Lecture 1 - Markov Decision Process

Guiliang Liu

The Chinese University of Hong Kong, Shenzhen

DDA4230: Reinforcement Learning
Course Page: [Click]

DDA 4230 Resources

Join our Wechat discussion group.

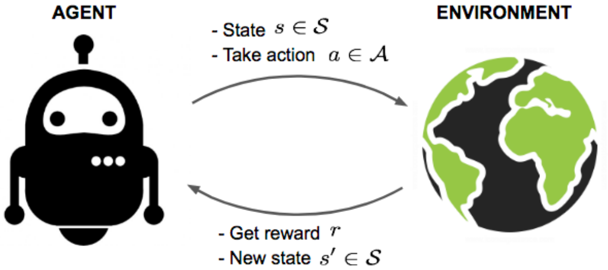
Check our course page.

Course Page Link (all the course relevant materials will be posted here):

https://guiliang.github.io/courses/cuhk-dda-4230/dda_4230.html

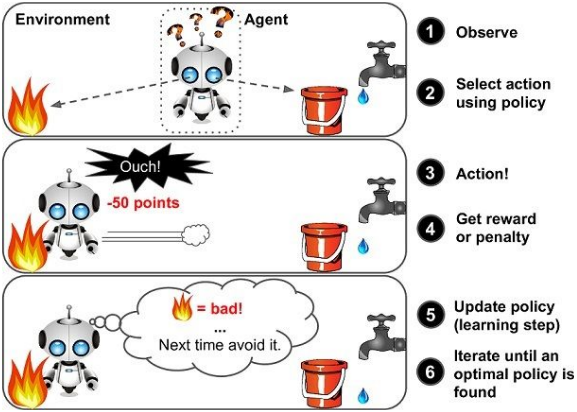
Reinforcement Learning

A reinforcement learning **agent** interacts with its **world** and from that learns how to **maximize cumulative reward** over time.



Reference: https://lilianweng.github.io/posts/2018-02-19-rl-overview/

Reinforcement Learning



Reference: https://www.odinschool.com/

No **teacher** or **knowledge** of the world model. Learn to act through **trial and error**.

- **E1**: Agent + Fire -> -50.
- **E2**: Agent + Bucket + Fire -> -50.
- **E3**: Agent + Bucket + Water + fire -> +50.
- **E4**: What will the agent do?

Connection with other learning problems

Reinforcement Learning

- No **teacher** or **knowledge** of the world model (e.g., environment).

Supervised Learning

Connection with other learning problems

Reinforcement Learning

- No **teacher** or **knowledge** of the world model (e.g., environment).
- **Interact** with the environment.

Supervised Learning

- Given a dataset, which consists of examples and labels (knowledge).

Connection with other learning problems

Reinforcement Learning

- No **teacher** or **knowledge** of the world model (e.g., environment).
- **Interact** with the environment.
- Making **sequential decisions**.

Supervised Learning

- Given a dataset, which consists of examples and labels (knowledge).
- No interaction. Only an offline dataset.

Connection with other learning problems

Reinforcement Learning

- No **teacher** or **knowledge** of the world model (e.g., environment).
- **Interact** with the environment.
- Making **sequential decisions**.
- Learn a policy to maximize **cumulative** rewards.

Supervised Learning

- Given a dataset, which consists of examples and labels (knowledge).
- No interaction. Only an offline dataset.
- Making one-step predictions.

Connection with other learning problems

Reinforcement Learning

- No **teacher** or **knowledge** of the world model (e.g., environment).
- **Interact** with the environment.
- Making **sequential decisions**.
- Learn a policy to maximize **cumulative** rewards.

Supervised Learning

- Given a dataset, which consists of examples and labels (knowledge).
- No interaction. Only an offline dataset.
- Making one-step predictions.
- Learn a predictor to maximizing point-wise prediction accuracy.

Connection with other learning problems

Reinforcement Learning

- No **teacher** or **knowledge** of the world model (e.g., environment).
- **Interact** with the environment.
- Making **sequential decisions**.
- Learn a policy to maximize **cumulative** rewards.

Unsupervised (Contrastive) Learning

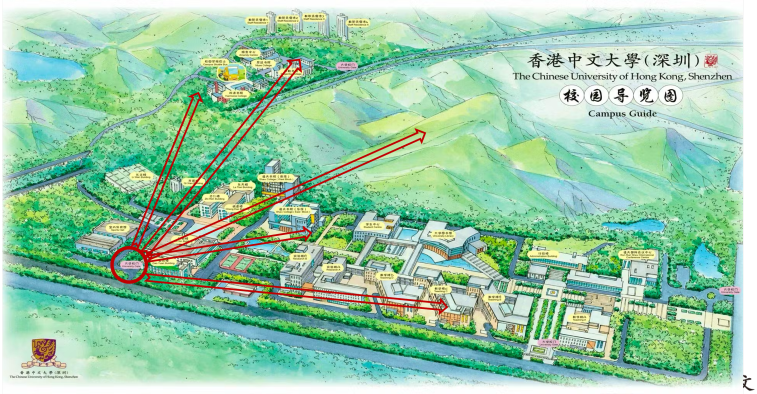
- Given a dataset, which consists of only examples (No labels).
- No interaction. Only an offline dataset.
- Learning latent structure of dataset.
- Learn the latent features for classification or identification.

Challenges in Reinforcement Learning

- How to balance exploration and exploitation?
- How to generalize its experience?
- How to model the delayed consequences of actions (look-ahead).

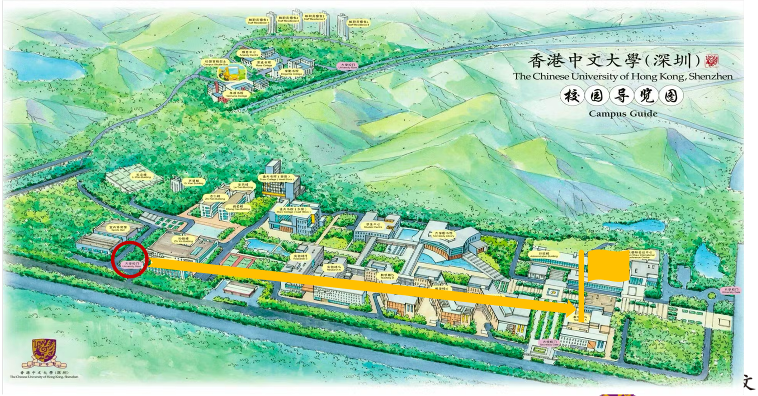
Challenges in Reinforcement Learning

Exploration: Collect information as much as you can.



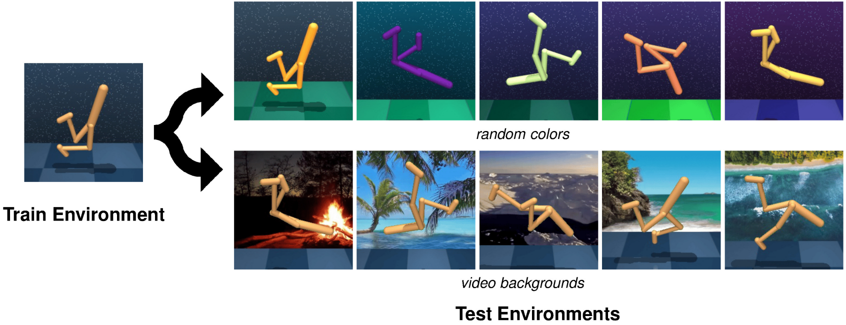
Challenges in Reinforcement Learning

exploitation: Reach the destination as fast as you can.



Challenges in Reinforcement Learning

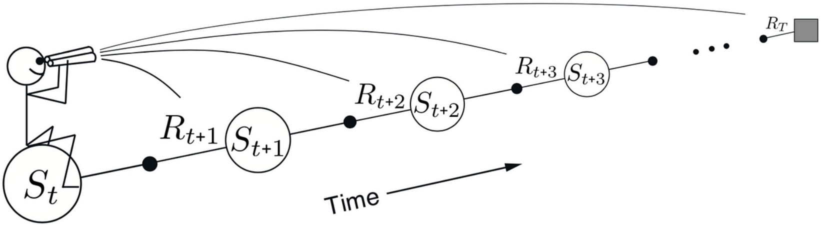
Generalization: Learn whether some actions are good/bad in previously unseen states.



香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Challenges in Reinforcement Learning

Look-ahead: estimate the delayed consequences of actions.



Markov Decision Process (MDP)

Discrete-time Markov decision process (MDP), denoted as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho_0, \gamma)$.

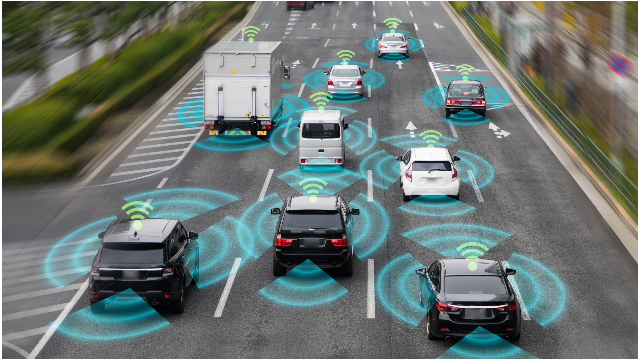
- $\mathcal{S}$ the state space;
- $\mathcal{A}$ the action space. $\mathcal{A}$ can depend on the state s for $s \in \mathcal{S}$;
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ the environment transition probability function;
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ the reward function;
- $\rho_0 \in \Delta(\mathcal{S})$ the initial state distribution;
- $\gamma \in [0, 1]$ the discount factor.

Note that $\Delta(\mathcal{X})$ denotes the set of all distributions over set $\mathcal{X}$.

Markov Decision Process (MDP)

Map the Reinforcement Learning (RL) environment to an MDP.

Application 1: Autonomous Driving.



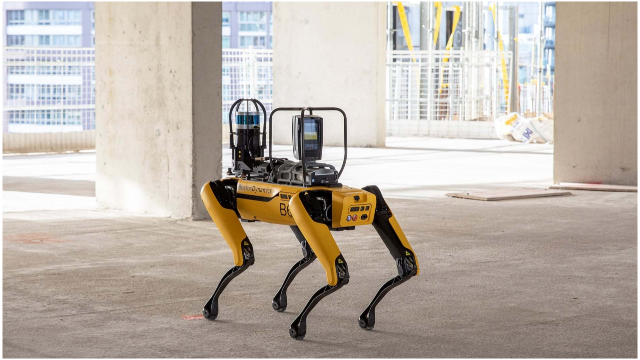
香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Reference: https://www.topgear.com/

Markov Decision Process (MDP)

Map the Reinforcement Learning (RL) environment to an MDP.

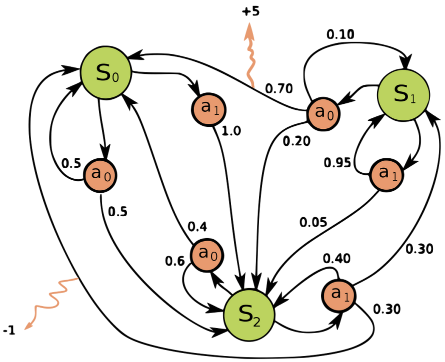
Application 2: Robot Control.



香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Reference: https://bostondynamics.com/

Markov Decision Process (MDP)

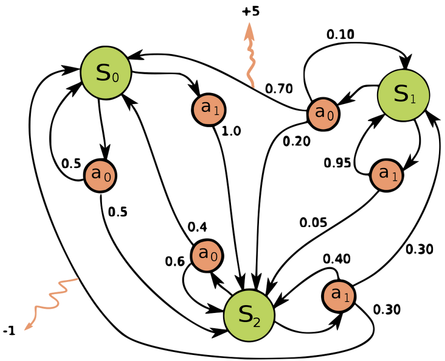


In an MDP, the choice of action a_t depends only on the state s_t. A policy defines the mapping from $\mathcal{S}$ to $\mathcal{A}$.

- Stochastic policy: $\pi : \mathcal{S} \to \Delta(\mathcal{A})$
- Deterministic policy: $a_t = \pi(s_t)$

Reference: https://en.wikipedia.org/wiki/Markov_decision_process

Markov Decision Process (MDP)

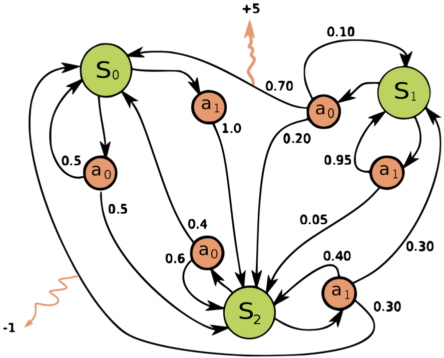


For $t = 0, 1, \ldots$, start with $s_0 \sim \rho_0$.

- The agent observes the state s_t;
- The agent's policy chooses an action $a_t = \pi(s_t)$;
- The agent receives the reward $r_t \sim \mathcal{R}(s_t, a_t)$;
- The environment transitions to a subsequent state: $s_{t+1} \sim \mathcal{T}(s_t, a_t)$.

Reference: https://en.wikipedia.org/wiki/Markov_decision_process

Markov Decision Process (MDP)



Trajectory generation.

- This process generates the sequence $s_0, a_0, r_0, s_1, \ldots,$ until when s_T is a terminal state, or indefinitely.
- The sequence up to time t is defined as the trajectory indexed by t, as $\tau_t = (s_0, a_0, r_0, s_1, \ldots, r_t)$.

Reference: https://en.wikipedia.org/wiki/Markov_decision_process

Important Variables in MDPs

- The return is defined as the discounted cumulative reward as a random variable.

$$R_t = \sum_{t' \geq t}^{\infty} \gamma^{t'} r_{t'} .$$

- The expectation of the return is the objective to be maximized by the agent

$$J = \mathbb{E}_{s_t, a_t, r_t, t \geq 0}\left[R_0\right] = \mathbb{E}_{s_t, a_t, r_t, t \geq 0}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \text{ and } \pi = \arg\max_{\pi} J$$

  The expectation is subject to random variables $(s_0, a_0, r_0, s_1, \ldots, r_\infty)$ with a complicate trajectory space $(\mathcal{S} \times \mathcal{A} \times \mathcal{R})^\infty$. The optimization problem is not characterisable (non-linear, non-convex, non-quadratic..) in general.

Important Variables in MDPs

The stochasticity of a Markov chain given the MDP and the policy may come from four components:

- Stochastic Markovian dynamics: $P_{\mathcal{T}}(s_{t+1}|s_t, a_t)$;
- Stochastic policies: $\pi(a_t|s_t)$;
- Initial state distribution: $\rho_0(s_0)$;
- Stochastic rewards: $P_{\mathcal{R}}(r_t|s_t, a_t)$;

Important Variables in MDPs

- The action value Q-function of a given policy π

$$Q^\pi(s,a) = \mathbb{E}_{s_t,a_t,r_t,t\geq 0}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s, a_0 = a\Big]$$

which is the expected return of policy π at state s after taking action a.

- The state value function of a given policy π

$$V^\pi(s) = \mathbb{E}_{a\sim\pi(s)}[Q^\pi(s,a)]$$

which is the expected return given the initial state only.

Important Variables in MDPs

Example: Game Go

- Rewards: $r_0 = 0, r_2 = 0, ..., r_{T-1} = 0$. If win $r_T = 1$ otherwise $r_T = 0$.
- Discount: $\gamma \to 1$.
- $Q^\pi(s, a)$: winning probability of making a move a under state s by following policy π.
- $V^\pi(s)$: winning probability of at the state s by following policy π.

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Important Variables in MDPs

- The advantage function of a given policy π can be defined as:

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s)$$

- Based on the value functions, define the temporal-difference error

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)\,.$$

Remark:

1. r_t: rewards at t.
2. $V(s_{t+1})$: expected cumulative rewards at $t+1, t+2, \ldots$.
3. $V(s_t)$: expected rewards at $t, t+1, t+2, \ldots$.
4. ~~if π is optimal, $\mathbb{E}[\delta_t] = 0$~~ This is incorrect.

Important Variables in MDPs

The TD error in reinforcement learning will go to zero when the agent's estimated value of a state-action pair perfectly matches the actual return it receives.

- The TD error going to zero does not necessarily mean the agent has learned the optimal policy. It just means the agent's value estimates are accurate under the current policy. The agent needs to explore more to find the optimal policy.
- This is an ideal case and may not occur in practical scenarios due to stochasticity in the environment, the function approximation errors,the inherent complexity and non-linearity of the problem

The Bellman Equation

- State-value Bellman equation (named after Richard E. Bellman):

$$V(s_t) = \mathbb{E}[r_t + \gamma V(s_{t+1})] \quad \text{and} \quad V(s_T) = \mathbb{E}[r_T]\,.$$

  for non-terminal and terminal states, respectively.

- Action-value Bellman equation:

$$Q(s_t, a_t) = \mathbb{E}[r_t + \gamma Q(s_{t+1}, a) \mid a \sim \pi(a \mid s_{t+1})] \quad \text{and} \quad Q(s_T, a_T) = \mathbb{E}[r_T]$$

  for non-terminal and terminal states, respectively.

Stationarity of MDPs and Agents

Stationarity MDP

- Markovian dynamics of s_{t+1} depends only on s_t and a_t as $s_{t+1} \sim P_{\mathcal{T}}(s_t, a_t)$.
- The reward r_t depends only on s_t and a_t as $r_t \sim P_{\mathcal{R}}(s_t, a_t)$.
- A policy is stationary if the action depends only on the state: $a_t \sim \pi(s_t)$.

Remark: If there is an optimal policy, there is an optimal stationary policy if the process has a non-fixed horizon.

Non-Stationarity MDP

- The transition dynamics depend on the time t as $s_{t+1} \sim P_{\mathcal{T},t}(s_t, a_t)$.
- The reward depend on the time t as $r_t \sim P_{\mathcal{R},t}(s_t, a_t)$.
- A policy is non-stationary if the action also depends on t: $a_t \sim \pi_t(s_t)$.

Stationarity of MDPs and Agents

- If the planning horizon H is finite, we should assume the policy is not stationary since $Q_t^{\pi}(s,a) = \mathbb{E}[\sum_{\iota=0}^{H-t-1} \gamma^{\iota} r(s_{\iota}, a_{\iota})]$,

$$\text{If } t \leq t', Q_t^{\pi}(s,a) \geq Q_{t'}^{\pi}(s,a)$$

  Since the value function depends on time, the corresponding policy must depend on time as well.

- If the planning horizon H is infinite, we commonly apply stationary policy.

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

State and Action Spaces

Two common settings of the state space and the action space are:

- $\mathcal{S} \in \mathbb{R}^n$ the n dimensional state space, $\mathcal{A} \in \mathbb{R}^m$ the m dimensional action space;
- $\mathcal{S} \in [n]$ the size-n discrete state space, $\mathcal{A} \in [m]$ the size-m discrete action space.

Discount of Rewards

The discount factor $\gamma \in [0, 1]$ balances the short-term and long-term rewards. When the objective is discounted $(\gamma < 1)$

$$R_0 = r_0 + \gamma r_1 + \gamma^2 r_2 + \dots,$$

Two extreme cases are $\gamma = 0$ and $\gamma = 1$, where the former corresponds to $R_0 = r_0$ as a one-step MDP and the latter corresponds to $R_0 = r_0 + r_1 + r_2 + \dots$.

Agents of RL

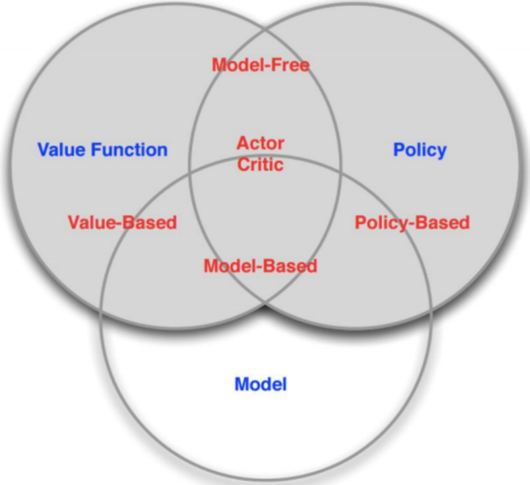
We can classify our agents in a number of ways:

Agent type	Policy	Value Function	Model
Value-based	Implicit	✓	?
Policy-based	✓	✗	?
Actor-critic	✓	✓	?
Model-based	?	?	✓
Model-free	?	?	✗

- ✓ indicates that the agent has the component.
- x indicates that it must not have the component.
- ? indicates that the agent may have that component.

Agents of RL

Classification of different reinforcement learning agents.



Classification of Markov structures

Markov structure		Do actions have influence over the state transitions?	
		NO	YES
Are the states fully observable?	YES	Markov process (Markov chain)	Markov decision process
	NO	Hidden Markov model	Partially observable Markov decision process

Classification of Markov structures

Fully observable Markov decision process

- All the players are observable.

Classification of Markov structures

Partially observable Markov decision process

- Only a partial number of the players are observable.

Question and Answering (Q&A)