t^* = \underset{a \in \mathcal{A}_{s_t}}{argmax} \{r_t(s, a) + V_{t+1}(f_t(s_t, a)\} \tag{3}$$

Examples: Self-Study

- Shortest path problems (Bellman-Ford Algorithm)
- Viterbi decoding algorithms for convolutional codes `https://www2.isye.gatech.edu/~yxie77/ece587/viterbi_algorithm.pdf`
- Knapsack problem `https://www.es.ele.tue.nl/education/5MC10/Solutions/knapsack.pdf`
- LQ regularization problem (Bertsekas, RL and Optimal Control book)
- Forward Dynamic programming (Bertsekas)

Stochastic Dynamic Programming

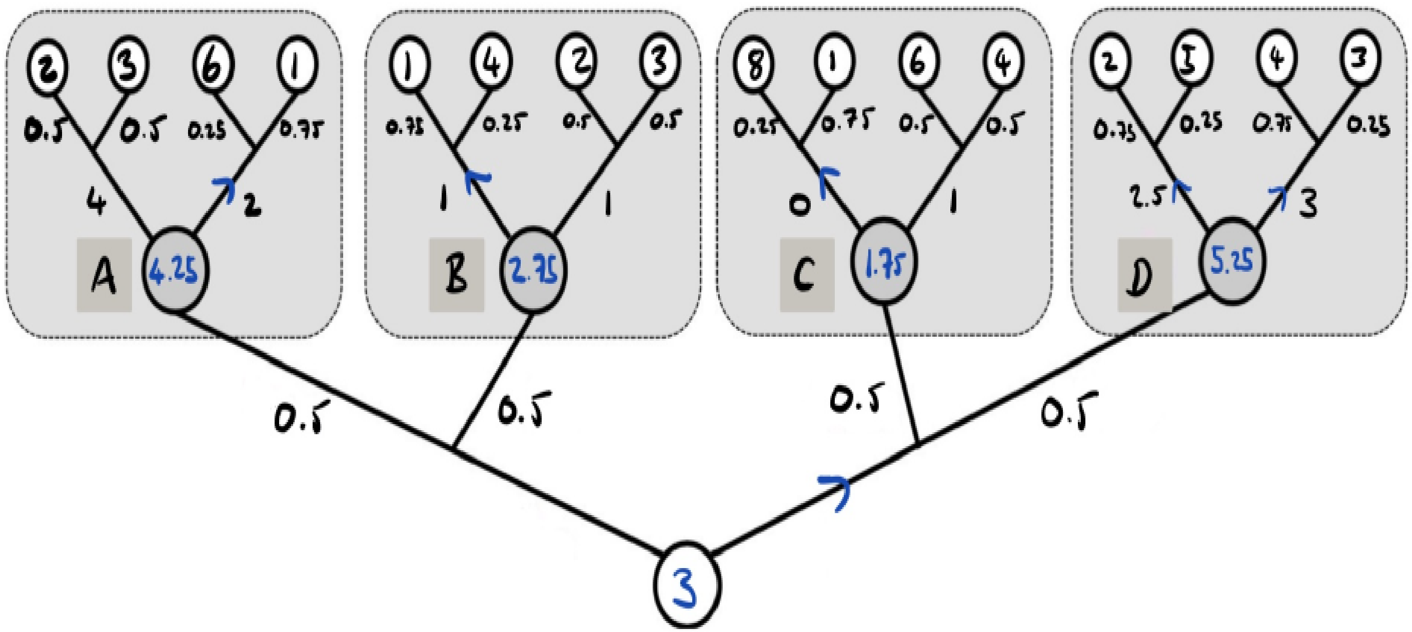
A stochastic shortest path example



- We want to find the least costly path from R to any leaf. [1]
- Leaf nodes additionally have terminal costs.
- The leaf node that you reach after taking an action is a random variable.
- Hence we calculate expected costs.

[1] Example from Neil Walton's notes on Stochastic Control

An Example



- As earlier, the original problem can be broken down into sequence of simpler problems (shaded boxes)
- Key difference is the need to take expectations.

MDP's: State, Action, Reward, State

- Lets consider discrete set of times $t = 0, 1, \ldots, T$
- Let $\mathcal{S}$ denote the state space and $\mathcal{A}$ denote the action space.
- Unless specified, we assume that $\mathcal{S}$ and $\mathcal{A}$ are countable sets.
- $S_t \in \mathcal{S}$ denotes the random state of the system at time t.
- Let A_t denote the action (possibly random) at time t
- $S_{t+1} = f_t(S_t, A_t, W_t)$ is the dynamics where W_t is i.i.d noise.
- As seen earlier, this implies state transitions are Markovian with transition probabilities $\mathbb{P}(S_{t+1} = s' | S_t = s, A_t = a) = P(s'|s, a)$
- $r(S_t, A_t)$ is the random reward from action A_t in state S_t.
- $r_T(S_T)$ denotes the reward for terminating in S_T at time T.

Types of policies

- A policy $\pi = (\pi_t : t = 0, 1, \ldots, T-1)$ specifies action $\pi_t \in \mathcal{A}$ to be taken at time t.
- If $\pi_t : t \to \mathcal{A}$, its a state independent, deterministic policy.
- If $\pi_t : t \to \mathcal{P}(\mathcal{A})$, its a state independent, randomized policy. $\mathcal{P}(\mathcal{A})$ denotes the set of probability distributions over $\mathcal{A}$.
- Let $\mathcal{H}_t = (S_{1:t}, A_{1:t-1})$. Then $\pi_t : \mathcal{H}_t \to \mathcal{P}(\mathcal{A})$, its a history dependent, randomized policy.
- If $\pi_t : (s_t, t) \to \mathcal{A}$, its a Markovian, non-stationary and deterministic policy.
- If $\pi_t : s \to \mathcal{A}$, its a Markovian, stationary and deterministic policy.
- We let Π^K denote class of policies with property K where $K \in \{HR, HD, MR, MD, DS\}$. Π denotes space of all policies.

MDP: Cumulative reward, value function

- How good is any policy $\pi = (\pi_1, \ldots, \pi_{T-1})$? Measured using expected cumulative reward (ak.a value of a policy)
- $V^\pi(s_0) := \mathbb{E}_{s_0}[r(s_0, \pi_0) + r(S_1, \pi_1) + \ldots + r_T(S_T)]$.
- $\mathbb{E}_{s_0}$ denotes expectation conditioned on starting in s_0. Sometimes, the notation $\mathbb{E}^\pi_{s_0}$ is used to denote dependence on π. We will however supress this notation throughout.
- How do we get the best policy π^*?

Problem P3: $V(s_0) := \sup_{\pi \in \Pi} V^\pi(s_0)$

- Note: the optimal policy depends on the starting state.

Optimality of Deterministic Markovian policies

Theorem

The cost incurred by the best Markovian strategy, is same as the cost incurred by the best history dependent strategy, i.e.,
$V(s_0) = \sup_{\pi \in \Pi^{HR}} V^{\pi}(s_0) = \sup_{\pi \in \Pi^{MD}} V^{\pi}(s_0)$

- Proof is outside scope of the course and uses Balckwell's result.
- See Putterman Chapter 4, Thm 4.4.2 or Aditya Mahajan notes.
- Note that the policy need not be stationary.
- We will only focus on deterministic Markovian policies henceforth.
- HW: Why supremum and not maximum ? When can you replace supremum by maximum?