Lecture 3: Markov Decision Processes

Bolei Zhou

The Chinese University of Hong Kong

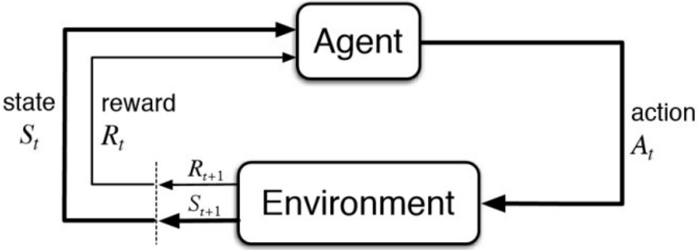
bzhou@ie.cuhk.edu.hk

January 14, 2020

This week's Plan

1. Last Time
   1. Key elements of an RL agent: model, value, policy
2. This Week: Decision Making in MDP
   1. Markov decision processes (MDP)
   2. Policy evaluation in MDP
   3. Control in MDP: policy iteration and value iteration

Markov Decision Process (MDP)



1. Markov Decision Process can model a lot of real-world problem. It formally describes the framework of reinforcement learning
2. Under MDP, the environment is fully observable.
   1. Optimal control primarily deals with continuous MDPs
   2. Partially observable problems can be converted into MDPs

Define the model of the environment

- Markov Processes
- Markov Reward Processes(MRPs)
- Markov Decision Processes (MDPs)

Markov Property

1. The history of states: $h_t = \{s_1, s_2, s_3, ..., s_t\}$
2. State s_t is Markov if and only if:

$$p(s_{t+1}|s_t) = p(s_{t+1}|h_t) \quad (1)$$
$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t) \quad (2)$$

3. "The future is independent of the past given the present"

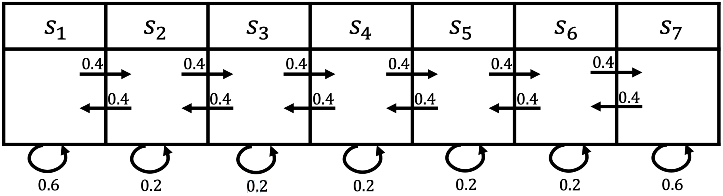
Markov Process/Markov Chain

1. P is the dynamics/transition model that specifies $p(s_{t+1} = s'|s_t = s)$
2. State transition matrix: (From|To)

$$P = \begin{bmatrix} P(s_1|s_1) & P(s_2|s_1) & \ldots & P(s_N|s_1) \\ P(s_1|s_2) & P(s_2|s_2) & \ldots & P(s_N|s_2) \\ \vdots & \vdots & \ddots & \vdots \\ P(s_1|s_N) & P(s_2|s_N) & \ldots & P(s_N|s_N) \end{bmatrix}$$

3. Note that there are no rewards or no actions.

Example of MP

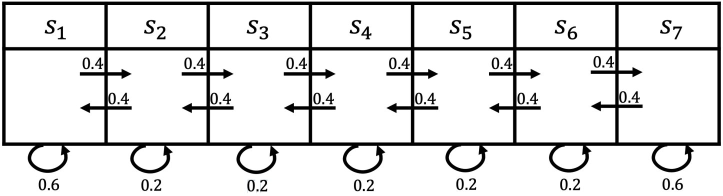


1. Samples episodes starting from s_3
   1. s_3, s_4, s_5, s_6, s_6
   2. s_3, s_2, s_3, s_2, s_1
   3. s_3, s_4, s_4, s_5, s_5

Markov Reward Process (MRP)

1. Markov Reward Process is a Markov Chain + reward
2. Definition of Markov Reward Process (MRP)
   1. S is a (finite) set of states ($s \in S$)
   2. P is dynamics/transition model that specifies $P(S_{t+1} = s' | s_t = s)$
   3. R is a reward function $R(s_t = s) = \mathbb{E}[r_t | s_t = s]$
   4. Discount factor $\gamma \in [0, 1]$
3. If finite number of states, R can be a vector

Example of MRP



Reward: $+5$ in s_1, $+10$ in s_7, 0 in all other states. So that we can represent $R = [5, 0, 0, 0, 0, 0, 10]$

Return function and Value function

1. Definition of Horizon
   1. Number of maximum time steps in each episode
   2. Can be infinite, otherwise called finite Markov (reward) Process
2. Definition of Return
   1. Discounted sum of rewards from time step t to horizon

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + ... + \gamma^{T-t-1} R_T$$

3. Definition of state value function $V_t(s)$ for a MRP
   1. Expected return from t in state s

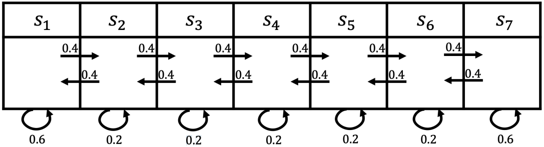
$$\begin{aligned} V_t(s) =& \mathbb{E}[G_t | s_t = s] \\ =& \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... + \gamma^{T-t-1} R_T | s_t = s] \end{aligned}$$

   2. Present value of future rewards

Why Discount Factor γ

1. Mathematically convenient to discount rewards
2. Avoids infinite returns in cyclic Markov processes
3. Uncertainty about the future may not be fully represented
4. If the reward is financial, immediate rewards may earn more interest than delayed rewards
5. Animal/human behaviour shows preference for immediate reward
6. It is sometimes possible to use undiscounted Markov reward processes (i.e. $\gamma = 1$), e.g. if all sequences terminate.
    1. $\gamma = 0$: Only care about the immediate reward
    2. $\gamma = 1$: Future reward is equal to the immediate reward.

Example of MRP



1. Reward: $+5$ in s_1, $+10$ in s_7, 0 in all other states. So that we can represent $R = [5, 0, 0, 0, 0, 0, 10]$
2. Sample returns for a 4-step episodes with $\gamma = 1/2$
   1. return for s_4, s_5, s_6, s_7 : $0 + \frac{1}{2} \times 0 + \frac{1}{4} \times 0 + \frac{1}{8} \times 10 = 1.25$
   2. return for s_4, s_3, s_2, s_1 : $0 + \frac{1}{2} \times 0 + \frac{1}{4} \times 0 + \frac{1}{8} \times 5 = 0.625$
   3. return $s_4, s_5, s_6, s6$: $= 0$
   4. How to compute the value function?

Computing the Value of a Markov Reward Process

1. Value function: expected return from starting in state s

$$V(s) = \mathbb{E}[G_t|s_t = s] = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ...|s_t = s]$$

2. MRP value function satisfies the following Bellman equation:

$$V(s) = \underbrace{R(s)}_{\text{Immediate reward}} + \underbrace{\gamma \sum_{s' \in S} P(s'|s)V(s')}_{\text{Discounted sum of future reward}}$$

3. Practice: To derive the Bellman equation for V(s)

Matrix Form of Bellman Equation for MRP

Therefore, we can express V(s) using the matrix form:

$$\begin{bmatrix} V(s_1) \\ V(s_2) \\ \vdots \\ V(s_N) \end{bmatrix} = \begin{bmatrix} R(s_1) \\ R(s_2) \\ \vdots \\ R(s_N) \end{bmatrix} + \gamma \begin{bmatrix} P(s_1|s_1) & P(s_2|s_1) & \dots & P(s_N|s_1) \\ P(s_1|s_2) & P(s_2|s_2) & \dots & P(s_N|s_2) \\ \vdots & \vdots & \ddots & \vdots \\ P(s_1|s_N) & P(s_2|s_N) & \dots & P(s_N|s_N) \end{bmatrix} \begin{bmatrix} V(s_1) \\ V(s_2) \\ \vdots \\ V(s_N) \end{bmatrix}$$

$V = R + \gamma P V$

1. Analytic solution for value of MRP: $V = (I - \gamma P)^{-1} R$
   1. Matrix inverse takes the complexity $O(N^3)$ for N states
   2. Only possible for a small MRPs

Iterative Algorithm for Computing Value of a MRP

1. Iterative methods for large MRPs:
    1. Dynamic Programming
    2. Monte-Carlo evaluation
    3. Temporal-Difference learning

Iterative Algorithm for Computing Value of a MRP

Algorithm 1 Monte Carlo simulation to calculate MRP value function

1: $i \leftarrow 0, G_t \leftarrow 0$
2: **while** $i \neq N$ **do**
3: generate an episode, starting from state s and time t
4: Using the generated episode, calculate return $g = \sum_{i=t}^{H-1} \gamma^{i-t} r_i$
5: $G_t \leftarrow G_t + g, i \leftarrow i + 1$
6: **end while**
7: $V_t(s) \leftarrow G_t/N$

Iterative Algorithm for Computing Value of a MRP

Algorithm 2 Iterative algorithm to calculate MRP value function

1: for all states $s \in S, V'(s) \leftarrow 0, V(s) \leftarrow \infty$
2: **while** $||V - V'|| > \epsilon$ **do**
3: $\quad V \leftarrow V'$
4: $\quad$ For all states $s \in S, V'(s) = R(s) + \gamma \sum_{s' \in S} P(s'|s)V(s')$
5: **end while**
6: return $V'(s)$ for all $s \in S$

Markov Decision Process (MDP)

1. Markov Decision Process is Markov Reward Process with decisions.
2. Definition of MDP
    1. S is a finite set of states
    2. A is a finite set of actions
    3. P^a is dynamics/transition model for each action $P(s_{t+1} = s' | s_t = s, a_t = a)$
    4. R is a reward function $R(s_t = s, a_t = a) = \mathbb{E}[r_t | s_t = s, a_t = a]$
    5. Discount factor $\gamma \in [0, 1]$
3. MDP is a tuple: (S, A, P, R, γ)

Policy in MDP

1. Policy specifies what action to take in each state
2. Give a state, specify a distribution over actions
3. Policy: $\pi(a|s) = P(a_t = a|s_t = s)$
4. Policies are stationary (time-independent), $A_t \sim \pi(a|s)$ for any $t > 0$

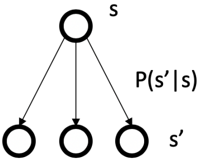
Policy in MDP

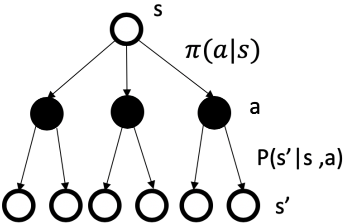
1. Given an MDP (S, A, P, R, γ) and a policy π
2. The state sequence $S_1, S_2, ...$ is a Markov process (S, P^{π})
3. The state and reward sequence $S_1, R_2, S_2, R_2, ...$ is a Markov reward process $(S, P^{\pi}, R^{\pi}, \gamma)$ where,

$$P^{\pi}(s'|s) = \sum_{a \in A} \pi(a|s) P(s'|s, a)$$

$$R^{\pi}(s) = \sum_{a \in A} \pi(a|s) R(s, a)$$

Comparison of MP/MRP and MDP





Value function for MDP

1. The state-value function $v_\pi(s)$ of an MDP is the expected return starting from state s, and following policy π

$$v_\pi(s) = \mathbb{E}_\pi[G_t | s_t = s] \tag{3}$$

2. The action-value function $q_\pi(s, a)$ is the expected return starting from state s, taking action a, and then following policy π

$$Q_\pi(s, a) = \mathbb{E}_\pi[G_t | s_t = s, A_t = a] \tag{4}$$

3. We have the relation between $v_\pi(s)$ and $Q_\pi(s, a)$

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) Q_\pi(s, a) \tag{5}$$

Bellman Expectation Equation

1. The state-value function can be decomposed into immediate reward plus discounted value of the successor state,

$$V_\pi(s) = E_\pi[R_{t+1} + \gamma V_\pi(s_{t+1})|s_t = s] \tag{6}$$

2. The action-value function can similarly be decomposed

$$Q_\pi(s, a) = E_\pi[R_{t+1} + \gamma Q_\pi(s_{t+1}, A_{t+1})|s_t = s, A_t = a] \tag{7}$$

Bellman Expectation Equation for V^π and Q^π

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) q_\pi(s,a) \tag{8}$$

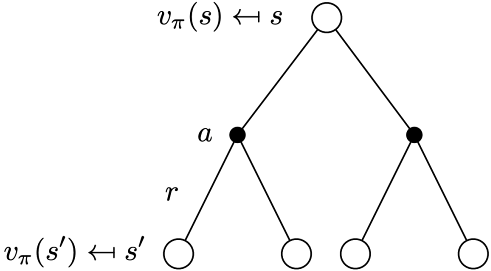
$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P(s'|s,a) v_\pi(s') \tag{9}$$

Thus

$$v_\pi(s) = \sum_{a \in A} \pi(a|s)(R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) v_\pi(s')) \tag{10}$$

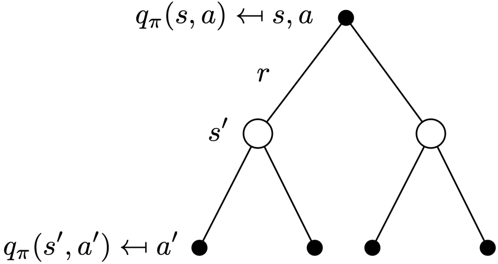
$$q_\pi(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \sum_{a' \in A} \pi(a'|s') q_\pi(s',a') \tag{11}$$

Backup Diagram for V^π



$$v_\pi(s) = \sum_{a \in A} \pi(a|s)(R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) v_\pi(s')) \quad (12)$$

Backup Diagram for Q^π

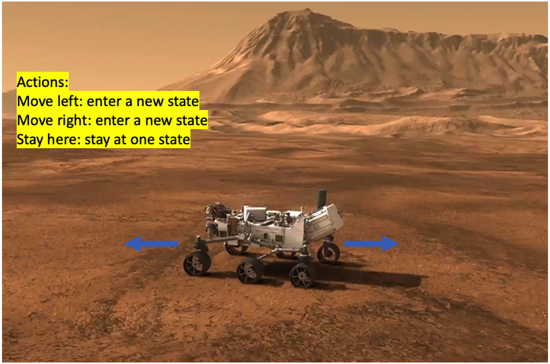


$$q_\pi(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \sum_{a' \in A} \pi(a'|s') q_\pi(s',a') \qquad (13)$$

Policy Evaluation

1. Evaluate the value of state given a policy: compute $v_\pi(s)$
2. Also called as prediction

Example: Mars Rover



Example: Policy Evaluation

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

1. Two actions: *Left* and *Right*
2. For all actions, reward: $+5$ in s_1, $+10$ in s_7, 0 in all other states. So that we can represent $R = [5, 0, 0, 0, 0, 0, 10]$
3. Let's have a deterministic policy $\pi(s) = Left$ and $\gamma = 0$ for any state s, then what is the value of the policy?
4. $V^\pi = [5, 0, 0, 0, 0, 0, 10]$
5. Iterative: $v_k^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} P(s'|s, \pi(s)) v_{k-1}^\pi(s')$

Example: Policy Evaluation

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

1. $R = [5, 0, 0, 0, 0, 0, 10]$
2. Practice 1: Deterministic policy $\pi(s) = \mathit{Left}$ with $\gamma = 0.5$ for any state s, then what are the state values under the policy?
3. Practice 2: Stochastic policy $P(\pi(s) = \mathit{Left}) = 0.5$ and $P(\pi(s) = \mathit{Right}) = 0.5$ and $\gamma = 0.5$ for any state s, then what are the state values under the policy?
4. Following the iteration: $v_k^\pi(s) = r(s, \pi(s)) + \gamma \sum_{s' \in S} P(s'|s, \pi(s)) v_{k-1}^\pi(s')$

Decision Making in Markov Decision Process

1. Prediction (evaluate a given policy):
    1. Input: MDP $<\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma>$ and policy π or MRP $<\mathcal{S}, \mathcal{P}^{\pi}, \mathcal{R}^{\pi}, \gamma>$
    2. Output: value function v_{π}
2. Control (search the optimal policy):
    1. Input: MDP $<\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma>$
    2. Output: optimal value function v_* and optimal policy π_*
3. Prediction and control can be solved by dynamic programming.

Dynamic Programming

Dynamic Programming is a very general solution method for problems which have two properties:

1. Optimal substructure
   1. Principle of optimality applies
   2. Optimal solution can be decomposed into subproblems
2. Overlapping subproblems
   1. Subproblems recur many times
   2. Solutions can be cached and reused

Markov decision processes satisfy both properties

1. Bellman equation gives recursive decomposition
2. Value function stores and reuses solutions

Policy evaluation on MDP

1. Problem: Evaluate a given policy π for a MDP
2. Output the value function under policy v_π
3. Solution: iteration on Bellman expectation backup
4. Synchronous backup algorithm:
    1. At each iteration k+1
       update $v_{k+1}(s)$ from $v_k(s')$ for all states $s \in \mathcal{S}$ where s' is a successor state of s

$$v_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s)(R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) v_k(s')) \qquad (14)$$

5. Convergence: $v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_\pi$

Policy evaluation: Iteration on Bellman expectation backup

Bellman expectation backup for a particular policy

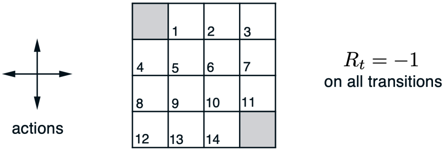
$$v_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s)(R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) v_k(s')) \tag{15}$$

Or if in the form of MRP $< \mathcal{S}, \mathcal{P}^{\pi}, \mathcal{R}, \gamma >$

$$v_{k+1}(s) = R^{\pi}(s) + \gamma P^{\pi}(s'|s) v_k(s') \tag{16}$$

Evaluating a Random Policy in the Small Gridworld

Example 4.1 in the Sutton RL textbook.



1. Undiscounted episodic MDP ($\gamma = 1$)
2. Nonterminal states 1, ..., 14
3. Two terminal states (two shaded squares)
4. Action leading out of grid leaves state unchanged, $P(7|7, right) = 1$
5. Reward is -1 until the terminal state is reach
6. Transition is deterministic given the action, e.g., $P(6|5, right) = 1$
7. Uniform random policy $\pi(l|.) = \pi(r|.) = \pi(u|.) = \pi(d|.) = 0.25$

Evaluating a Random Policy in the Small Gridworld

1. Iteratively evaluate the random policy

v_k for the Random Policy

$k = 0$

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

$k = 1$

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

$k = 2$

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

$k = 3$

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

$k = 10$

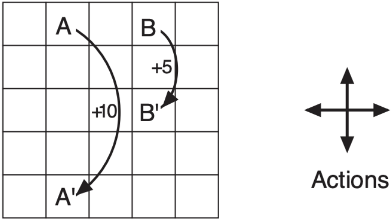
0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0

$k = \infty$

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Practice: Gridworld

Textbook Example 3.5:GridWorld



3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

Optimal Value Function

1. The optimal state-value function $v_*(s)$ is the maximum value function over all policies

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

2. The optimal policy

$$\pi_*(s) = \arg\max_\pi v_\pi(s)$$

3. An MDP is "solved" when we know the optimal value
4. There exists a unique optimal value function, but could be multiple optimal policies (two actions that have the same optimal value function)

Finding Optimal Policy

1. An optimal policy can be found by maximizing over $q_*(s, a)$,

$$\pi_*(a|s) = \begin{cases} 1, & \text{if } a = \arg\max_{a \in A} q_*(s, a) \\ 0, & \text{otherwise} \end{cases}$$

2. There is always a deterministic optimal policy for any MDP
3. If we know $q_*(s, a)$, we immediately have the optimal policy