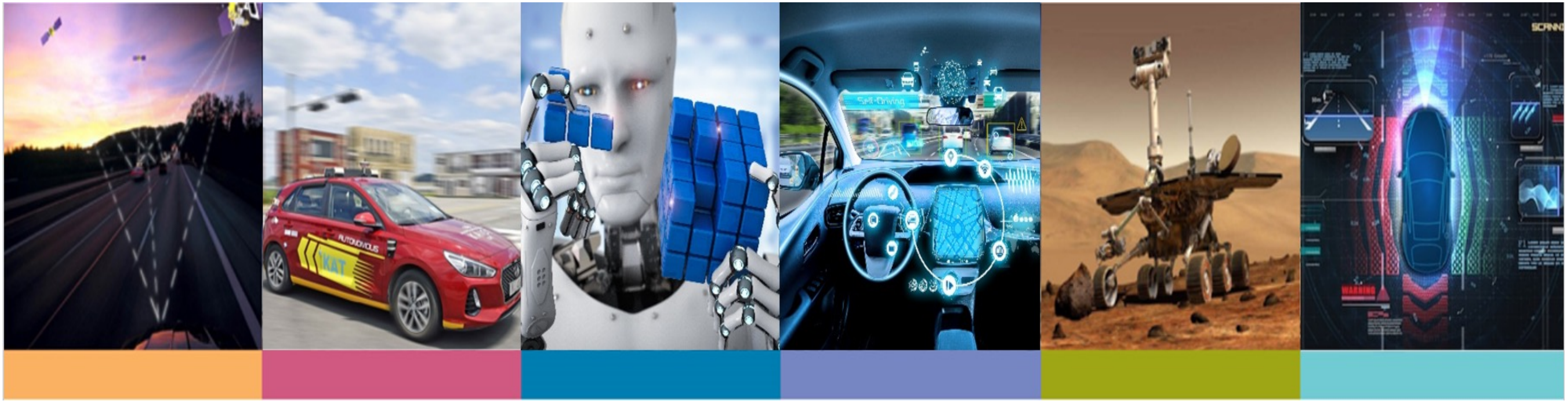


Failure is not the Opposite of Success: Reinforcement Learning

1st Seminar, 2023 AVE Lab Summer Internship

Hongtae Kim

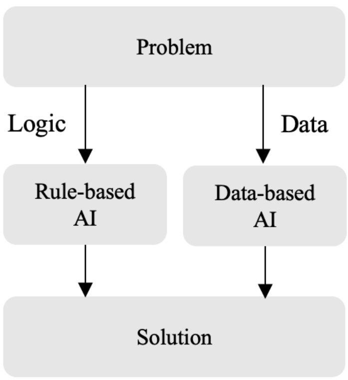
2023.07.20

Contents

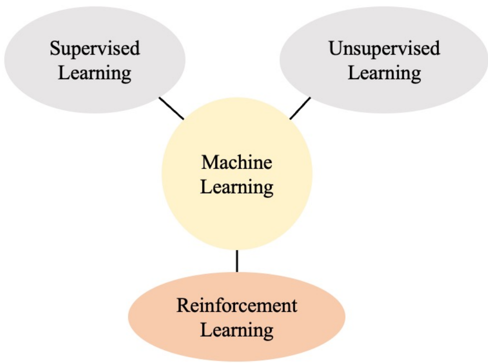
KAIST Cho Chun Shik Graduate School of Mobility | avelab Autonomous Vehicles and Electronics Lab

Reinforcement Learning

- Trend : Rule-based AI → Data-based AI
- Supervised Learning, Unsupervised Learning → Only one time solution
- When reinforcement learning can be used?
    - Answer to an action sequence, which is a series of decisions over time
    - Action sequence (also called ‘Policy’)

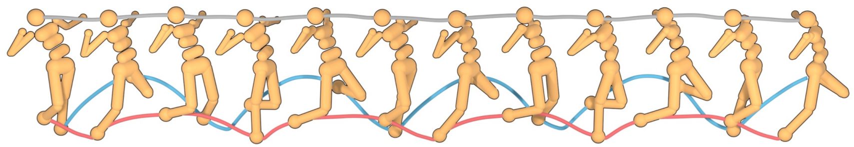


Rule-based AI & Data-based AI



Machine Learning

Supervised Learning, Classification



Reinforcement Learning, Action Sequences

- Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." 2009 IEEE conference on computer vision and pattern recognition. Ieee, 2009.
- Janner, Michael, Qiyang Li, and Sergey Levine. "Offline reinforcement learning as one big sequence modeling problem." *Advances in neural information processing systems* 34 (2021): 1273-1286.

Reinforcement Learning – Setting

- The general RL problem is formalized as a **discrete-time stochastic control process** where an agent interacts with its environment as follows:

Step 1) Agent starts in a given environment state $\boldsymbol{s_0}$, $\boldsymbol{s_0} \in \boldsymbol{S}$

and gathering an initial observation $\boldsymbol{\omega_0}$, $\boldsymbol{\omega_0} \in \boldsymbol{\Omega}$

Step 2) At each time step t, the agent take an action $\boldsymbol{a_t}$, $\boldsymbol{a_t} \in \boldsymbol{A}$

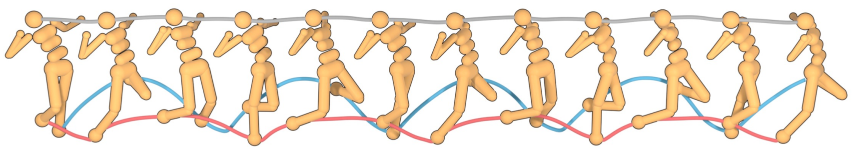
Action follows three consequences

1) obtains a reward $\boldsymbol{r_t}$

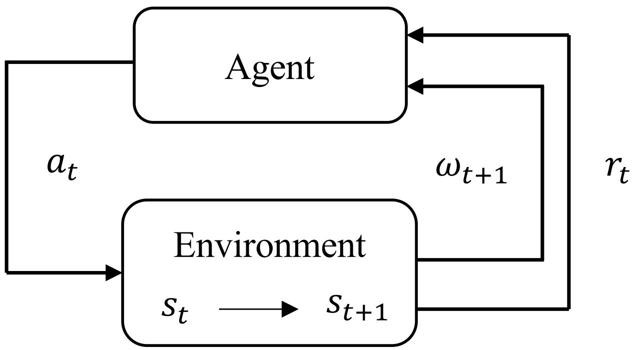
2) state transitions to $\boldsymbol{s_{t+1}}$

3) obtains an observation $\boldsymbol{\omega_{t+1}}$

discrete-time stochastic control process



t t+1 t+2 t+3 …



RL Diagram

Markov Decision Process (MDP)

- How can we solve RL? MDP!
- MDP provides **mathematical model**, discrete-time stochastic control process problems
- RL is an algorithm for solving problems represented by MDP

Markov Decision Process (MDP)

Markov Property

Action

Sequences

$$\mathbb{P}[S_{t+1} \mid S_0, \cdots, S_t] = \mathbb{P}[S_{t+1} \mid S_t]$$

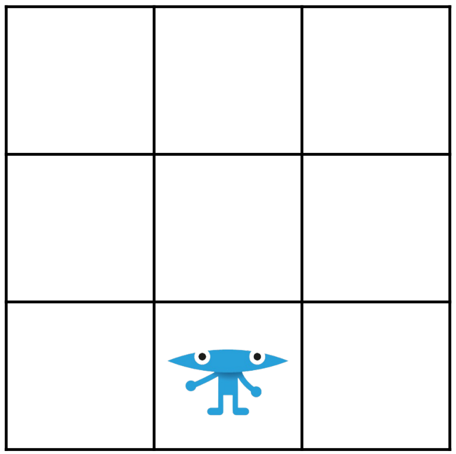
State	Action	State Transition Probability	Reward	Discount Factor
S	**A**	**P**	**R**	$\boldsymbol{\gamma}$

Components of MDP

Markov Decision Process (MDP) – State

- How can we solve RL? MDP!
- MDP provides **mathematical model**, discrete-time stochastic control process problems
- RL is an algorithm for solving problems represented by MDP

State	Action	State Transition Probability	Reward	Discount Factor
S	**A**	**P**	**R**	γ

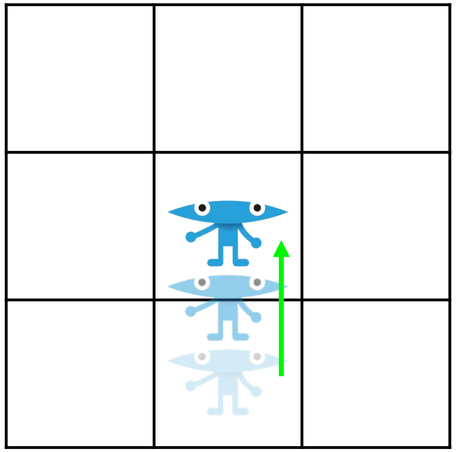


- **State** of the agent in the environment
- Grid world : Location (x, y)
- s_0 : starting state
- s : current state
- s' : next state
- s_t : state at time t

Markov Decision Process (MDP) – Action

- How can we solve RL? MDP!
- MDP provides **mathematical model**, discrete-time stochastic control process problems
- RL is an algorithm for solving problems represented by MDP

State	Action	State Transition Probability	Reward	Discount Factor
S	**A**	**P**	**R**	γ

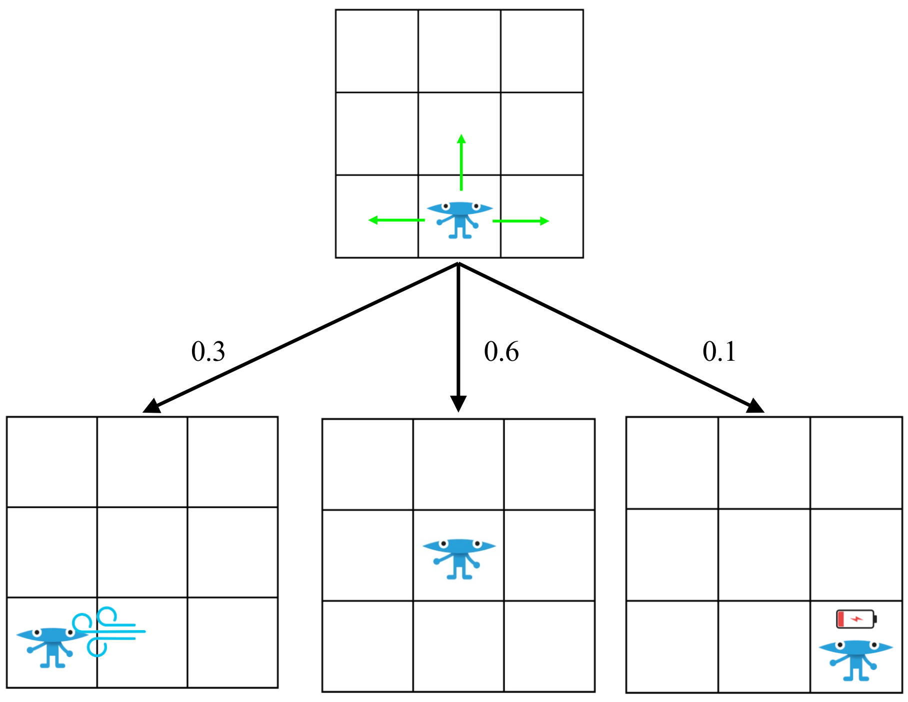


- Agent can change its state by taking an **action**
- Up, Down, Left, Right
- $s \rightarrow s'$

Markov Decision Process (MDP) – State Transition Probability

- How can we solve RL? MDP!
- MDP provides **mathematical model**, discrete-time stochastic control process problems
- RL is an algorithm for solving problems represented by MDP

State	Action	State Transition Probability	Reward	Discount Factor
S	A	**P**	R	γ

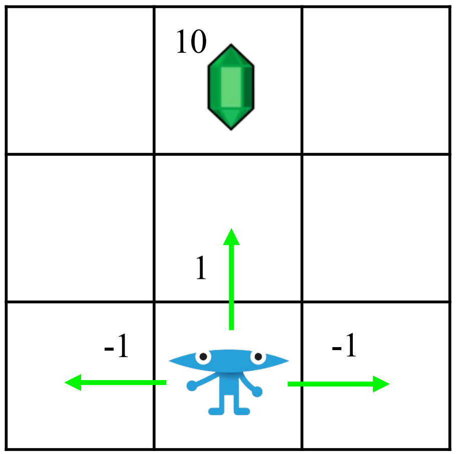


- When an action is taken, the state becomes **probabilistic** (not deterministic)
- Disturbances and control errors
- $P(s' | s, a)$

Markov Decision Process (MDP) – Reward

- How can we solve RL? MDP!
- MDP provides **mathematical model**, discrete-time stochastic control process problems
- RL is an algorithm for solving problems represented by MDP

State	Action	State Transition Probability	Reward	Discount Factor
S	**A**	**P**	**R**	γ

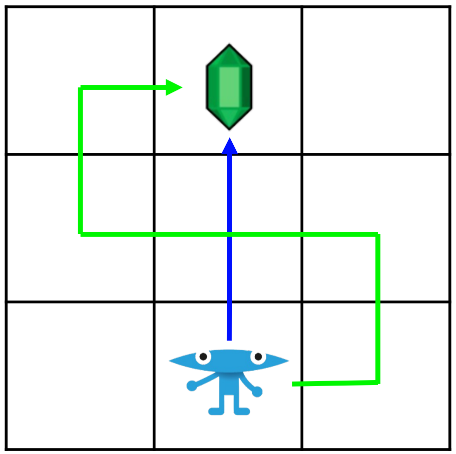


- Value that received when a specific Action is taken in a specific State
- Determine good and bad actions
- Start of Value Function
- $R(s, a, s')$

Markov Decision Process (MDP) – Discount Factor

- How can we solve RL? MDP!
- MDP provides **mathematical model**, discrete-time stochastic control process problems
- RL is an algorithm for solving problems represented by MDP

State	Action	State Transition Probability	Reward	Discount Factor
S	**A**	**P**	**R**	$\boldsymbol{\gamma}$



- Penalty for future rewards
- Ensure rewards converge to finite values
- $0 \leq \gamma \leq 1$
- $\gamma \approx 0$, Future rewards are less valuable
- Blue : $1 + 10 = 11$
- Green : $1 + 1 + 1 + 1 + 1 + 10 = 15$

Value Function – Return & Q-Function

- Returns are totals for single episodes
- But, RL handles stochastic situations (not deterministic)
- So, we use the expected return
- Q Function (= Action Value Function)
  - Agent's expected reward for a given **state-action pair**
  - Indicator of **what actions** the agent will take

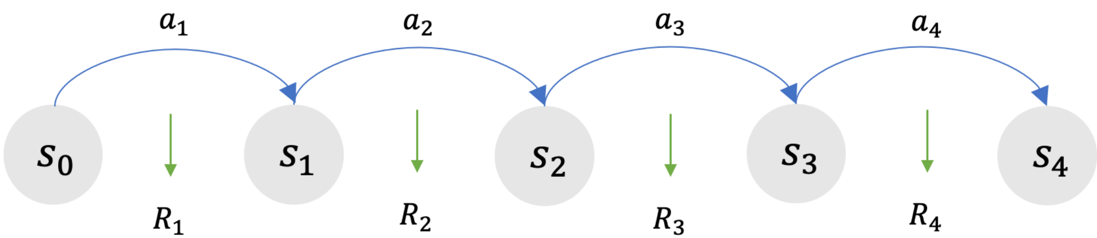


Diagram of Return

Single episode

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$
$$= \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Return function

Expected value of episode

$$q_\pi(s, a) = E_\pi[G_t \mid S_t = s, A_t = a]$$

Q Function(= Action Value Function)

Value Function – Policy

- Policy is a **strategy** that determines what actions to take in each state in a given episode
- Our goal: Find policy(π) that **maximize the Value**

$\boldsymbol{\pi(s,a)}$

Policy

↓ Indicator

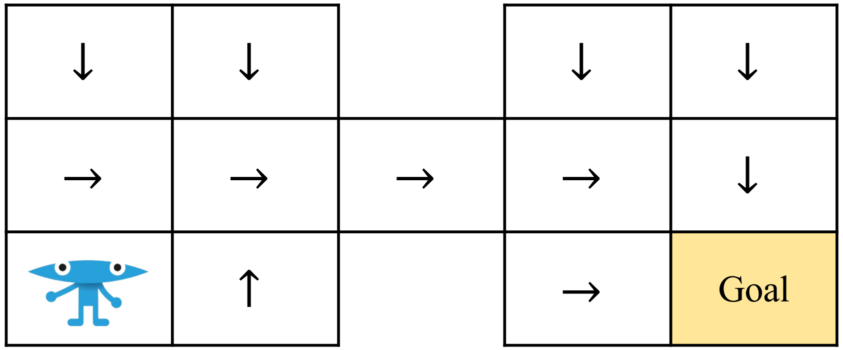
$E_\pi[G_t \mid S_t = s, A_t = a]$

Q Function

↓ Maximize

$\boldsymbol{\pi^*(s,a)}$

Optimal policy (π^*)

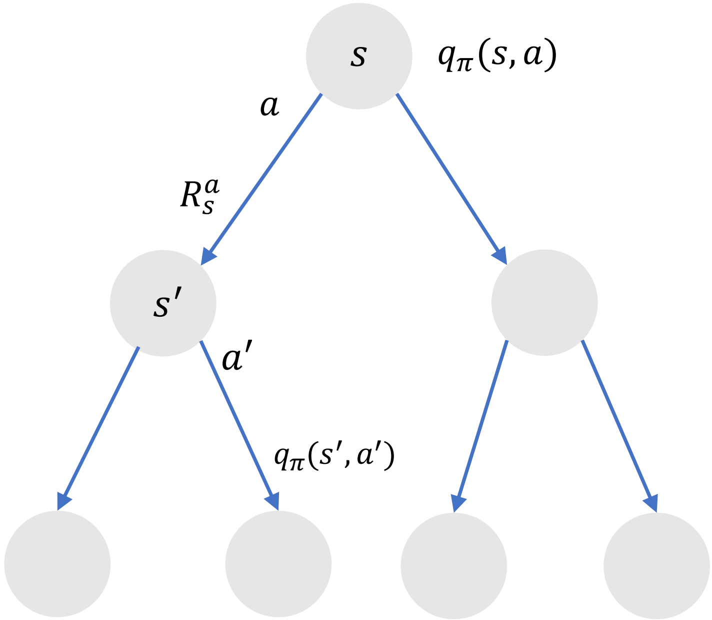


Optimal policy (π^*)

If every state can take 4 actions, there are 4^{13} policies

Bellman Equation

- Bellman Equation to find the optimal policy
- Relationship between the **value function of a state at time t** and the **value function of a state at time t+1**
- Find the **value function at time t** through the **value function at time t+1**



$$q_\pi(s,a) = R_s^a$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a$$

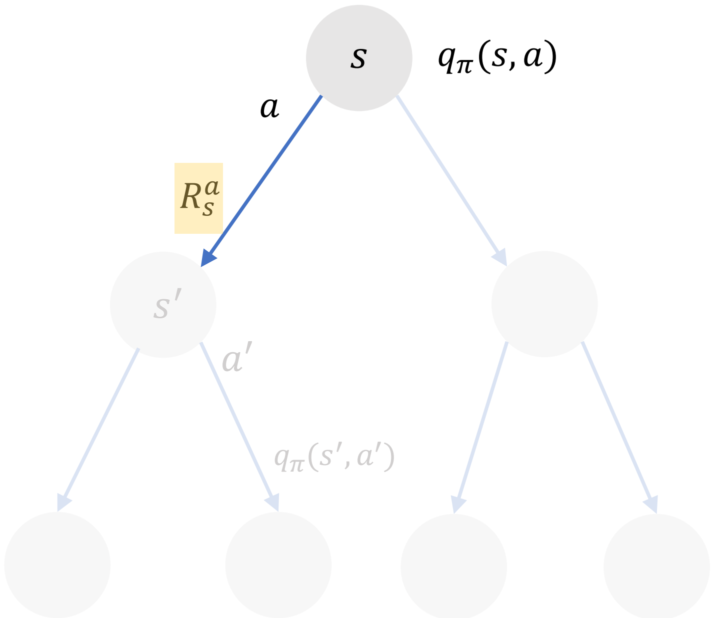
$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')$$

Bellman Equation

Bellman Equation

- Bellman Equation to find the optimal policy
- Relationship between the **value function of a state at time t** and the **value function of a state at time t+1**
- Find the **value function at time t** through the **value function at time t+1**



$$q_\pi(s,a) = \boxed{R_s^a}$$

Immediate reward

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a$$

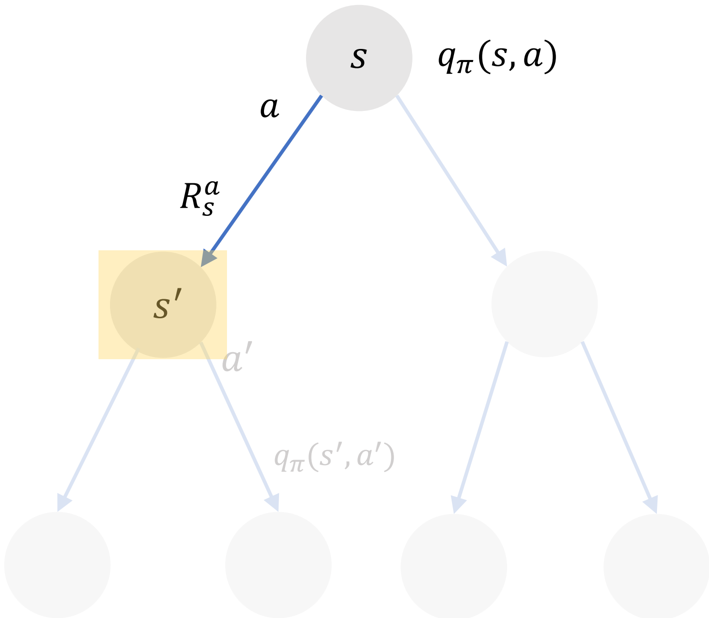
$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')$$

Bellman Equation

Bellman Equation

- Bellman Equation to find the optimal policy
- Relationship between the **value function of a state at time t** and the **value function of a state at time t+1**
- Find the **value function at time t** through the **value function at time t+1**



$$q_\pi(s,a) = R_s^a$$

$$q_\pi(s,a) = R_s^a + \gamma \boxed{\sum_{s' \in S} P_{ss'}^a}$$

State Transition Probability

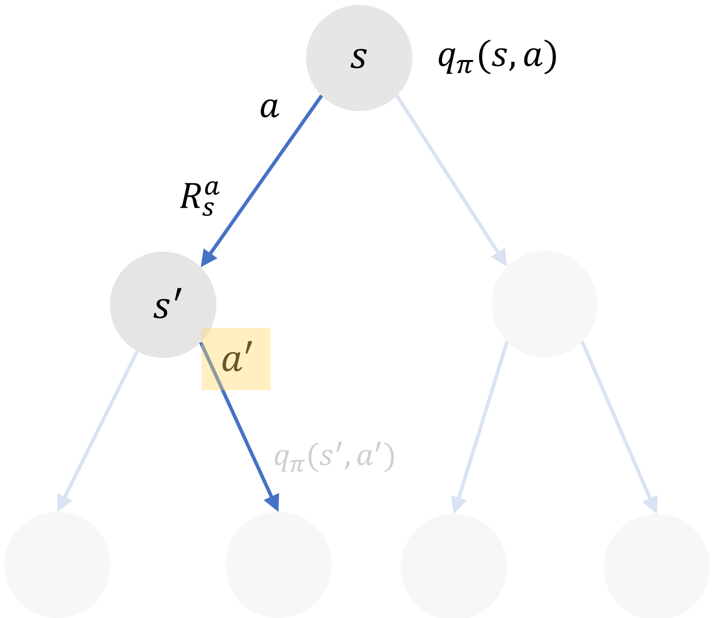
$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')$$

Bellman Equation

Bellman Equation

- Bellman Equation to find the optimal policy
- Relationship between the **value function of a state at time t** and the **value function of a state at time t+1**
- Find the **value function at time t** through the **value function at time t+1**



$$q_\pi(s,a) = R_s^a$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a$$

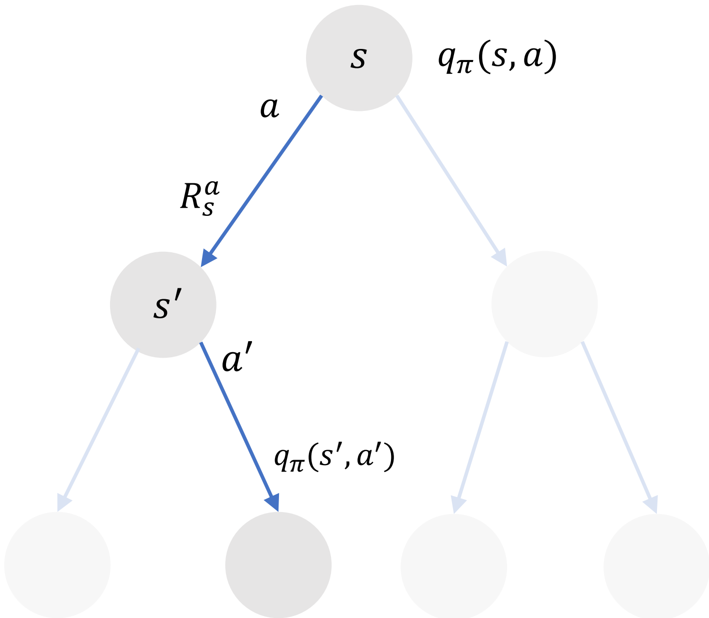
$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \underbrace{\pi(a'|s')}_{\text{Probability of Action}}$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')$$

Bellman Equation

Bellman Equation

- Bellman Equation to find the optimal policy
- Relationship between the **value function of a state at time t** and the **value function of a state at time t+1**
- Find the **value function at time t** through the **value function at time t+1**



$$q_\pi(s,a) = R_s^a$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')$$

$$q_\pi(s,a) = R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s')\, q_\pi(s',a')$$

Q value of s'

Bellman Equation

Bellman Equation

- Bellman Equation to find the optimal policy
- Relationship between the **value function of a state at time t** and the **value function of a state at time t+1**
- Find the **value function at time t** through the **value function at time t+1**

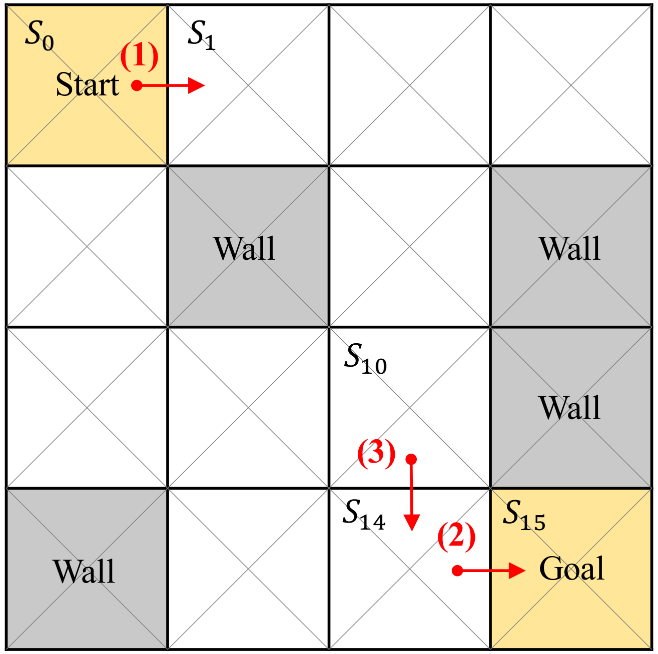
$$q_\pi(s,a) = \mathbb{E}[R_{t+1} + \gamma\, q(s_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$$

$$q_\pi(s,a) = \underbrace{R_t^a}_{\textbf{Immediate Reward}} + \underbrace{\gamma \sum_{a \in A} P_{ss'}^a\, v_\pi(s_{t+1}) \sum_{a \in A} \pi(a|s_{t+1})\, q_\pi(s_{t+1}, a)}_{\textbf{Next Value}}$$

Bellman Equation for Q-Function

Q-Learning

- Initialization step, Exploration step, Q value update step
- For simplicity, the Discount Factor (γ) and Probability (P) are not considered
- Initial state
  1) All 64 Q values are 0
  2) Only $R_{S_{15},L} = 1$, all others 0



Exploration step & Q value update step

(1) From S_0 to S_1
- $Q(S_0, A_R) = R + \max_a Q(S_1, A) = 0 + \max\{0,0,0,0\} = 0$

(2) From S_{14} to S_{15}
- $Q(S_{14}, A_R) = R + \max_a Q(S_{15}, A) = 1 + \max\{0,0,0,0\} = 1$

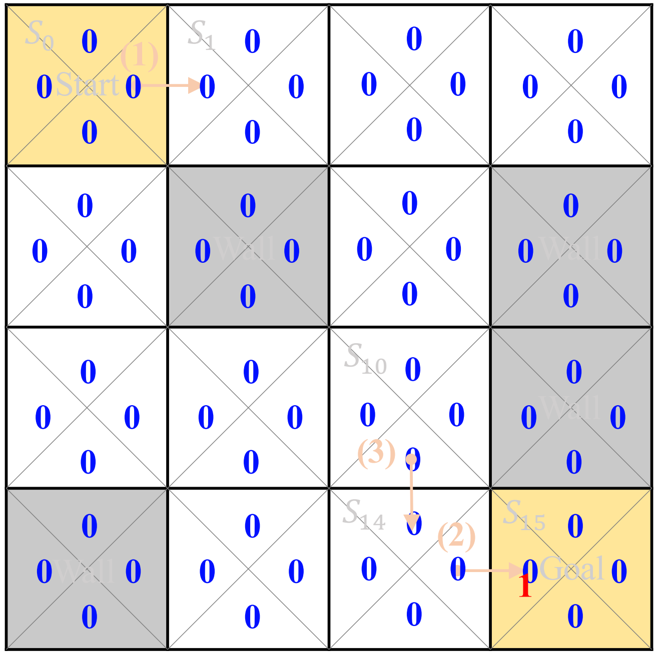
(3) From S_{10} to S_{14}
- $Q(S_{10}, A_D) = R + \max_a Q(S_{14}, A) = 0 + \max\{0,1,0,0\} = 1$

Q-Learning

- Initialization step, Exploration step, Q value update step
- For simplicity, the Discount Factor (γ) and State Transition Probability (P) are not considered
- **Initial state**

  1) All 64 Q values are 0

  2) Only $R_{S_{15},L} = 1$, all others 0



Blue : Q-Table

Red : Reward

Exploration step & Q value update step

(1) From S_0 to S_1

- $Q(S_0, A_R) = R + \max_a Q(S_1, A) = 0 + \max\{0,0,0,0\} = 0$

(2) From S_{14} to S_{15}

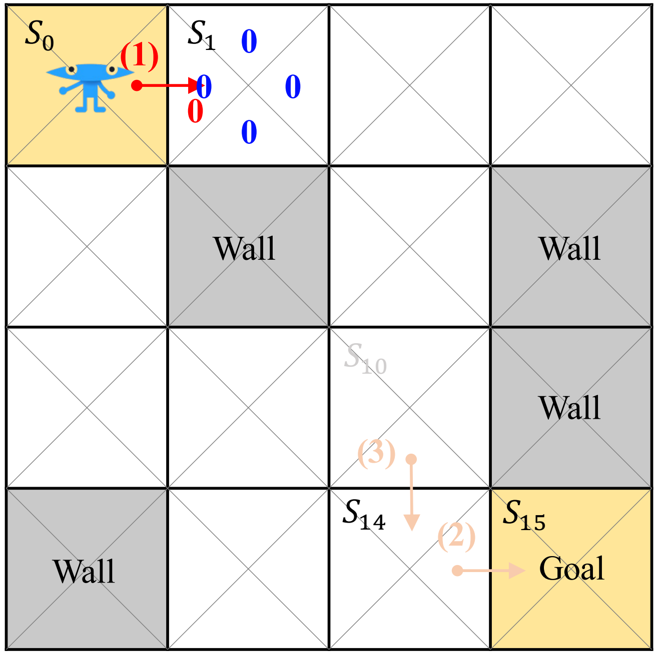
- $Q(S_{14}, A_R) = R + \max_a Q(S_{15}, A) = 1 + \max\{0,0,0,0\} = 1$

(3) From S_{10} to S_{14}

- $Q(S_{10}, A_D) = R + \max_a Q(S_{14}, A) = 0 + \max\{0,1,0,0\} = 1$

Q-Learning

- Initialization step, Exploration step, Q value update step
- For simplicity, the Discount Factor (γ) and State Transition Probability (P) are not considered
- Initial state
    1) All 64 Q values are 0
    2) Only $R_{S_{15},L} = 1$, all others 0



Exploration step & Q value update step

(1)From S_0 to S_1

- $Q(S_0, A_R) = R + \max_a Q(S_1, A) = 0 + \max\{0,0,0,0\} = 0$

(2) From S_{14} to S_{15}

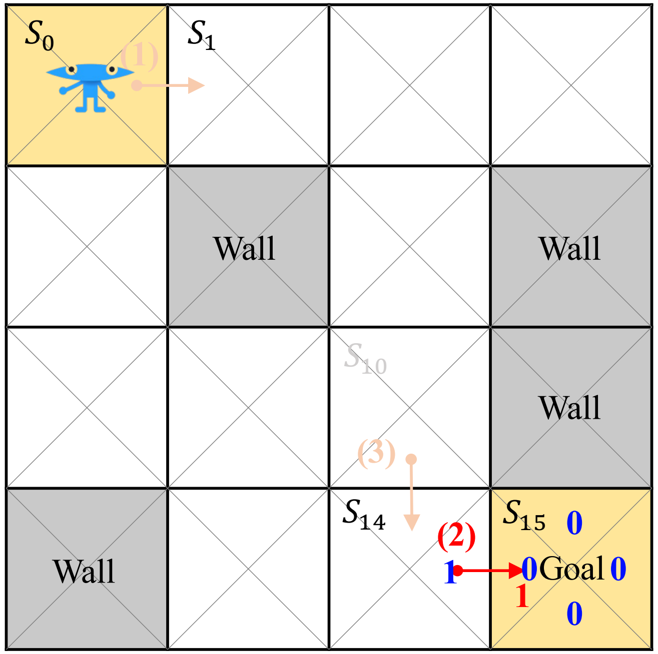
- $Q(S_{14}, A_R) = R + \max_a Q(S_{15}, A) = 1 + \max\{0,0,0,0\} = 1$

(3) From S_{10} to S_{14}

- $Q(S_{10}, A_D) = R + \max_a Q(S_{14}, A) = 0 + \max\{0,1,0,0\} = 1$

Q-Learning

- Initialization step, Exploration step, Q value update step
- For simplicity, the Discount Factor (γ) and State Transition Probability (P) are not considered
- Initial state
    1) All 64 Q values are 0
    2) Only $R_{S_{15},L} = 1$, all others 0



Exploration step & Q value update step

(1) From S_0 to S_1

- $\mathrm{Q}(S_0, A_R) = \mathrm{R} + \max_a Q(S_1, A) = 0 + \max\{0,0,0,0\} = 0$

(2) From S_{14} to S_{15}

- $\mathrm{Q}(S_{14}, A_R) = \mathrm{R} + \max_a Q(S_{15}, A) = 1 + \max\{0,0,0,0\} = 1$

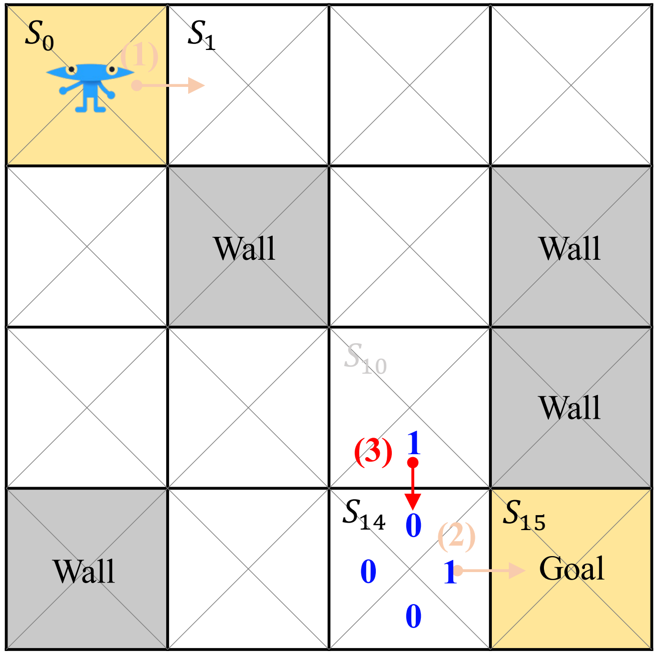
(3) From S_{10} to S_{14}

- $\mathrm{Q}(S_{10}, A_D) = \mathrm{R} + \max_a Q(S_{14}, A) = 0 + \max\{0,1,0,0\} = 1$

KAIST Cho Chun Shik Graduate School of Mobility | avelab Autonomous Vehicles and Electronics Lab

Q-Learning

- Initialization step, Exploration step, Q value update step
- For simplicity, the Discount Factor (γ) and State Transition Probability (P) are not considered
- Initial state
  1) All 64 Q values are 0
  2) Only $R_{S_{15},L} = 1$, all others 0



Exploration step & Q value update step

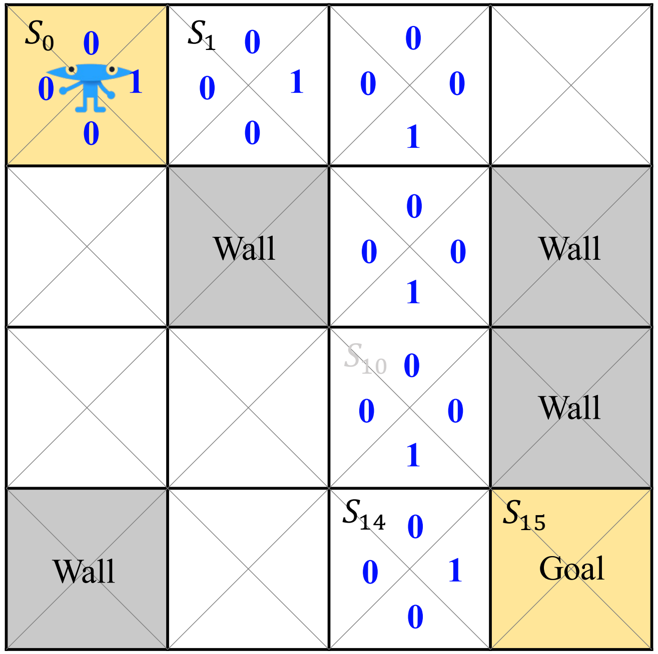
(1) From S_0 to S_1
- $Q(S_0, A_R) = R + \max_a Q(S_1, A) = 0 + \max\{0,0,0,0\} = 0$

(2) From S_{14} to S_{15}
- $Q(S_{14}, A_R) = R + \max_a Q(S_{15}, A) = 1 + \max\{0,0,0,0\} = 1$

(3) From S_{10} to S_{14}
- $Q(S_{10}, A_D) = R + \max_a Q(S_{14}, A) = 0 + \max\{0,1,0,0\} = 1$

Q-Learning

- Initialization step, Exploration step, Q value update step
- For simplicity, the Discount Factor (γ) and State Transition Probability (P) are not considered
- Initial state
  1) All 64 Q values are 0
  2) Only $R_{S_{15},L} = 1$, all others 0



Exploration step & Q value update step

(1) From S_0 to S_1
- $Q(S_0, A_R) = R + \max_a Q(S_1, A) = 0 + \max\{0,0,0,0\} = 0$

(2) From S_{14} to S_{15}
- $Q(S_{14}, A_R) = R + \max_a Q(S_{15}, A) = 1 + \max\{0,0,0,0\} = 1$

(3) From S_{10} to S_{14}
- $Q(S_{10}, A_D) = R + \max_a Q(S_{14}, A) = 0 + \max\{0,1,0,0\} = 1$

Outro – Summary

Reinforcement Learning

Discrete-time Stochastic Control Process

→

Markov Decision Process

Mathematical model

↙

Value Function

Action selection indicators

→

Bellman Equation

To find the optimal policy
Optimal policy (π^*)

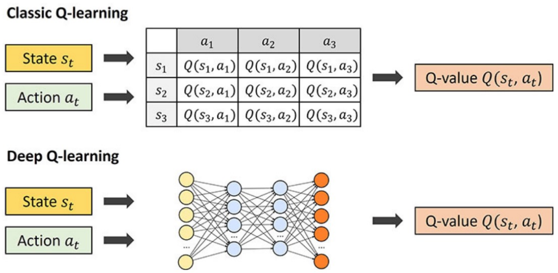
↙

Q-Learning

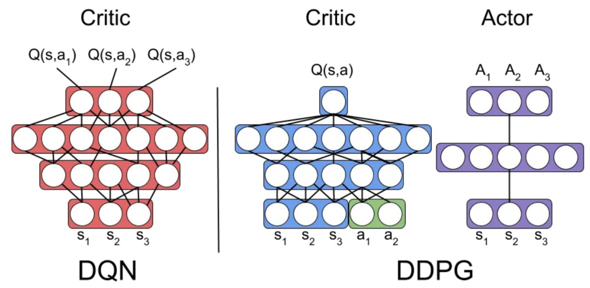
Model-Free RL Algorithm

Outro – 2nd Seminar, August 9 2023

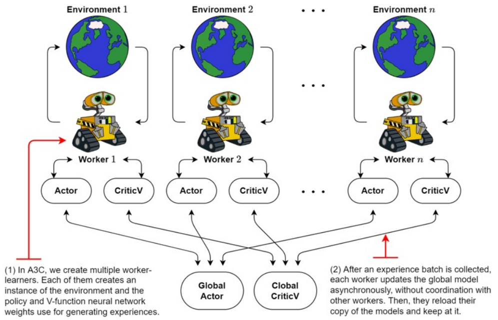
- **DQN** : Combining deep learning neural networks and Q-learning
- **DDPG** : Operates on a continuous behavior space, an extended form of the Actor-Critic method
- **A3C** : Multiple agents interact with the environment in parallel to learn
- **Multi Agent Reinforcement Learning (MARL)** : Dealing with multiple agents interact



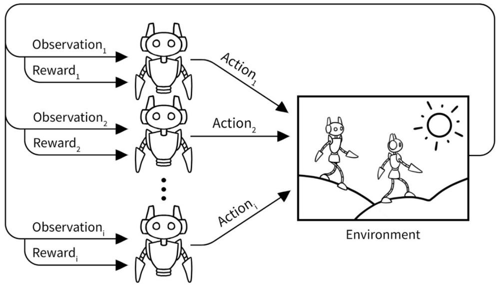
Deep Q-Network



Deep Deterministic Policy Gradient



Asynchronous Advantage Actor-Critic



Multi Agent Reinforcement Learning

- Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).
- Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).
- Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." International conference on machine learning. PMLR, 2016.

KAIST Cho Chun Shik Graduate School of Mobility | avelab Autonomous Vehicles and Electronics Lab