Reinforcement Learning

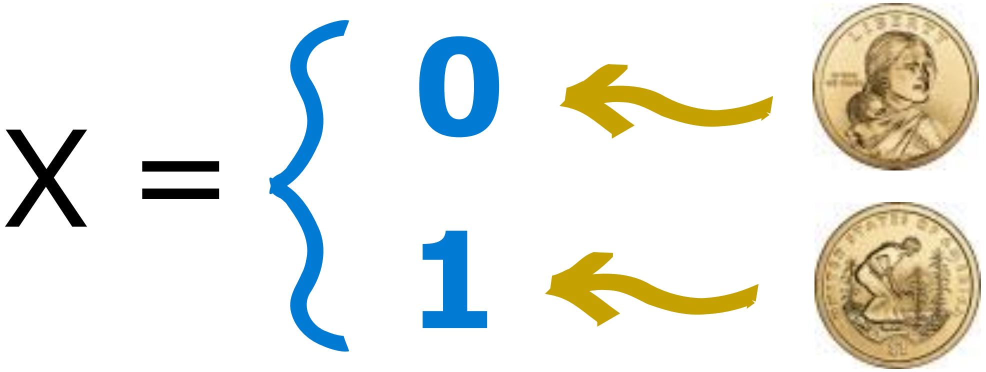
Shusen Wang

A little bit probability theory…

Random Variable

- Random variable: a variable whose values depend on outcomes of a random event.
- Uppercase letter X for random variable.

Random Variable *Possible Values* *Random Events* *Probabilities*



$$X = \begin{cases} 0 \\ 1 \end{cases}$$

$\mathbb{P}(X = 0) = 0.5$

$\mathbb{P}(X = 1) = 0.5$

Random Variable

- Random variable: a variable whose values depend on outcomes of a random event.
- Uppercase letter X for random variable.
- Lowercase letter x for an observed value.
- For example, I flipped a coin 4 times and observed:
    - $x_1 = 1,$
    - $x_2 = 1,$
    - $x_3 = 0,$
    - $x_4 = 1.$

Probability Density Function (PDF)

- PDF provides a relative likelihood that the value of the random variable would equal that sample.

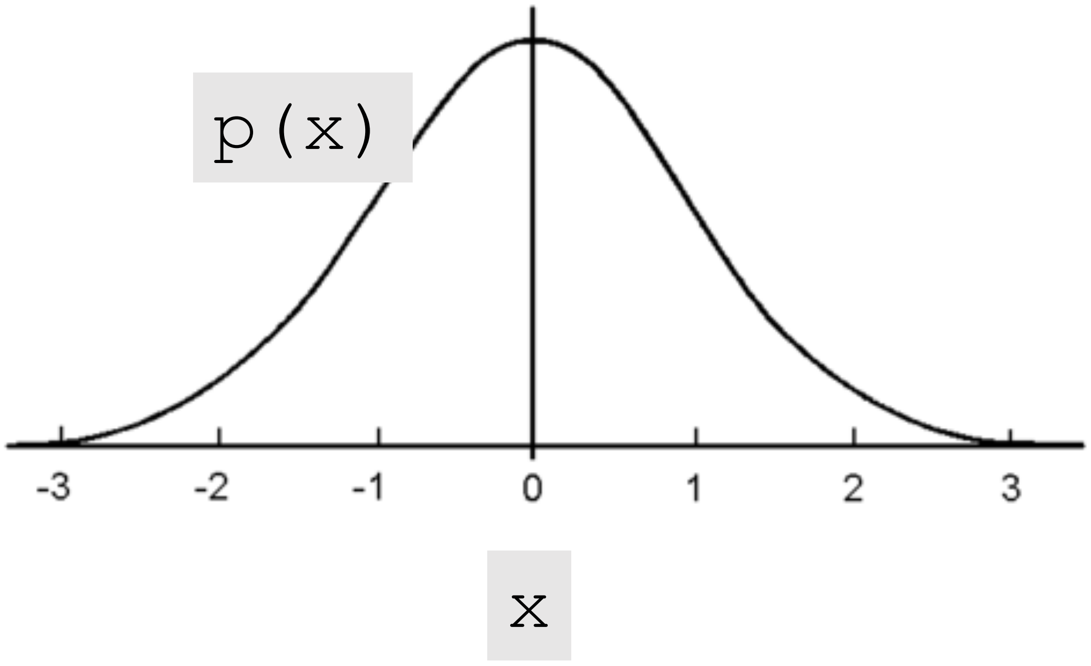
Probability Density Function (PDF)

- PDF provides a relative likelihood that the value of the random variable would equal that sample.

Example: Gaussian distribution

- It is a continuous distribution.
- PDF:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\,\sigma^2}\right).$$



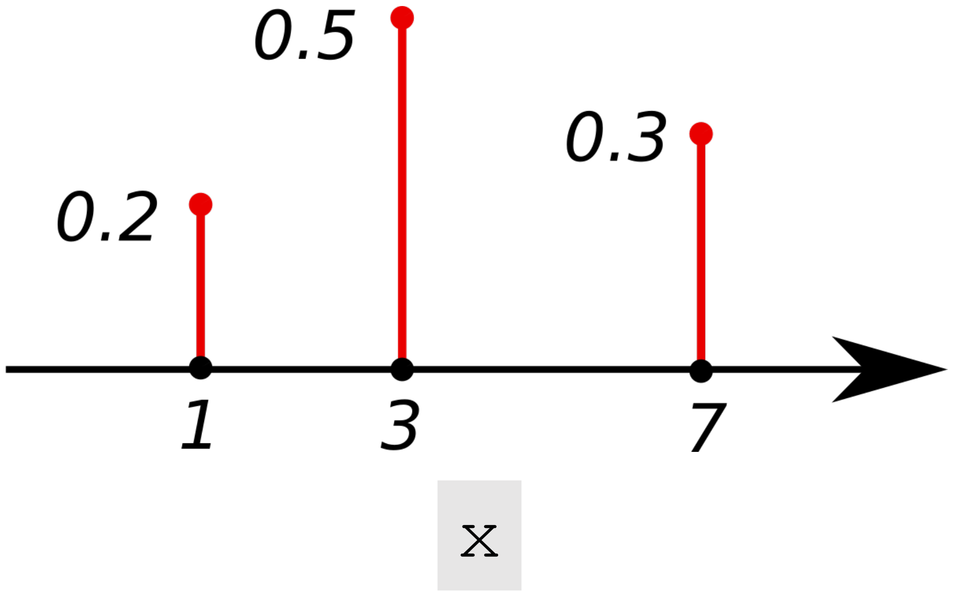
Probability Density Function (PDF)

- PDF provides a relative likelihood that the value of the random variable would equal that sample.

Example: Discrete distribution

- Discrete random variable: $X \in \{1, 3, 7\}$.
- PDF:

$$p(1) = 0.2,$$
$$p(3) = 0.5,$$
$$p(7) = 0.3.$$



Probability Density Function (PDF)

- Random variable X is in the domain $\mathcal{X}$.
- For continuous distribution,

$$\int_{\mathcal{X}} p(x)\, dx = 1.$$

- For discrete distribution,

$$\sum_{x \in \mathcal{X}} p(x) = 1.$$

Expectation

- Random variable X is in the domain $\mathcal{X}$.
- For continuous distribution, the expectation of $f(X)$ is:

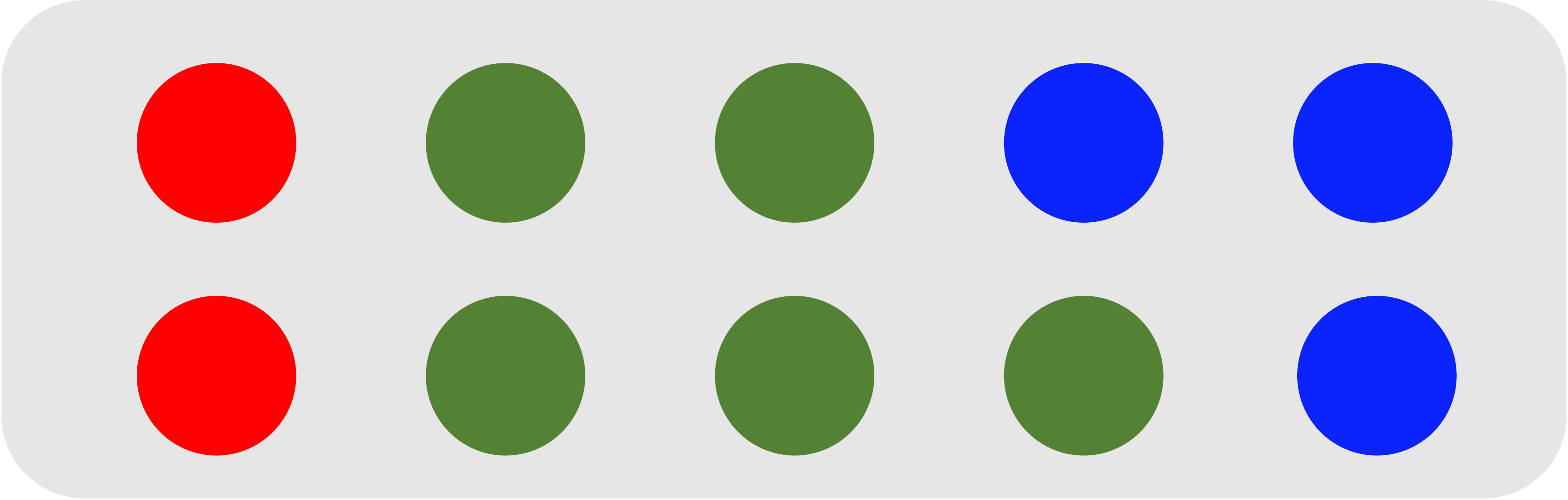
$$\mathbb{E}\left[f(X)\right] = \int_{\mathcal{X}} p(x) \cdot f(x)\, dx.$$

- For discrete distribution, the expectation of $f(X)$ is:

$$\mathbb{E}\left[f(X)\right] = \sum_{x \in \mathcal{X}} p(x) \cdot f(x)\,.$$

Random Sampling

- There are 10 balls in the bin: 2 are red, 5 are green, and 3 are blue.
- Randomly sample a ball.
- What will be the outcome?

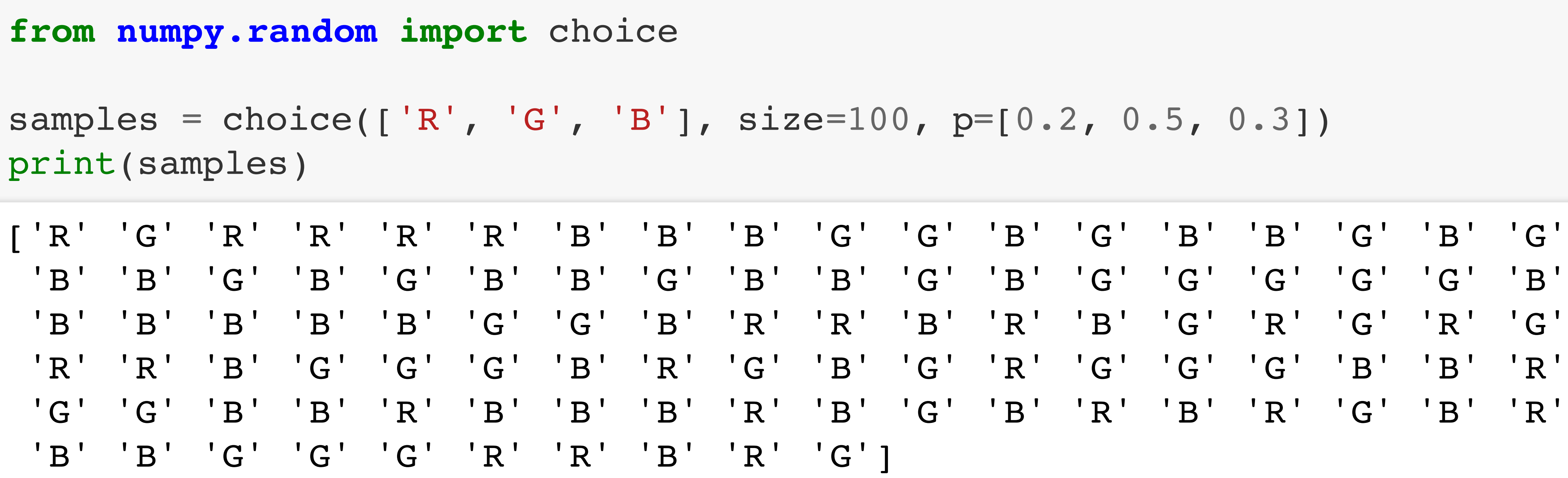


Random Sampling

- Sample red ball w.p. 0.2, green ball w.p. 0.5, and blue ball w.p. 0.3.
- Randomly sample a ball.
- What will be the outcome?

Random Sampling

- Sample red ball w.p. 0.2, green ball w.p. 0.5, and blue ball w.p. 0.3.
- Randomly sample a ball.
- What will be the outcome?



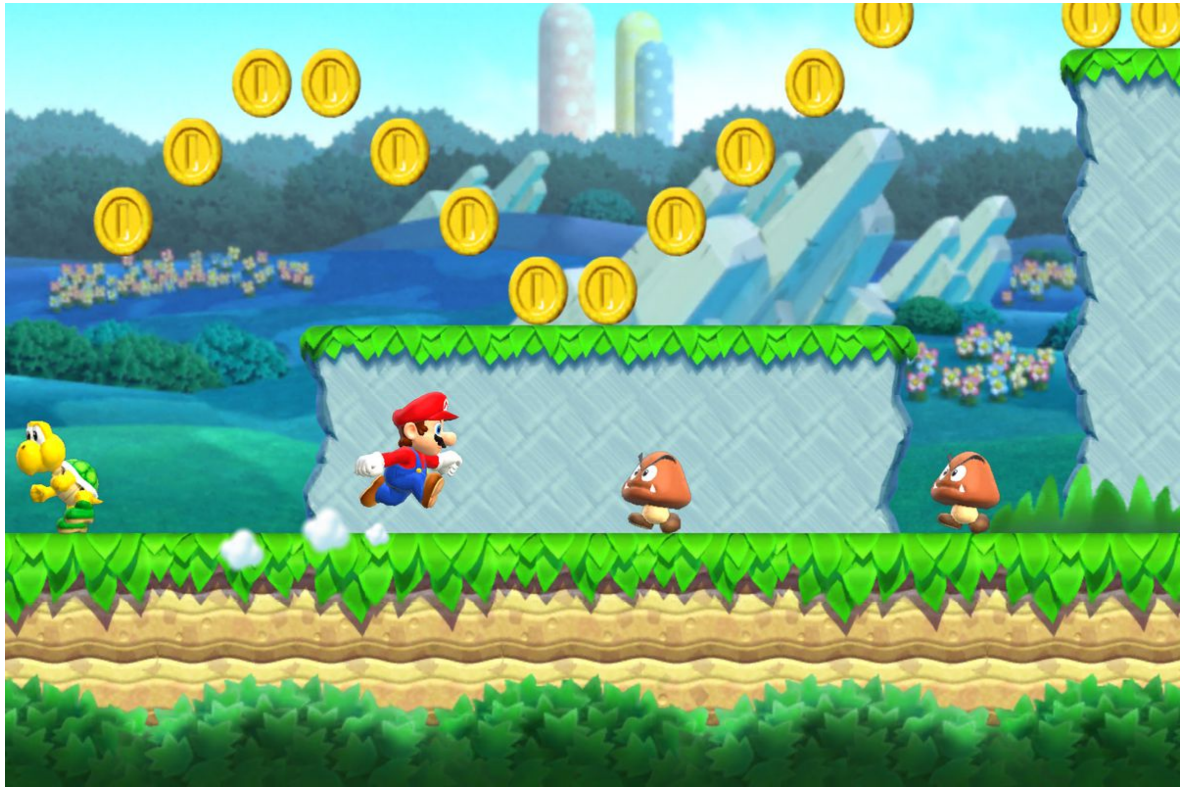
```python
from numpy.random import choice

samples = choice(['R', 'G', 'B'], size=100, p=[0.2, 0.5, 0.3])
print(samples)
```

```
['R' 'G' 'R' 'R' 'R' 'R' 'B' 'B' 'B' 'G' 'G' 'B' 'G' 'B' 'B' 'G' 'B' 'G'
 'B' 'B' 'G' 'B' 'G' 'B' 'B' 'G' 'B' 'B' 'G' 'B' 'G' 'G' 'G' 'G' 'G' 'B'
 'B' 'B' 'B' 'B' 'B' 'G' 'G' 'B' 'R' 'R' 'B' 'R' 'B' 'G' 'R' 'G' 'R' 'G'
 'R' 'R' 'B' 'G' 'G' 'G' 'B' 'R' 'G' 'B' 'G' 'R' 'G' 'G' 'G' 'B' 'B' 'R'
 'G' 'G' 'B' 'B' 'R' 'B' 'B' 'B' 'R' 'B' 'G' 'B' 'R' 'B' 'R' 'G' 'B' 'R'
 'B' 'B' 'G' 'G' 'G' 'R' 'R' 'B' 'R' 'G']
```

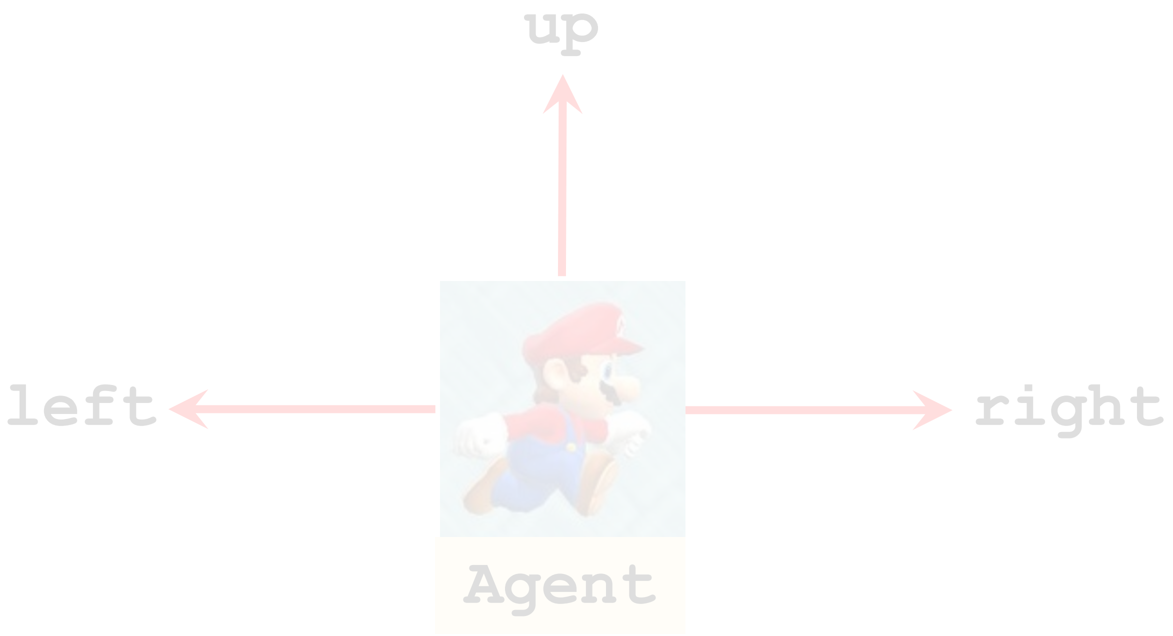
Terminologies

Terminology: state and action

state s (this frame)



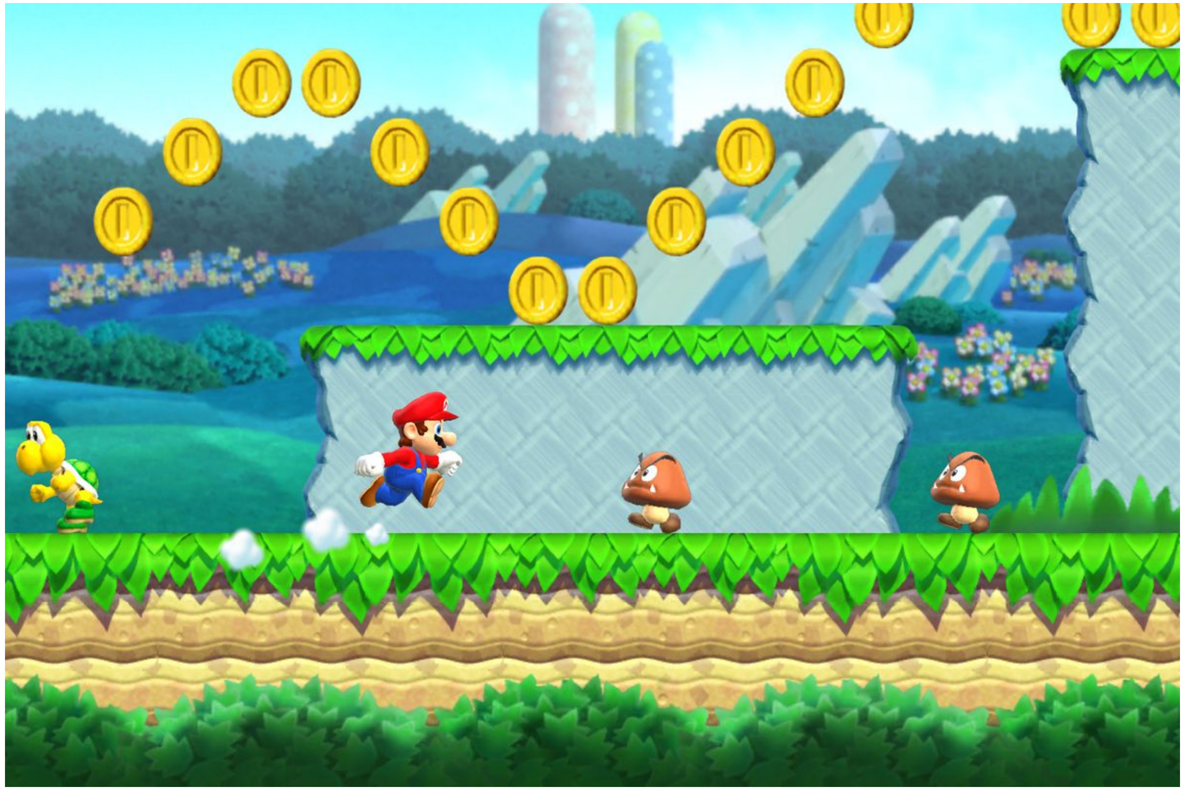
Action $a \in$ {left, right, up}

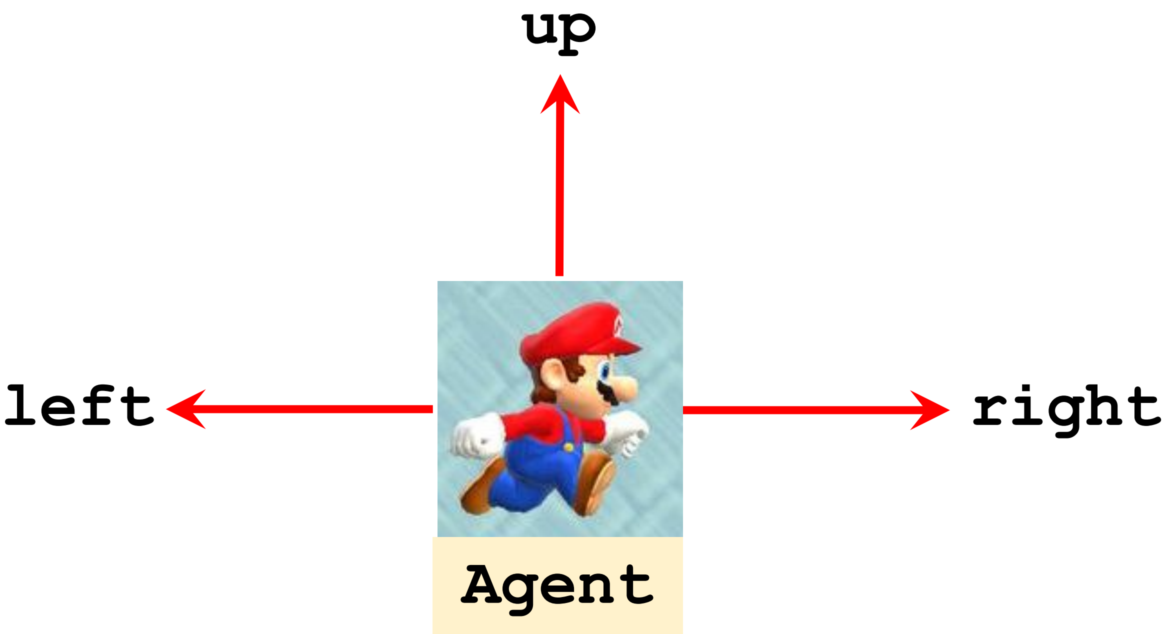


Terminology: state and action

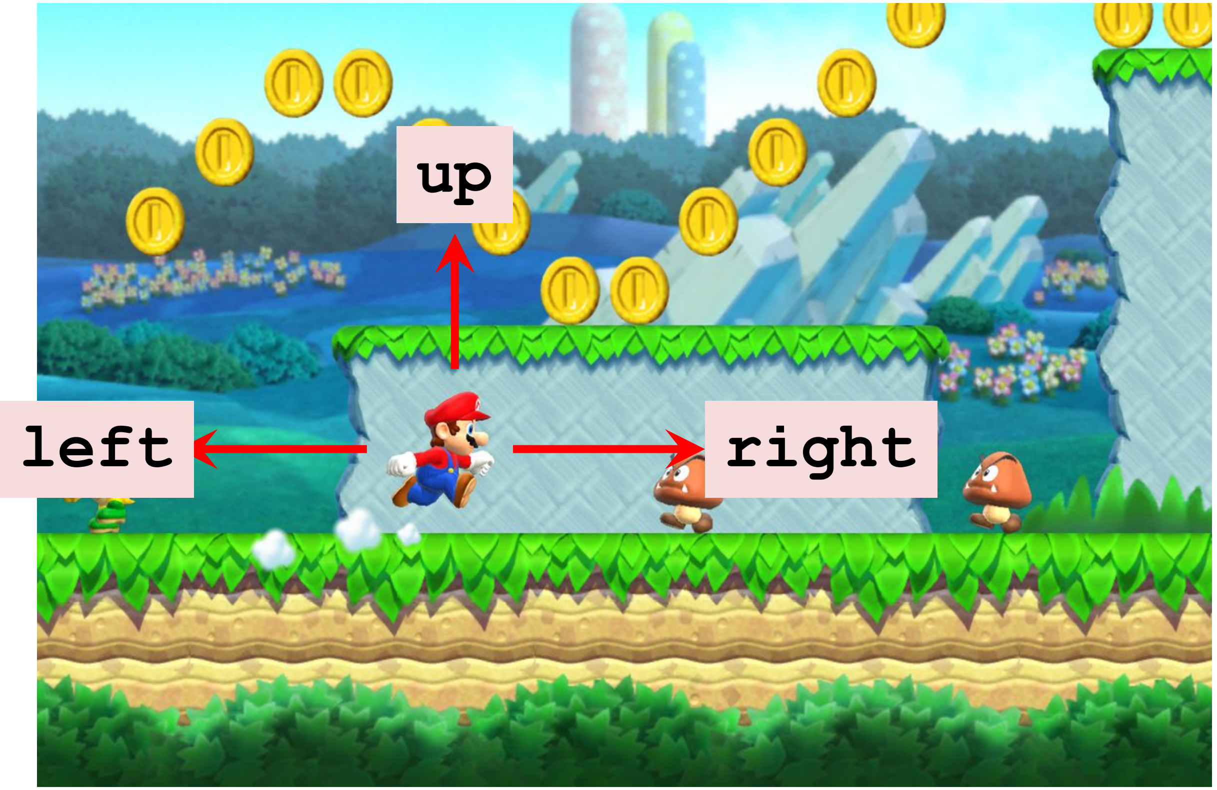
state s (this frame)

Action $a \in \{\text{left}, \text{right}, \text{up}\}$





Terminology: policy



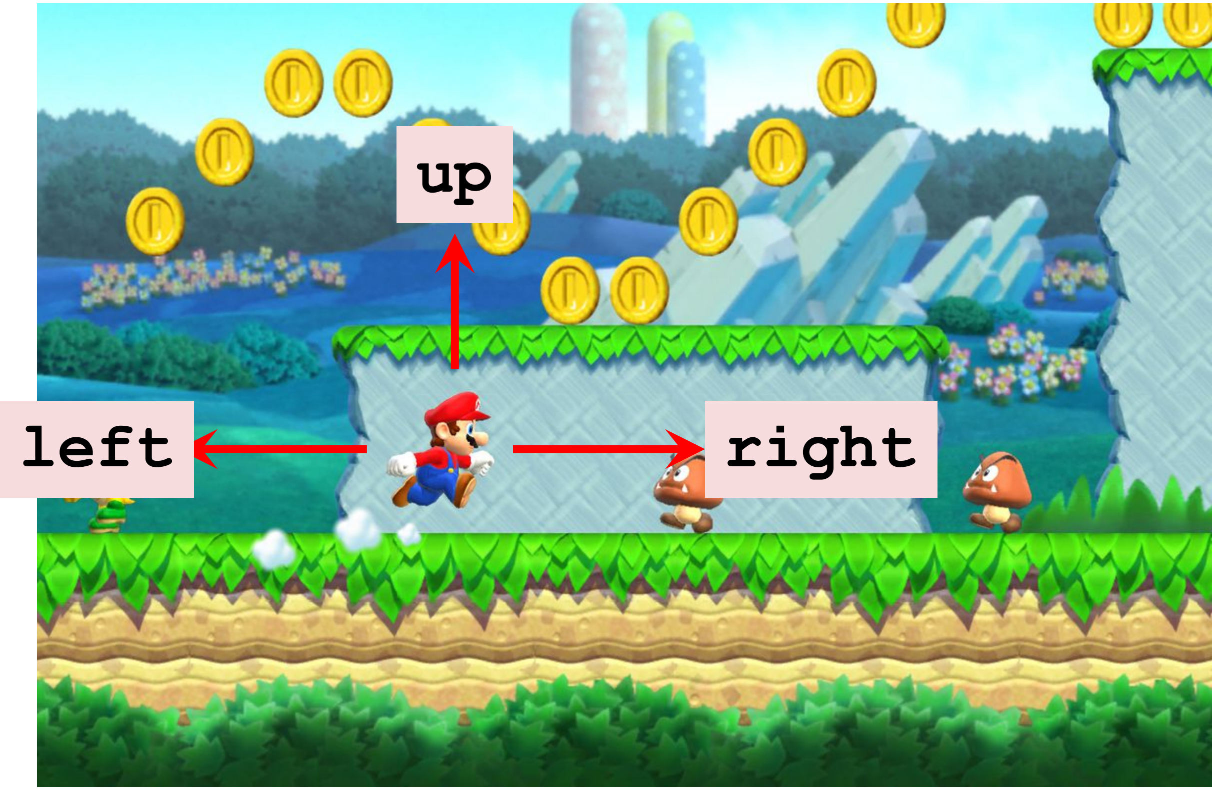
policy π

- Policy function $\pi: (s, a) \mapsto [0,1]$:

$$\pi(a \mid s) = \mathbb{P}(A = a \mid S = s).$$

- It is the probability of taking action $A = a$ given state s, e.g.,
  - $\pi(\text{left} \mid s) = 0.2$,
  - $\pi(\text{right} \mid s) = 0.1$,
  - $\pi(\text{up} \mid s) = 0.7$.
- Upon observing state $S = s$, the agent's action A can be random.

Terminology: policy



policy π

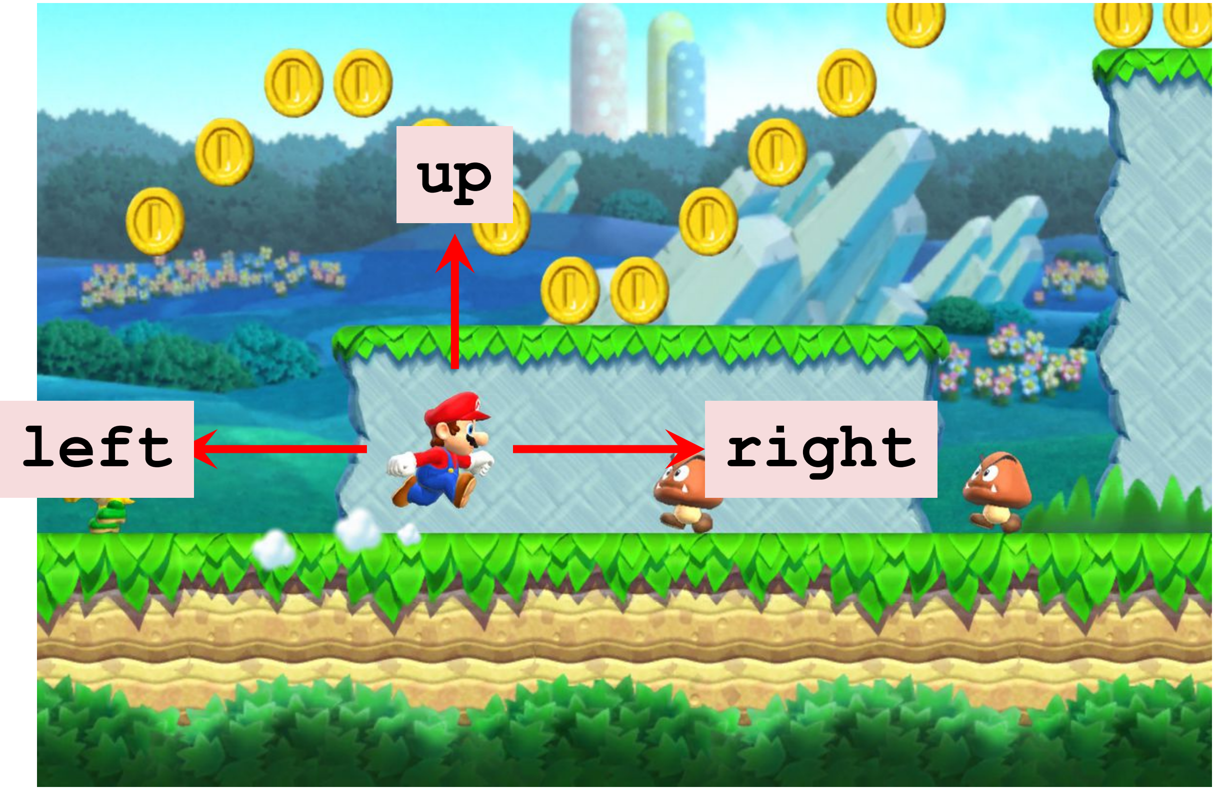
- Policy function π: $(s, a) \mapsto [0,1]$:

$$\pi(a \mid s) = \mathbb{P}(A = a \mid S = s).$$

- It is the probability of taking action $A = a$ given state s, e.g.,
  - $\pi(\text{left} \mid s) = 0.2$,
  - $\pi(\text{right} \mid s) = 0.1$,
  - $\pi(\text{up} \mid s) = 0.7$.

- Upon observing state $S = s$, the agent's action A can be random.

Terminology: policy



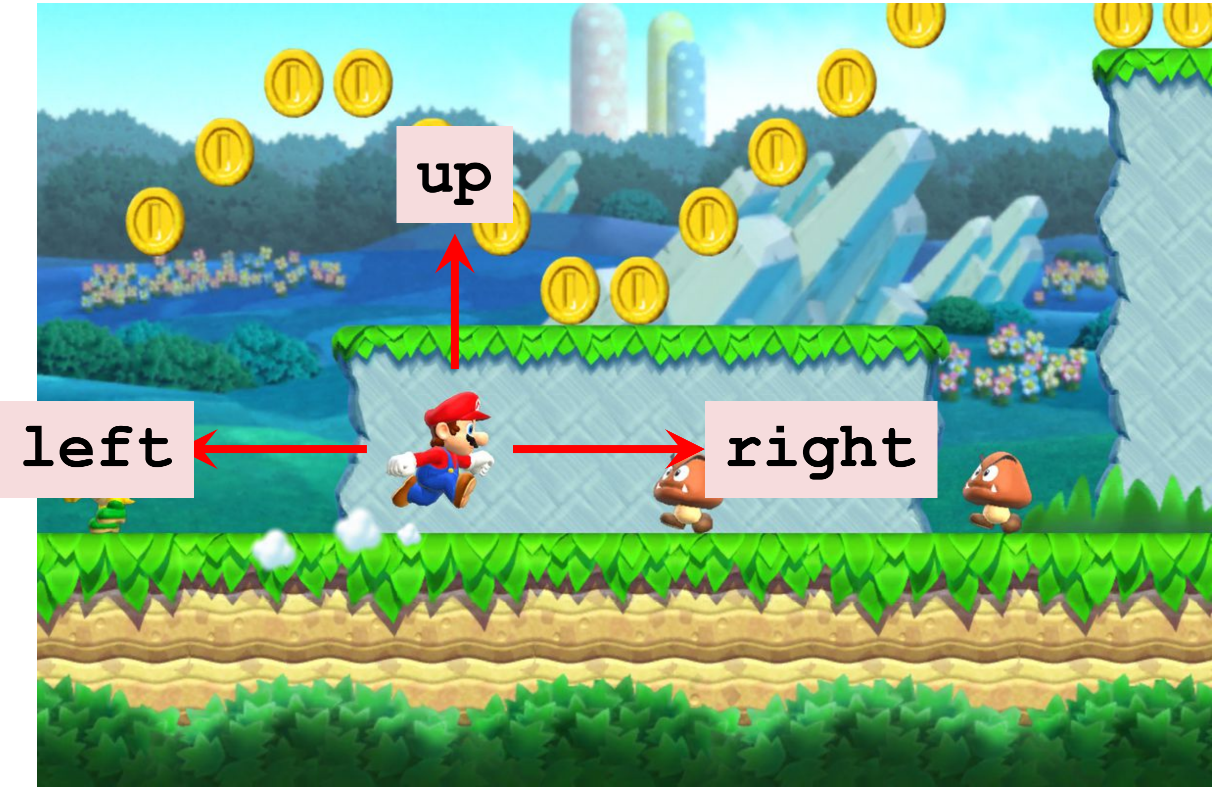
policy π

- Policy function $\pi: (s, a) \mapsto [0,1]$:

$$\pi(a \mid s) \; = \; \mathbb{P}(A = a \mid S = s).$$

- It is the probability of taking action $A = a$ given state s , e.g.,
  - $\pi(\text{left} \mid s) = 0.2,$
  - $\pi(\text{right} \mid s) = 0.1,$
  - $\pi(\text{up} \mid s) = 0.7.$
- Upon observing state $S = s$, the agent's action A can be random.

Terminology: policy



policy π

- Policy function $\pi: (s, a) \mapsto [0,1]$:

$$\pi(a \mid s) \;=\; \mathbb{P}(A = a \mid S = s).$$

- It is the probability of taking action $A = a$ given state s , e.g.,
  - $\pi(\text{left} \mid s) = 0.2,$
  - $\pi(\text{right} \mid s) = 0.1,$
  - $\pi(\text{up} \mid s) = 0.7.$
- Upon observing state $S = s$, the agent's action A can be random.

Terminology: reward

reward R

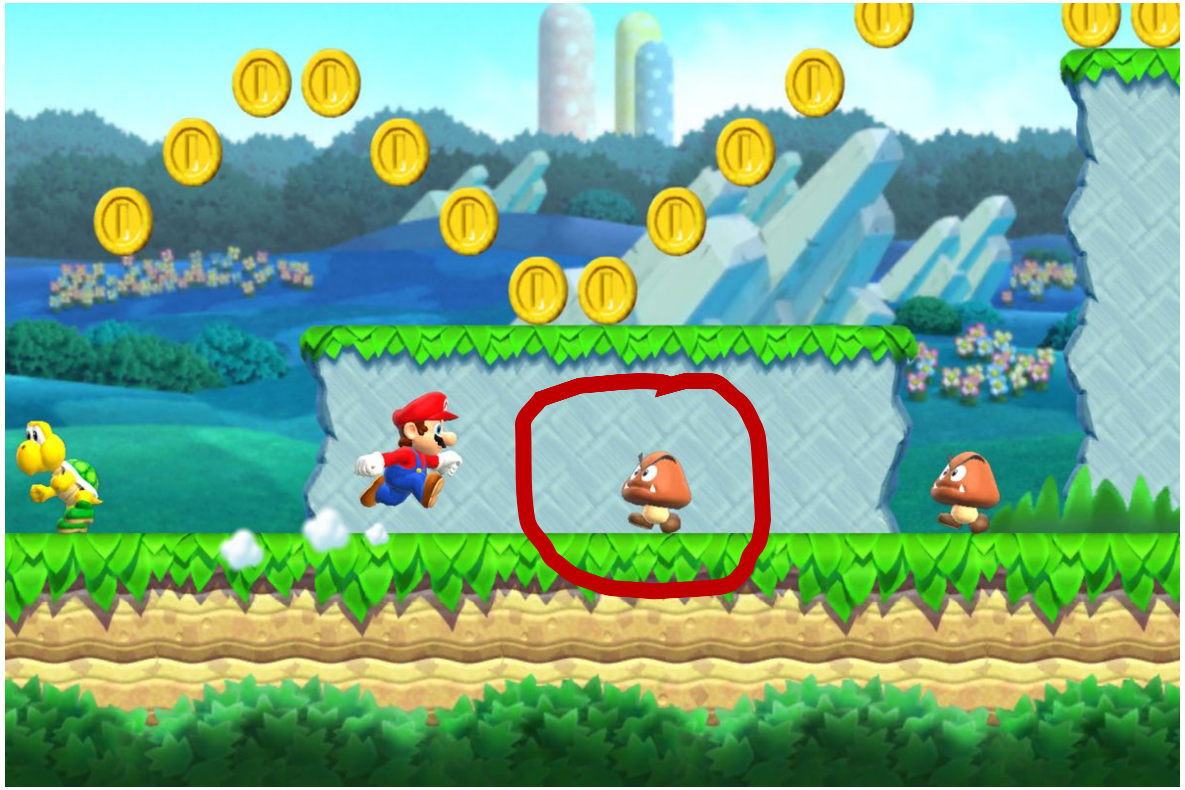
- Collect a coin: $R = +1$

Terminology: reward

reward R

- Collect a coin: $R = +1$
- Win the game: $R = +10000$

Terminology: reward



reward R

- Collect a coin: $R = +1$
- Win the game: $R = +10000$
- Touch a Goomba: $R = -10000$ (game over).

Terminology: reward

reward R

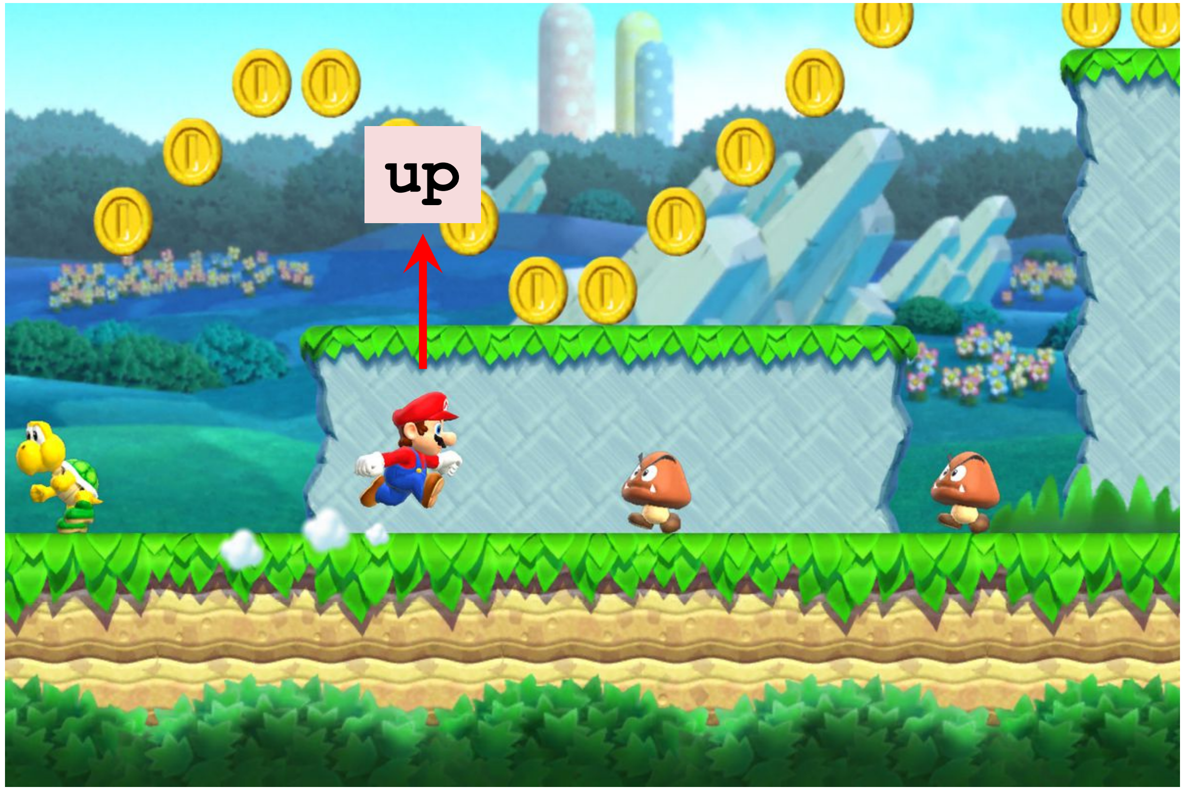
- Collect a coin: $R = +1$
- Win the game: $R = +10000$
- Touch a Goomba: $R = -10000$ (game over).
- Nothing happens: $R = 0$

Terminology: state transition

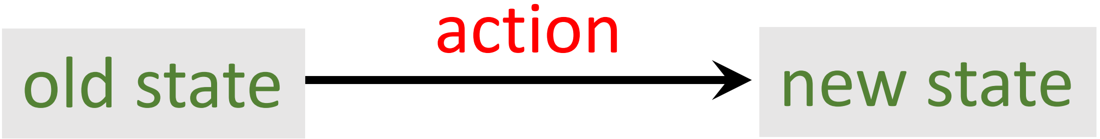
state transition

$$\text{old state} \xrightarrow{\text{action}} \text{new state}$$

Terminology: state transition

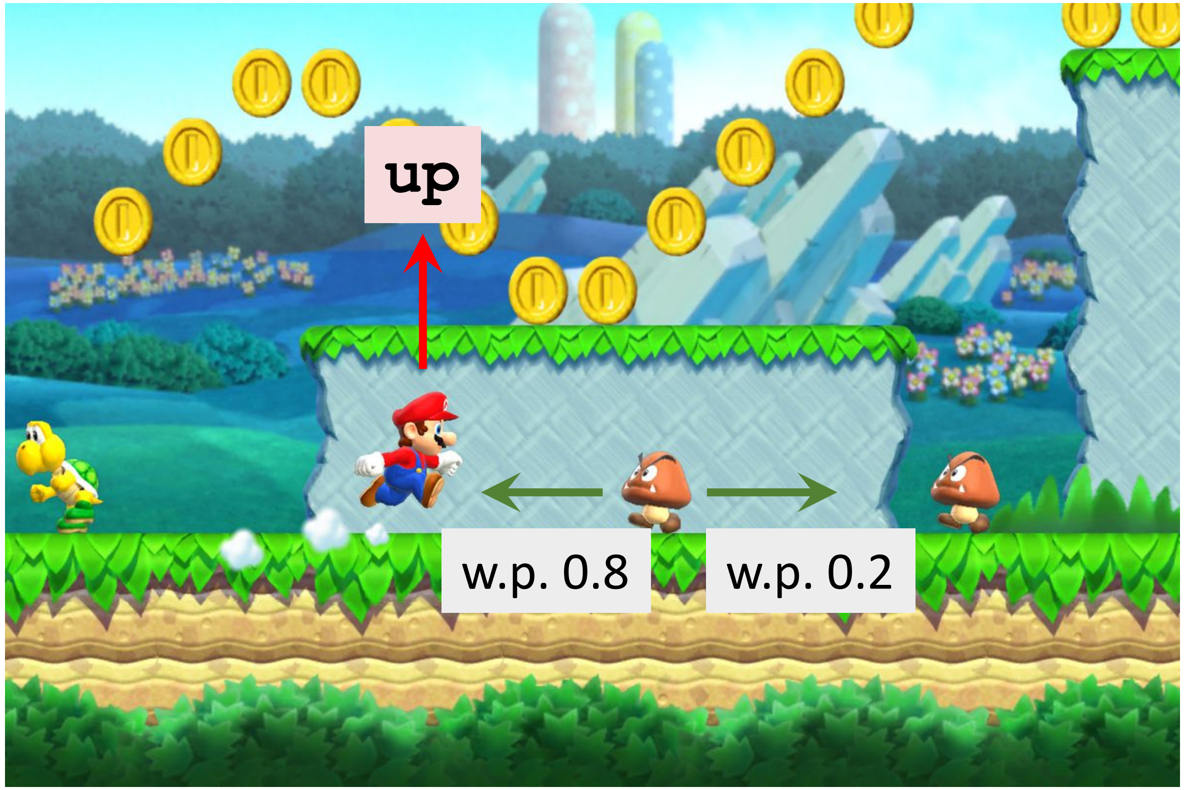


state transition



- E.g., “up” action leads to a new state.

Terminology: state transition



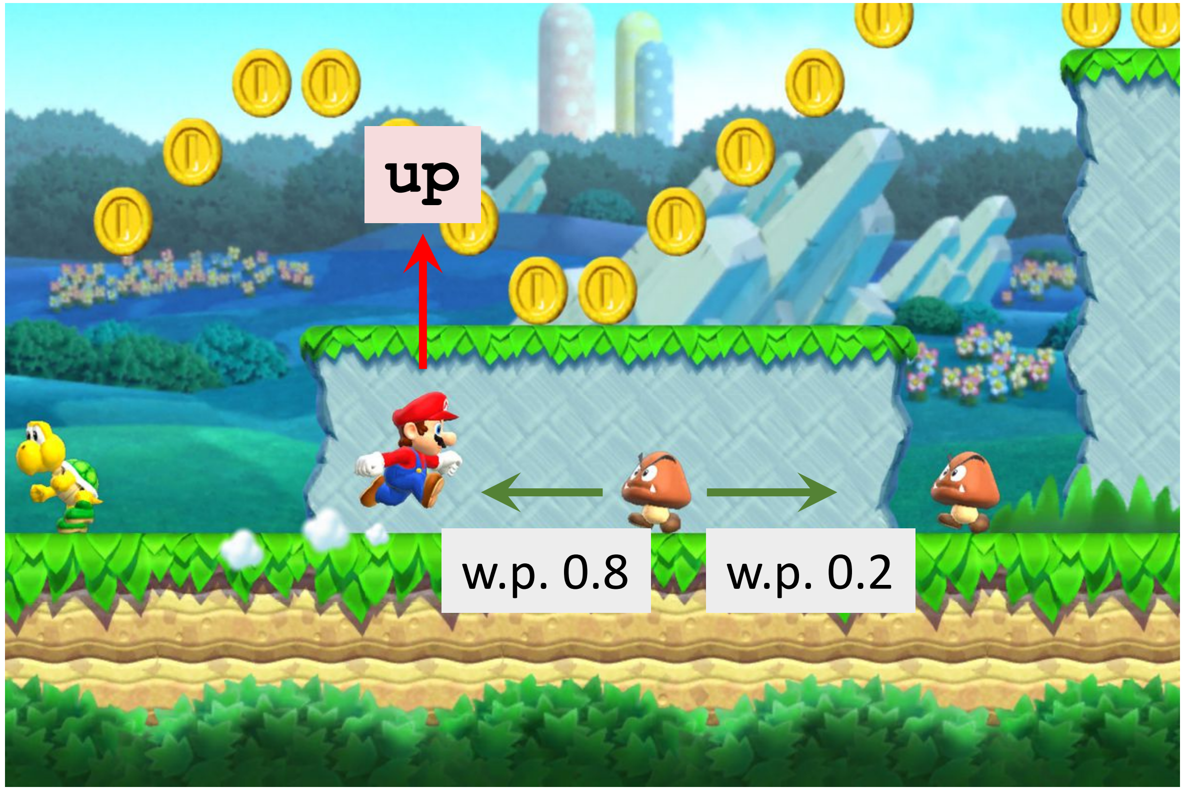
state transition

$$\text{old state} \xrightarrow{\text{action}} \text{new state}$$

- E.g., "up" action leads to a new state.

- State transition can be random.
- Randomness is from the environment.

Terminology: state transition



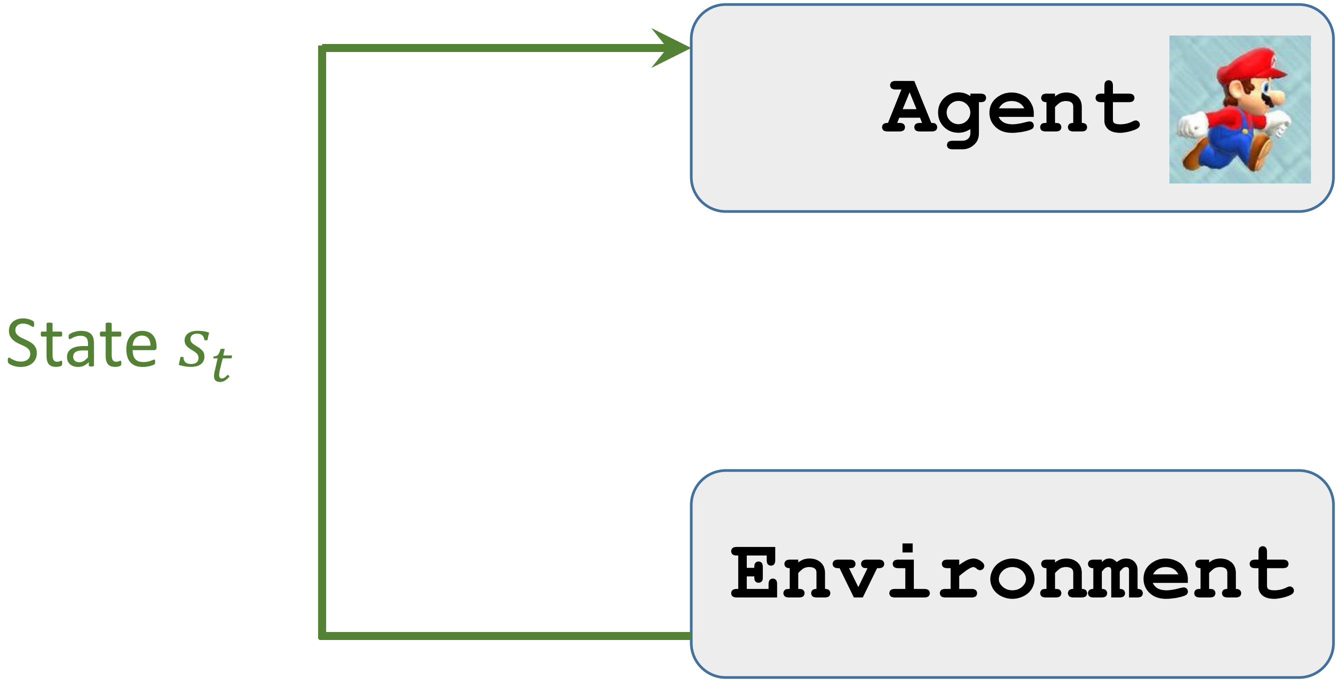
state transition

$$\text{old state} \xrightarrow{\text{action}} \text{new state}$$

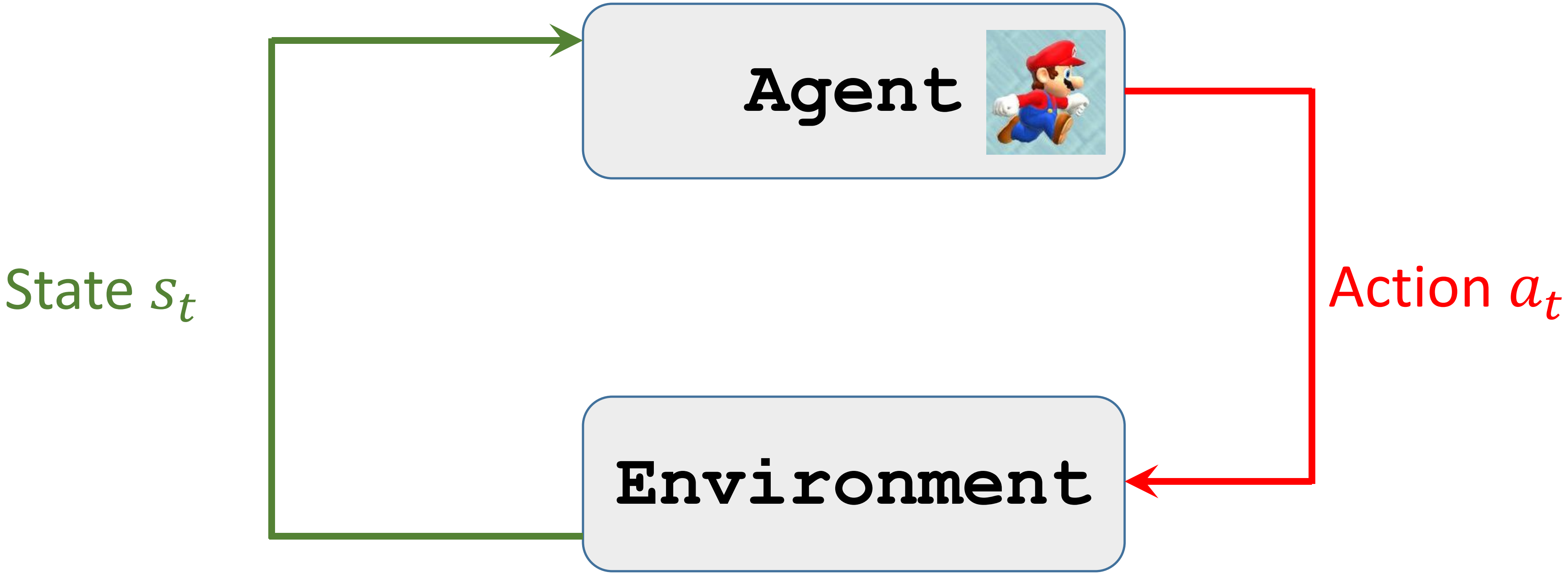
- E.g., “up” action leads to a new state.

- State transition can be random.
- Randomness is from the environment.
- $p(s'|s,a) = \mathbb{P}(S' = s' | S = s, A = a)$.

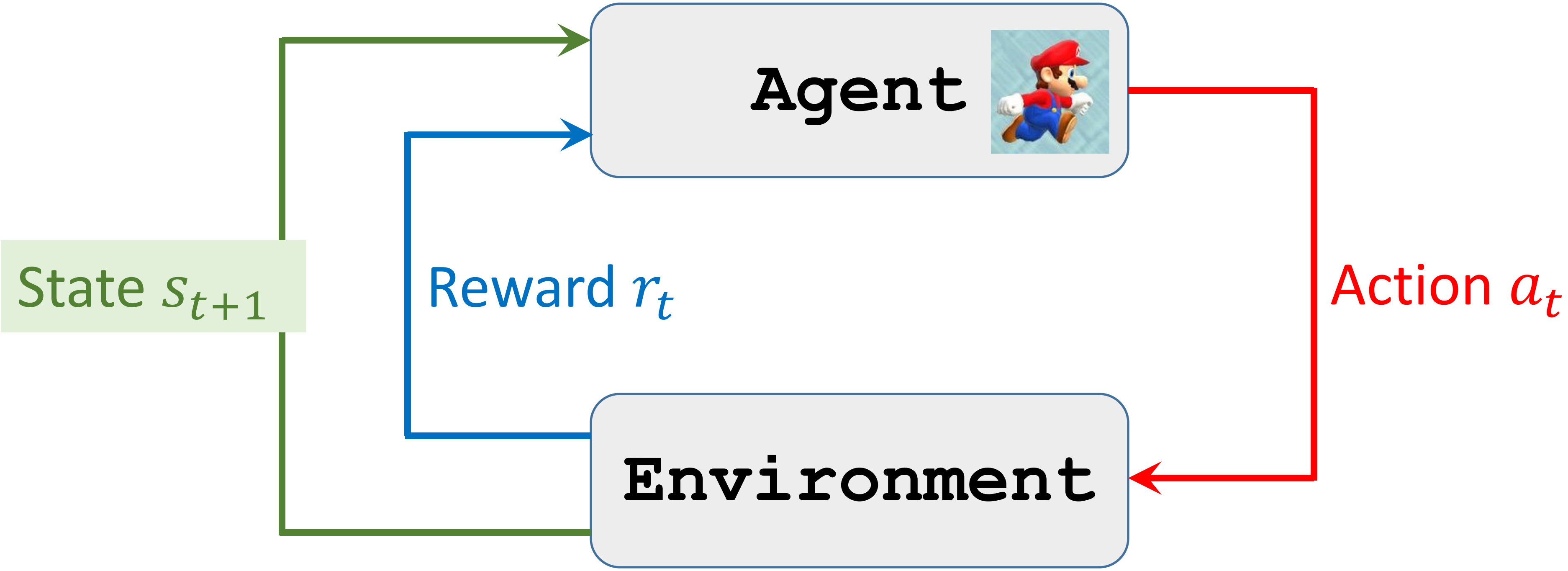
Terminology: agent environment interaction



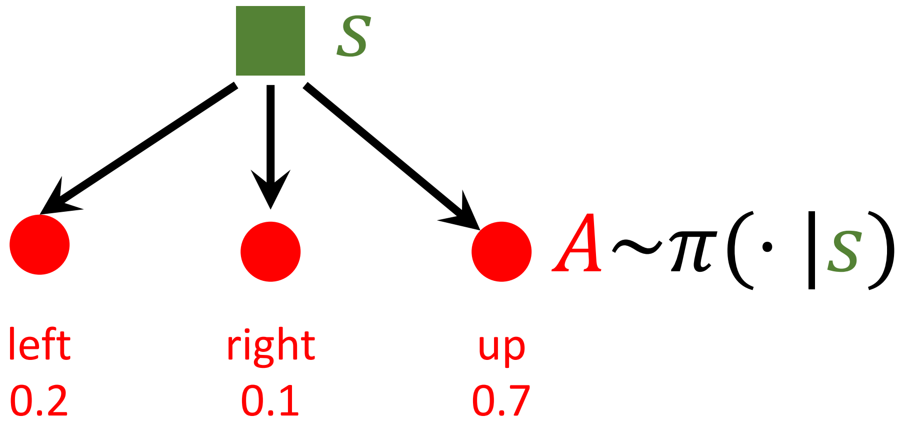
Terminology: agent environment interaction



Terminology: agent environment interaction



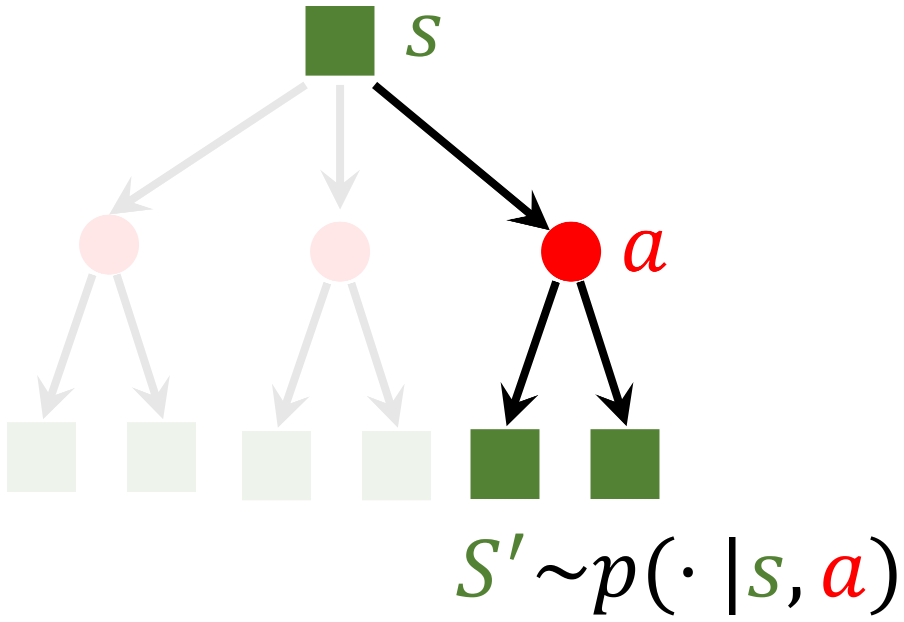
Randomness in Reinforcement Learning



Actions have randomness.

- Given state s, the action can be random, e.g., .
  - $\pi(\text{“left”} \mid s) = 0.2,$
  - $\pi(\text{“right”} \mid s) = 0.1,$
  - $\pi(\text{“up”} \mid s) = 0.7.$

Randomness in Reinforcement Learning



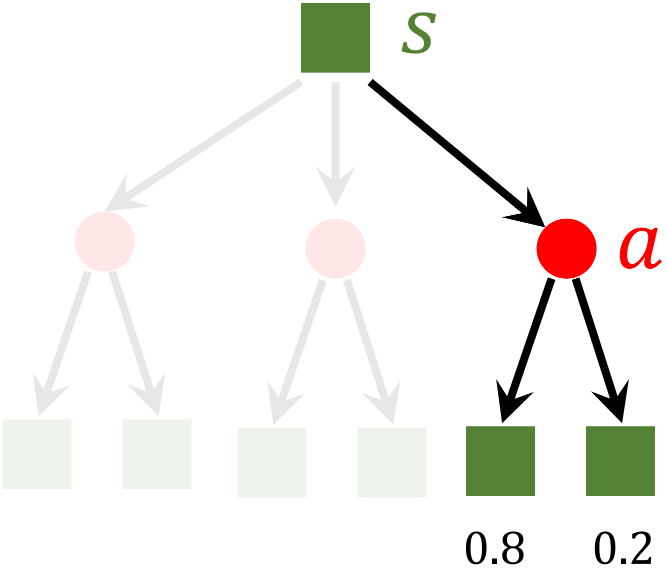
Actions have randomness.

- Given state s, the action can be random, e.g., .
  - $\pi(\text{“left”} \mid s) = 0.2$,
  - $\pi(\text{“right”} \mid s) = 0.1$,
  - $\pi(\text{“up”} \mid s) = 0.7$.

State transitions have randomness.

- Given state $S = s$ and action $A = a$, the environment randomly generates a new state S'.

Randomness in Reinforcement Learning



Actions have randomness.

- Given state s, the action can be random, e.g., .
  - $\pi(\text{“left”}|s) = 0.2$,
  - $\pi(\text{“right”}|s) = 0.1$,
  - $\pi(\text{“up”}|s) = 0.7$.

State transitions have randomness.

- Given state $S = s$ and action $A = a$, the environment randomly generates a new state S'.

Play the game using AI

- Observe a frame (state s_1)
- ➔ Make action a_1 (left, right, or up)
- ➔ Observe a new frame (state s_2) and reward r_1
- ➔ Make action a_2
- ➔ …

Play the game using AI

- Observe a frame (state s_1)
- ➔ Make action a_1 (left, right, or up)
- ➔ Observe a new frame (state s_2) and reward r_1
- ➔ Make action a_2
- ➔ …

- (state, action, reward) trajectory:

$$s_1, a_1, r_1, s_2, a_2, r_2, \quad \cdots \quad , s_T, a_T, r_T.$$

Rewards and Returns

Return

Definition: Return (aka cumulative future reward).

- $U_t = R_t + R_{t+1} + R_{t+2} + R_{t+3} + \cdots$

Return

Definition: Return (aka cumulative future reward).

- $U_t = R_t + R_{t+1} + R_{t+2} + R_{t+3} + \cdots$

Question: Are R_t and R_{t+1} equally important?

- Which of the followings do you prefer?
  - I give you \$100 right now.
  - I will give you \$100 one year later.

Return

Definition: Return (aka cumulative future reward).

- $U_t = R_t + R_{t+1} + R_{t+2} + R_{t+3} + \cdots$

Question: Are R_t and R_{t+1} equally important?

- Which of the followings do you prefer?
  - I give you $100 right now.
  - I will give you $100 one year later.

- Future reward is less valuable than present reward.
- R_{t+1} should be given less weight than R_t.

Return

Definition: Return (aka cumulative future reward).

- $U_t = R_t + R_{t+1} + R_{t+2} + R_{t+3} + \cdots$

Definition: Discounted return (aka cumulative discounted future reward).

- γ: discount rate (tuning hyper-parameter).
- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Randomness in Returns

Definition: Discounted return (at time step t).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

At time step t, the return U_t is random.

- Two sources of randomness:
  1. Action can be random: $\mathbb{P}[A = a \mid S = s] = \pi(a|s)$.
  2. New state can be random: $\mathbb{P}[S' = s' | S = s, A = a] = p(s'|s, a)$.

Randomness in Returns

Definition: Discounted return (at time step t).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

At time step t, the return U_t is random.

- Two sources of randomness:
  1. Action can be random: $\mathbb{P}[A = a \mid S = s] = \pi(a|s)$.
  2. New state can be random: $\mathbb{P}[S' = s' | S = s, A = a] = p(s'|s, a)$.
- For any $i \geq t$, the reward R_i depends on S_i and A_i.
- Thus, given s_t, the return U_t depends on the random variables:
  - $A_t, A_{t+1}, A_{t+2}, \cdots$ and $S_{t+1}, S_{t+2}, \cdots$.

Value Functions

Action-Value Function $Q(s, a)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Action-Value Function $Q(s, a)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Definition: Action-value function for policy π.

- $Q_\pi(s_t, a_t) = \mathbb{E}\left[U_t | S_t = s_t, A_t = a_t\right].$

- Return U_t depends on states $S_t, S_{t+1}, S_{t+2}, \cdots$ and actions $A_t, A_{t+1}, A_{t+2}, \cdots$.

Action-Value Function $Q(s, a)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Definition: Action-value function for policy π.

- $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t]$.

- Return U_t depends on states $S_t, S_{t+1}, S_{t+2}, \cdots$ and actions $A_t, A_{t+1}, A_{t+2}, \cdots$.
- Actions are random: $\mathbb{P}[A = a \mid S = s] = \pi(a|s)$. (Policy function.)
- States are random: $\mathbb{P}[S' = s' | S = s, A = a] = p(s'|s, a)$. (State transition.)

Action-Value Function $Q(s, a)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Definition: Action-value function for policy π.

- $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t]$.

Definition: Optimal action-value function.

- $Q^\star(s_t, a_t) = \max_\pi Q_\pi(s_t, a_t)$.

State-Value Function $V(s)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Definition: Action-value function for policy π.

- $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t]$.

Definition: State-value function.

- $V_\pi(s_t) = \mathbb{E}_A\,[Q_\pi(s_t, A)]$

State-Value Function $V(s)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Definition: Action-value function for policy π.

- $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t]$.

Definition: State-value function.

- $V_\pi(s_t) = \mathbb{E}_A\,[Q_\pi(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_\pi(s_t, a)$. (Actions are discrete.)

Taken w.r.t. the action $A \sim \pi(\cdot\,|s_t)$.

State-Value Function $V(s)$

Definition: Discounted return (aka cumulative discounted future reward).

- $U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \gamma^3\, R_{t+3} + \cdots$

Definition: Action-value function for policy π.

- $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t].$

Definition: State-value function.

- $V_\pi(s_t) = \mathbb{E}_A\,[Q_\pi(s_t, A)] = \sum_a \pi(a|s_t) \cdot Q_\pi(s_t, a).$ (Actions are discrete.)
- $V_\pi(s_t) = \mathbb{E}_A\,[Q_\pi(s_t, A)] = \int \pi(a|s_t) \cdot Q_\pi(s_t, a)\, da\,.$ (Actions are continuous.)

Understanding the Value Functions

- Action-value function: $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t]$.
- Given policy π, $Q_\pi(s, a)$ evaluates how good it is for an agent to pick action a while being in state s.

Understanding the Value Functions

- Action-value function: $Q_\pi(s_t, a_t) = \mathbb{E}\left[U_t | S_t = s_t, A_t = a_t\right]$.
- Given policy π, $Q_\pi(s, a)$ evaluates how good it is for an agent to pick action a while being in state s.

- State-value function: $V_\pi(s) = \mathbb{E}_A\left[Q_\pi(s, A)\right]$
- For fixed policy π, $V_\pi(s)$ evaluates how good the situation is in state s.
- $\mathbb{E}_S[V_\pi(S)]$ evaluates how good the policy π is.

Play games using reinforcement learning

How does AI control the agent?

Suppose we have a good policy $\pi(a|s)$.

- Upon observing the state s_t,
- random sampling: $a_t \sim \pi(\cdot\,|s_t)$.

How does AI control the agent?

Suppose we have a good policy $\pi(a|s)$.

- Upon observing the state s_t,
- random sampling: $a_t \sim \pi(\cdot\,|s_t)$.

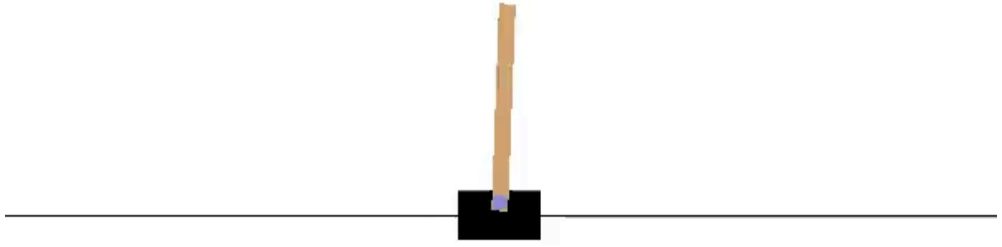
Suppose we know the optimal action-value function $Q^{\star}(s,a)$.

- Upon observe the state s_t,
- choose the action that maximizes the value: $a_t = \mathrm{argmax}_a\, Q^{\star}(s_t,a)$.

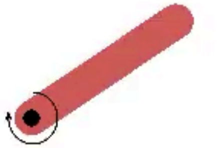
OpenAI Gym

- Gym is a toolkit for developing and comparing reinforcement learning algorithms.
- https://gym.openai.com/

Classical control problems



Cart Pole

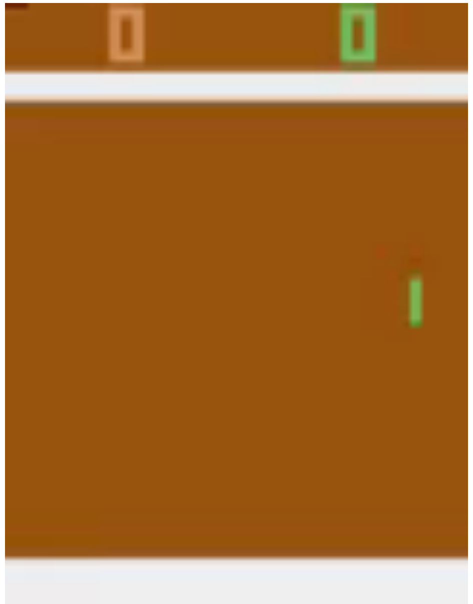


Pendulum

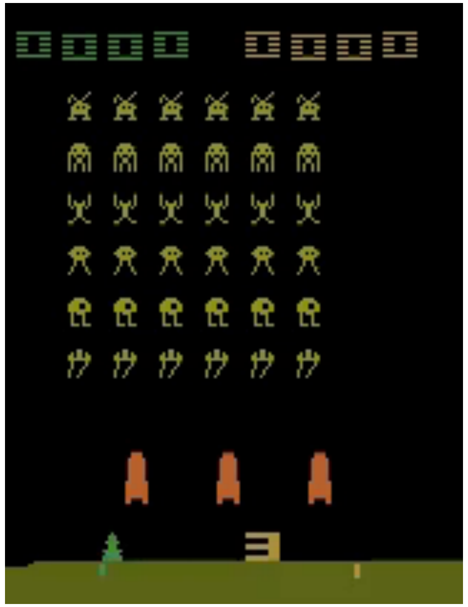
OpenAI Gym

- Gym is a toolkit for developing and comparing reinforcement learning algorithms.
- https://gym.openai.com/

Atari Games



Pong



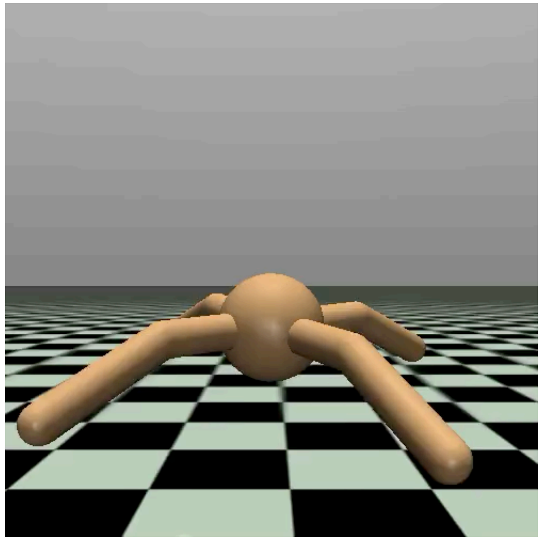
Space Invader

Breakout

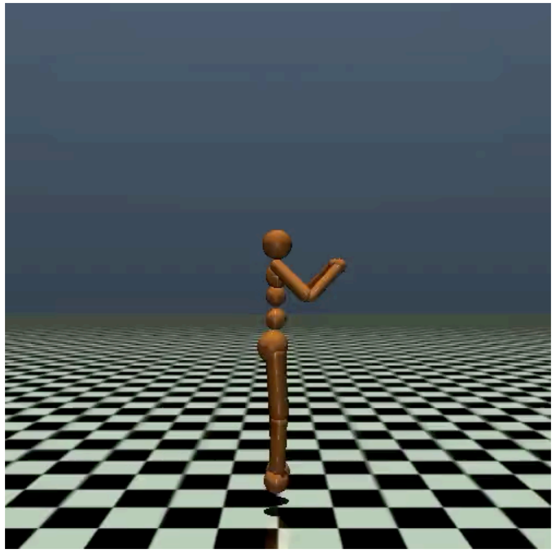
OpenAI Gym

- Gym is a toolkit for developing and comparing reinforcement learning algorithms.
- https://gym.openai.com/

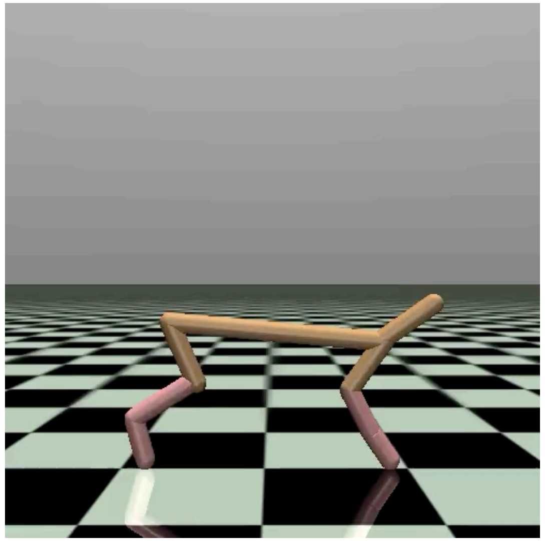
MuJoCo (Continuous control tasks.)



Ant

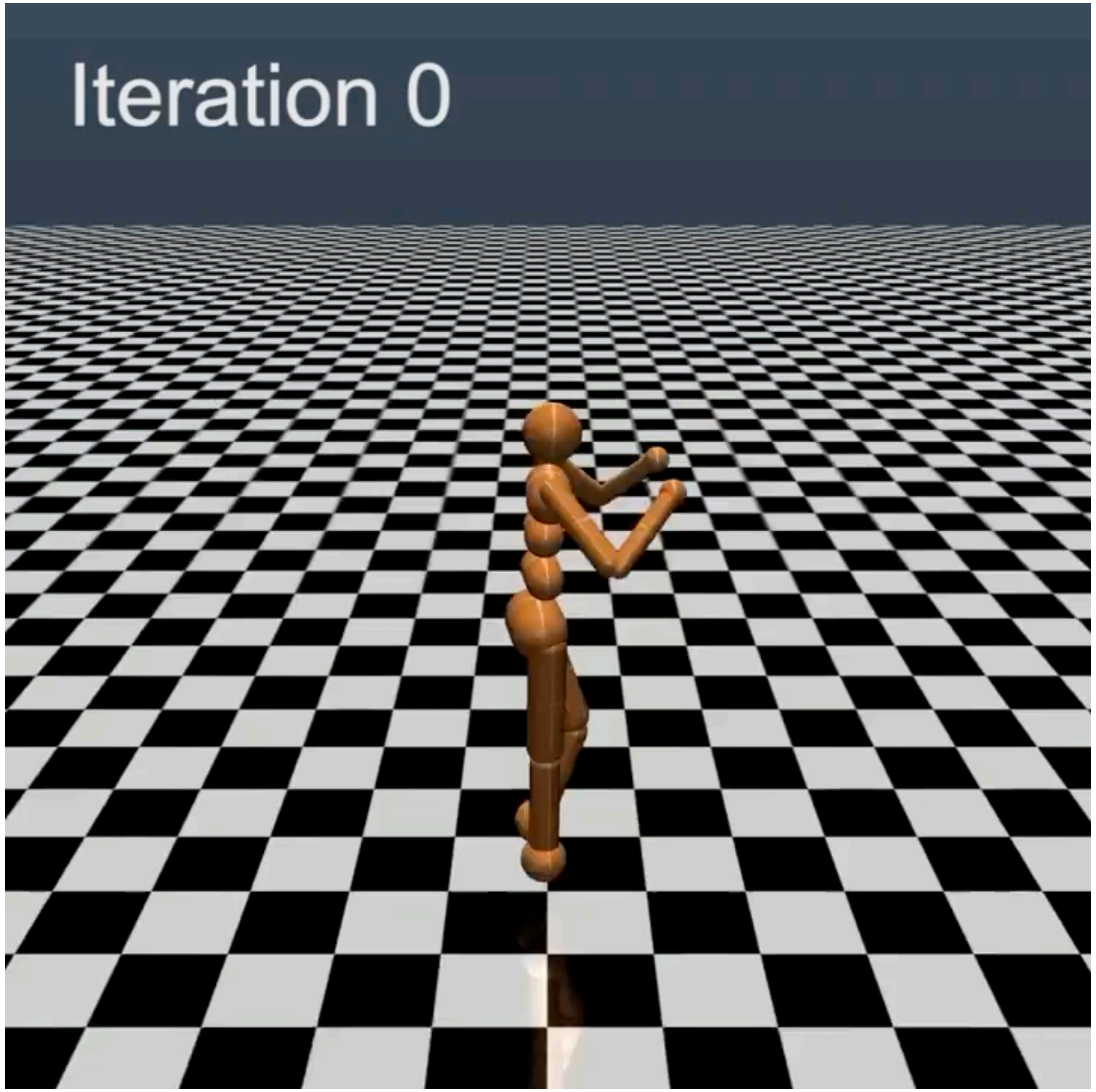


Humanoid



Half Cheetah

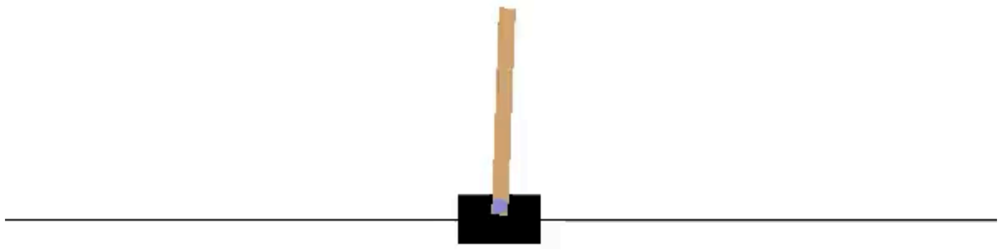
OpenAI Gym



Play CartPole Game

```python
import gym
env = gym.make('CartPole-v0')
```

- Get the environment of CartPole from Gym.
- “env” provides states and reward.



Play CartPole Game

```
state = env.reset()

for t in range(100):
    env.render()
    print(state)

    action = env.action_space.sample()
    state, reward, done, info = env.step(action)

    if done:
        print('Finished')
        break

env.close()
```

A window pops up rendering CartPole.

A random action.

“done=1” means finished (win or lose the game)

Summary

Summary

Terminologies

- Agent
- Environment
- State s.
- Action a.
- Reward r.
- Policy $\pi(a|s)$
- State transition $p(s'|s,a)$.

Summary

Terminologies

- Agent
- Environment
- State s.
- Action a.
- Reward r.
- Policy $\pi(a|s)$
- State transition $p(s'|s,a)$.

Return and Value

- Return:

$$U_t = R_t + \gamma\, R_{t+1} + \gamma^2\, R_{t+2} + \cdots$$

- Action-value function:

$$Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | s_t, a_t].$$

- Optimal action-value function:

$$Q^\star(s_t, a_t) = \max_\pi Q_\pi(s_t, a_t).$$

- State-value function:

$$V_\pi(s_t) = \mathbb{E}_A[Q_\pi(s_t, A)].$$

Play game using reinforcement learning

- Observe state s_t, make action a_t, environment gives s_{t+1} and reward r_t.

$$\begin{tikzcd}
s_t \arrow[r] & a_t \arrow[r] \arrow[dr] & s_{t+1} \arrow[r] & a_{t+1} \arrow[r] \arrow[dr] & s_{t+2} \arrow[r] & a_{t+2} \arrow[r] \arrow[dr] & s_{t+3} & \cdots \\
 & & r_t & & r_{t+1} & & r_{t+2} &
\end{tikzcd}$$

Play game using reinforcement learning

- Observe state s_t, make action a_t, environment gives s_{t+1} and reward r_t.

$$\begin{tikzcd}
s_t \arrow[r] & a_t \arrow[r] \arrow[dr] & s_{t+1} \arrow[r] & a_{t+1} \arrow[r] \arrow[dr] & s_{t+2} \arrow[r] & a_{t+2} \arrow[r] \arrow[dr] & s_{t+3} & \cdots \\
 & & r_t & & r_{t+1} & & r_{t+2} &
\end{tikzcd}$$

- The agent can be controlled by either $\pi(a|s)$ or $Q^{\star}(s, a)$.

We are going to study…

2. Value-based learning.
    - Deep Q network (DQN) for approximating $Q^{\star}(s, a)$.
    - Learn the network parameters using temporal different (TD).

3. Policy-based learning.
    - Policy network for approximating $\pi(a|s)$.
    - Learn the network parameters using policy gradient.

4. Actor-critic method. (Policy network + value network.)

5. Example: AlphaGo

Thank you!