Lecture 2: Markov Decision Processes

Friday, October 29, 2021

Reinforcement Learning, Winter Term 2021/22

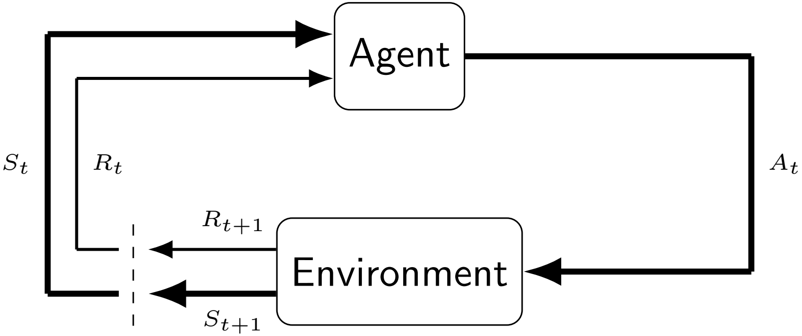
Joschka Boedecker, Gabriel Kalweit, Jasper Hoffmann

Neurorobotics Lab
University of Freiburg

Lecture Overview

Lecture Overview

Agent and Environment



Time steps t: $0, 1, 2, \ldots$
States: $S_0, S_1, S_2, \ldots$
Actions: $A_0, A_1, A_2, \ldots$
Rewards: $R_1, R_2, R_3, \ldots$

Markov Decision Processes

A finite Markov Decision Process (MDP) is a 4-tuple $\langle \mathcal{S}, \mathcal{A}, p, \mathcal{R} \rangle$, where

- $\mathcal{S}$ is a finite number of states,
- $\mathcal{A}$ is a finite number of actions,
- p is the transition probability function $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$,
- and $\mathcal{R}$ is a finite set of scalar rewards. We can then define expected reward $r(s, a) = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$ and $r(s, a, s') = \mathbb{E}[R_{t+1} | S_t = s, A_t = a, S_{t+1} = s']$.

Markov Property

A state-reward pair (S_{t+1}, R_{t+1}) has the Markov property iff:

$$\Pr\{S_{t+1}, R_{t+1} | S_t, A_t\} = \Pr\{S_{t+1}, R_{t+1} | S_t, A_t, \dots, S_0, A_0\}.$$

The future is independent of the past given the present.

Markov Property: Example

Question (3 min)

Imagine you want to apply the algorithms from this lecture on a real physical system. You get sensor input after each 0.01 seconds, but the execution of actions has a delay of 0.2 seconds.

Which adjustments to the state and/or action space would you have to do to fullfill the Markov Property?

Markov Property: Example

Solution (3 min)

Consider the history of the last 0.2 seconds as part of the state space.
However, this blows up the state space which makes computations more challenging (*curse of dimensionality*).

Rewards

- A reward R_t in time step t is a **scalar** feedback signal.
- R_t indicates how well an agent is performing **at single time step** t.

Reward Hypothesis

All of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).

Examples:

- Chess: $+1$ for winning, -1 for losing
- Walking: $+1$ for every time step not falling over
- Investment Portfolio: difference in value between two time steps

Return

- The agent aims at maximizing the expected **cumulative reward**
- Non-discounted: $G_t = R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$
- Discounted: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$
- Discounting with $\gamma \in [0, 1]$ to prevent from infinite returns (e.g. in infinite horizon control problems)
- Returns at successive time steps are related to each other:

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \dots \\ &= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \dots) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned}$$

MDP: Example

Question (5 min)

Imagine a house cleaning robot. It can have three charge levels: *high*, *low* and *none*. At every point in time, the robot can decide to *recharge* or to *explore* unless it has no battery. When exploring, the charge level can reduce with probability ρ. Exploring is preferable to recharging, however it has to avoid running out of battery.

Formalize the above problem as an MDP.

MDP: Example

Solution

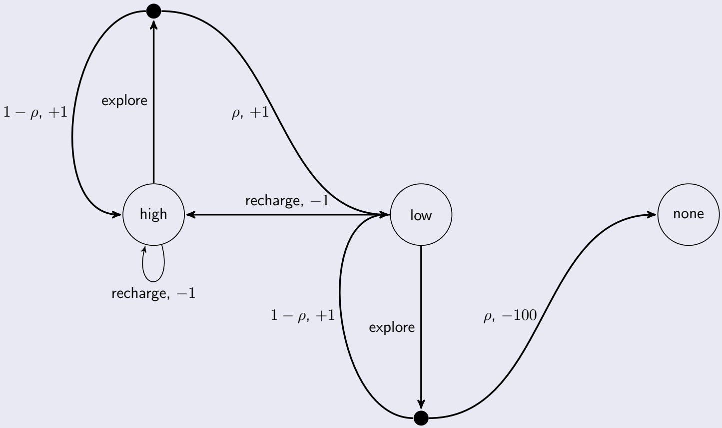
For the given problem, we set:

- $\mathcal{S} = \{\text{high}, \text{low}, \text{none}\}$
- $\mathcal{A} = \{\text{explore}, \text{recharge}\}$
- $\mathcal{R} = \{+1, -1, -100\}$ for exploring, recharging, and transitions leading to none, respectively.
- p has entries with value 1 for transitions $(\text{high}, -1, \text{high}, \text{recharge})$, $(\text{low}, -1, \text{high}, \text{recharge})$ and $(\text{none}, 0, \text{none}, \cdot)$. It further has entries with value ρ for transitions $(\text{high}, +1, \text{low}, \text{explore})$ and $(\text{low}, -100, \text{none}, \text{explore})$ and entries with value $1 - \rho$ for transitions $(\text{high}, +1, \text{high}, \text{explore})$ and $(\text{low}, +1, \text{low}, \text{explore})$.

MDP: Example

Solution

The transition graph therefore is:



Lecture Overview

Policies

- The policy defines the behaviour of the agent:
  - can be stochastic: $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$
  - or deterministic: $\pi(s) = a$
- Due to the Markov property, knowledge of the current state s is sufficient to make an informed decision.

Value Functions

- Value Function $v_\pi(s)$ is the expected return when starting in s and following π:

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \middle| S_t = s\right]$$

- Action-Value Function q_π is the expected return when starting in s, taking action a and following π thereafter:

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \middle| S_t = s, A_t = a\right]$$

- Simple connection:

$$v_\pi(s) = \mathbb{E}_\pi[q_\pi(s, \pi(s))] \tag{1}$$

Bellman Equation

- The Bellman Equation expresses a relationship between the value of a state and the values of its successor states
- The value function v_π is the unique solution to its Bellman Equation

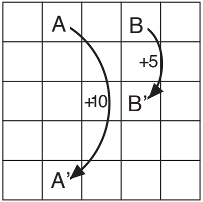
$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi[G_t | S_t = s] \\ &= \mathbb{E}_\pi[R_t + \gamma G_{t+1} | S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) \left[r + \gamma \mathbb{E}_\pi [G_{t+1} | S_{t+1} = s'] \right] \\ &= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) \left[r + \gamma v_\pi(s') \right] \end{aligned}$$

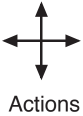
Bellman Equation for v_π

The Bellman Equation for v_π is defined as:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) \left[r + \gamma v_\pi(s') \right].$$

Bellman Equation: Example



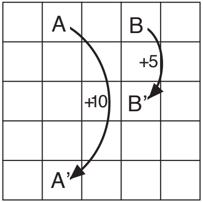


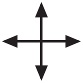
3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

Question (5 min)

Actions move the agent deterministically. Actions that would move the agent off the grid cost -1 with no state change. All other actions are free. However, every action performed by the agent in A moves it to A' with a reward of $+10$, each action in B moves it to B' with a reward of $+5$. Assume a uniform policy. v_π with a discounting factor of $\gamma = 0.9$ is to the right. Show exemplary for state $s_{0,0}$ with $v_\pi(s_{0,0}) = 3.3$ that the Bellman equation is satisfied.

Bellman Equation: Example





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

Solution

$$\begin{aligned}
v_\pi(s_{0,0}) &= 0.25 \cdot (-1 + \gamma \cdot 3.3) + \\
&\quad 0.25 \cdot (+0 + \gamma \cdot 8.8) + \\
&\quad 0.25 \cdot (+0 + \gamma \cdot 1.5) + \\
&\quad 0.25 \cdot (-1 + \gamma \cdot 3.3) \\
&= 3.3025 \\
&\approx 3.3
\end{aligned}$$

We equivalently obtain a corresponding system of equations for the Q-function:

Bellman Equation for q_π

The Bellman Equation for action-value function q_π is defined as:

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \sum_{a'} \pi(a'|s') q_\pi(s',a')\right].$$

Lecture Overview

Optimality of Policies

We consider a policy as optimal if the value (i.e. its expected return under the policy) in every state is at least as high as for any other policy:

Optimality of a policy π_*

A policy π_* is called *optimal* $:\Leftrightarrow$
For all $s \in S$:

$$v_{\pi_*}(s) \geq v_\pi(s) \text{ for all } \pi \tag{2}$$

The corresponding *optimal value function* is denoted by v_*.

- This requires a search among all, possibly infinitely many, policies. This seems to be rather impractical.
- Is there an easier way to check if a policy π and corresponding value function v_π is actually optimal?

Bellman Optimality Equation

Intuitively, the Bellman Optimality Equation expresses the fact that the value of a state under an optimal policy must equal the expected return for the best action from that state:

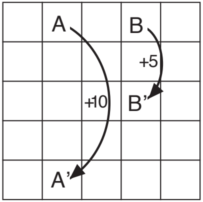
$$
\begin{aligned}
v_*(s) &= \max_a q_{\pi_*}(s, a) \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a] \\
&= \max_a \sum_{s', r} p(s', r | s, a)[r + \gamma v_*(s')]
\end{aligned}
$$

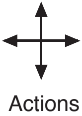
Bellman Optimality Equation for v_*

The Bellman Equation for the optimal value function v_* is defined as:

$$v_*(s) = \max_a \sum_{s', r} p(s', r | s, a)[r + \gamma v_*(s')].$$

Bellman Optimality Equation: Example





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

Non-optimality of the uniform random policy

$$\begin{aligned} v_\pi(s_{0,0}) = 3.3025 \quad &\neq \quad \max\{-1 + \gamma \cdot 3.3, 0 + \gamma \cdot 8.8, \\ &\qquad 0 + \gamma \cdot 1.5, -1 + \gamma \cdot 3.3\} \\ &= \quad 7.92 \end{aligned}$$

$\Rightarrow$ random policy π is *not* optimal.

Bellman Optimality Equation

Equivalently, there exists a Bellman optimality equation for Q-functions:

Bellman Optimality Equation for q_*

The Bellman Equation for the optimal action-value function q_* is:

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')].$$

Lecture Overview

Extensions to the MDP framework: POMDPs

A Partially Observable Markov Decision Process (POMDP) is a 6-tuple $\langle \mathcal{S}, \mathcal{A}, p, \mathcal{R}, \Omega, p_O \rangle$, where

- $\mathcal{S}$ is a finite number of states,
- $\mathcal{A}$ is a finite number of actions,
- Ω is a finite number of observations,
- p is the transition probability function $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$,
- $\mathcal{R}$ is a finite set of scalar rewards. We can then define expected reward $r(s, a) = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$ and $r(s, a, s') = \mathbb{E}[R_{t+1} | S_t = s, A_t = a, S_{t+1} = s']$.
- and p_0 is the observation probability function, $p_O : \Omega \times \mathcal{S} \mapsto [0, 1]$

The transitions of the true state S_t still obey the Markov property, *but* observations O_t do not.

Lecture Overview

Summary by Learning Goals

Having heard this lecture, you can now...

- formulate a given problem as an MDP
- explain the Markov-Property
- describe the relationship between rewards and values
- state the Bellmann Expectation and Optimality equations
- explain the relationship between optimal value function and optimal policy