

Introduction to Machine Learning

Lecture 20: Elementary Reinforcement Learning – Stochastic Environment

Dec 20, 2021

Jie Wang

Machine Intelligence Research and Applications Lab

Department of Electronic Engineering and Information Science (EEIS)

<http://staff.ustc.edu.cn/~jwangx/>

jjewangx@ustc.edu.cn



Machine Intelligence Research and Applications Lab

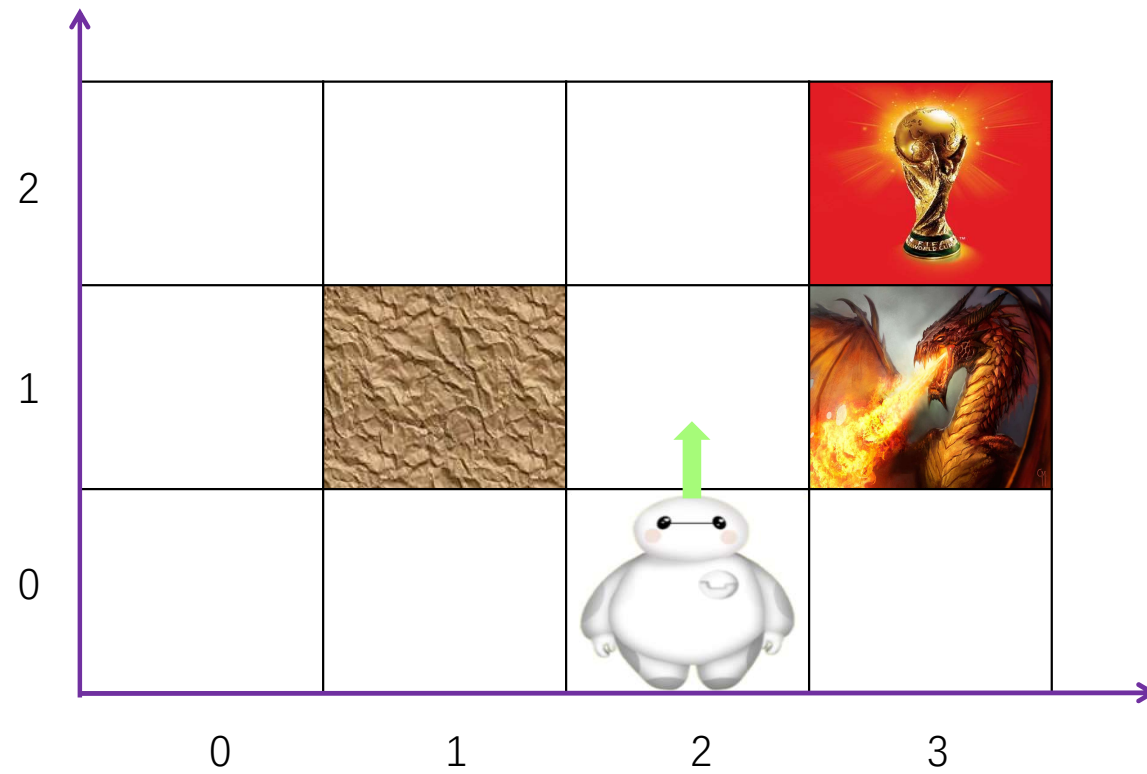


Contents

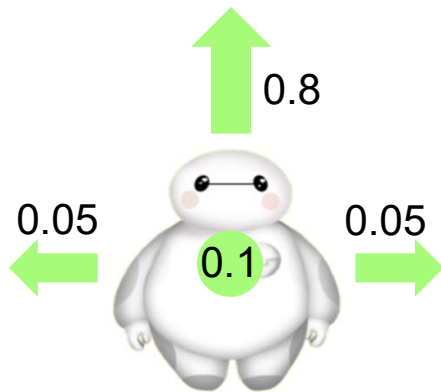
- **Stochastic Environment**
- **Planning Algorithms**
- **Learning Algorithms**

Stochastic Environment

Grid World



State Transition



State transition probabilities:

After the agent picks and performs a certain action, there are **four possibilities** for the next state: the destination state, the current state, the states to the right and left of the current state. If the states are **reachable**, the corresponding probabilities are 0.8, 0.1, 0.05, and 0.05, respectively; otherwise, the agent stays where it is.

State Transition

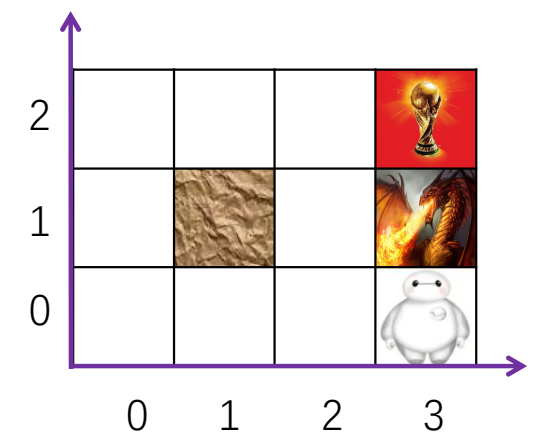
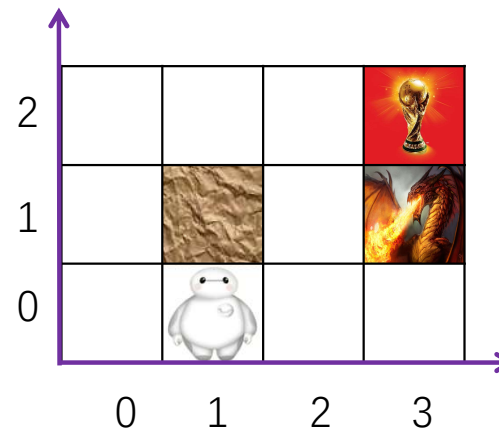
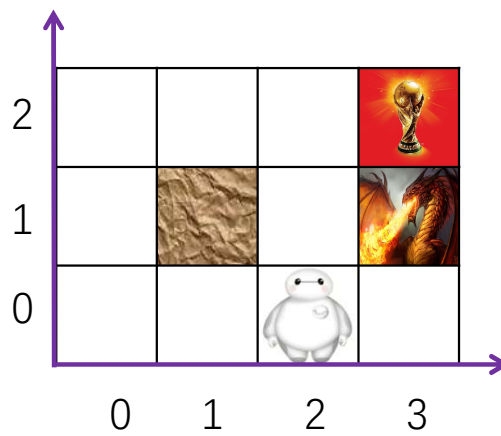


$$P((2,1)|(2,0), \text{up}) = 0.8$$

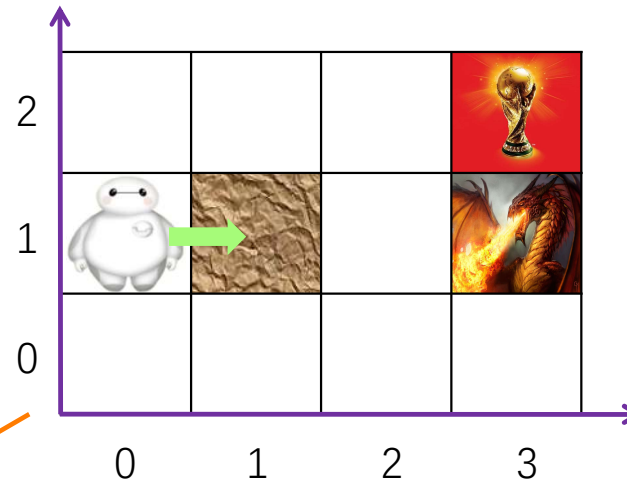
$$P((2,0)|(2,0), \text{up}) = 0.1$$

$$P((1,0)|(2,0), \text{up}) = 0.05$$

$$P((3,0)|(2,0), \text{up}) = 0.05$$



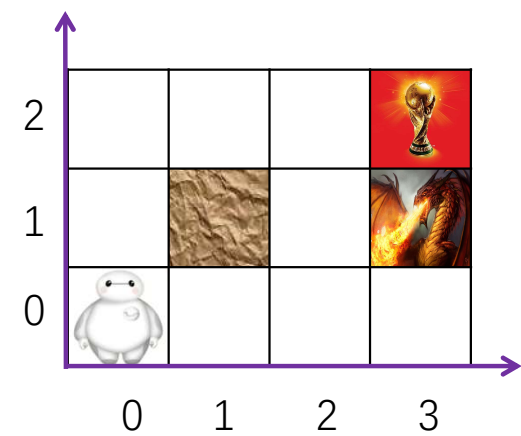
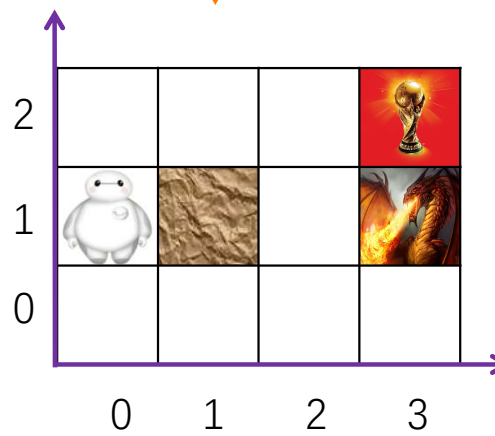
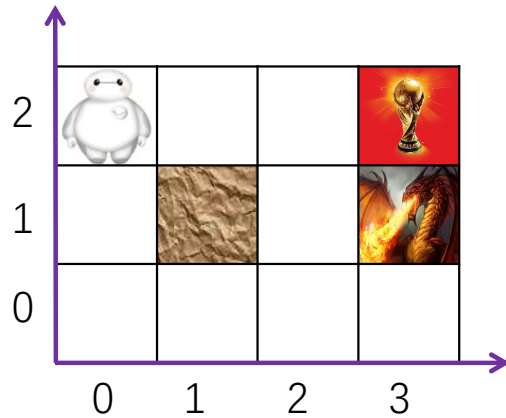
State Transition



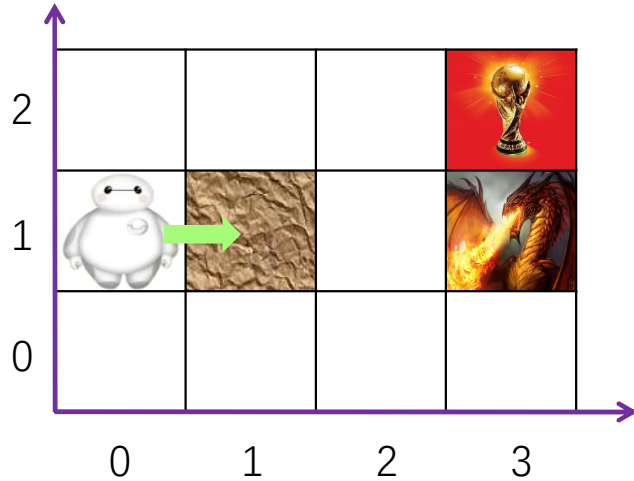
$$\mathbf{P}((2,0)|(1,0), \text{right}) = 0.05$$

$$\mathbf{P}((1,0)|(1,0), \text{right}) = 0.9$$

$$\mathbf{P}((0,0)|(1,0), \text{right}) = 0.05$$

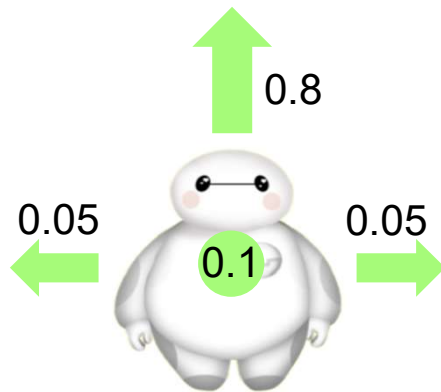


Reward

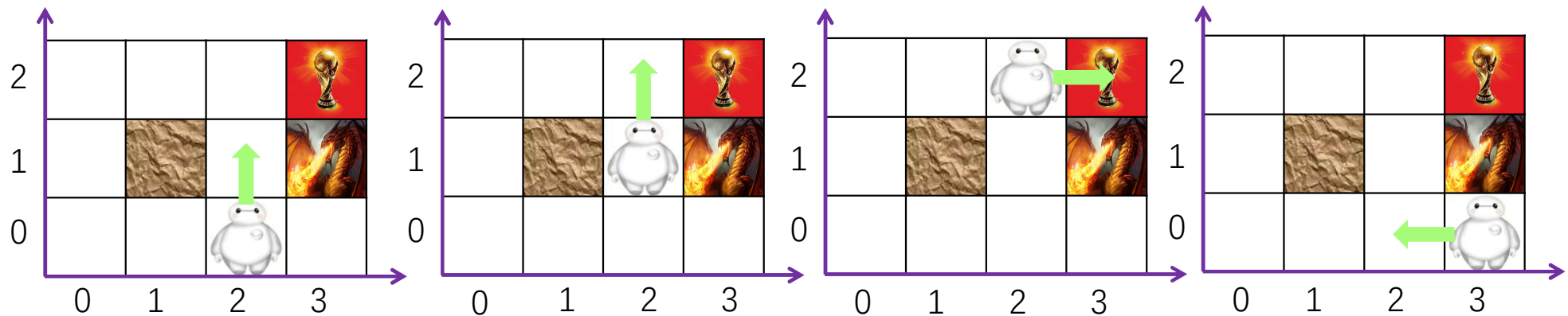


Reward:

After the agent picks and performs a certain action at its current state, it receives rewards of 100, -100, and 0, if it **arrives at** states (3,2), (2,2), and all the other states, respectively.



Reward



$$\mathbf{E}[r((2,0), \text{up})] = 0.8 \times 0 + 0.1 \times 0 + 0.05 \times 0 + 0.05 \times 0 = 0$$

$$\mathbf{E}[r((2,1), \text{up})] = 0.8 \times 0 + 0.1 \times 0 + 0.05 \times -100 + 0.05 \times 0 = -5$$

$$\mathbf{E}[r((2,2), \text{right})] = 0.8 \times 100 + 0.1 \times 0 + 0.05 \times 0 + 0.05 \times 0 = 80$$

$$\mathbf{E}[r((3,0), \text{left})] = 0.8 \times 0 + 0.1 \times 0 + 0.05 \times -100 + 0.05 \times 0 = -5$$

Markov Decision Process (MDP)

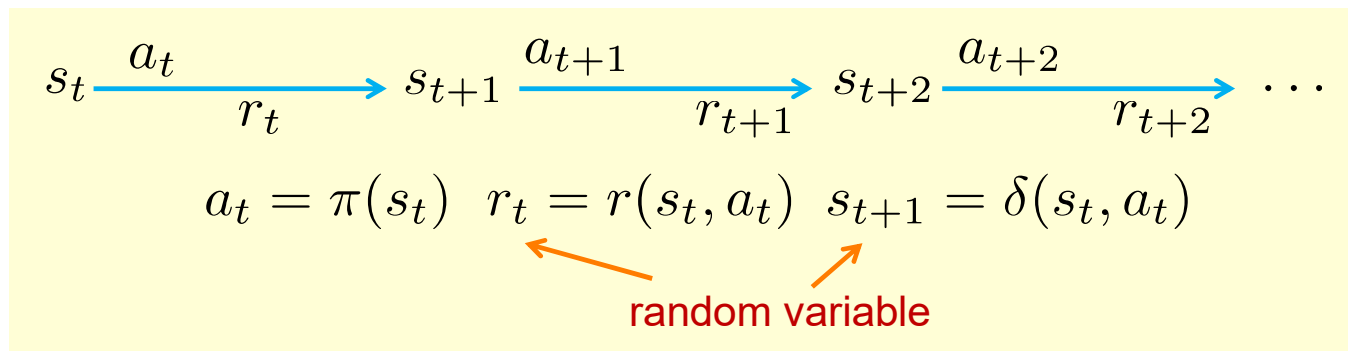
- Indeed, we have already introduced the so-called MDP, which is defined (rigorously) by
 - a set of **states** \mathcal{S} , possibly infinite
 - a set of **actions** \mathcal{A} , possibly infinite
 - an initial state $s_0 \in \mathcal{S}$
 - a **transition** probability $\mathbf{P}[s'|s, a]$: distribution over destination states $s' = \delta(s, a)$
 - a **reward** probability $\mathbf{P}[r|s, a]$: distribution over rewards $r' = r(s, a)$
- This model is **Markovian** because the transition and reward probabilities only depend on the current state and the action picked and performed at the current state, instead of the previous sequence of states and actions performed.
- In this lecture, we assume that
 - the states and the actions are **finite**

[MRT](#) Chapter 14

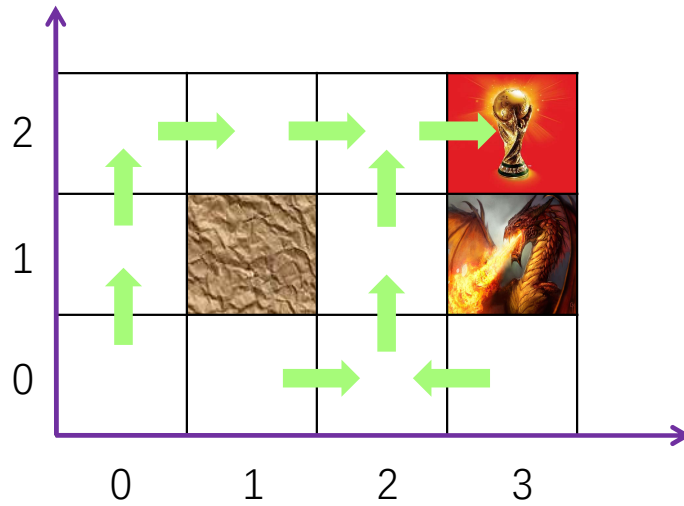
Value Function

- Suppose that a policy π is given.
- Starting from an arbitrary state s_t , the **expected** cumulative reward by following π is

$$V^\pi(s_t) := \mathbf{E}[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots | S_t = s_t] = \mathbf{E} \left[\sum_{i=0}^{\infty} \gamma^i R_{t+i} | S_t = s_t \right]$$

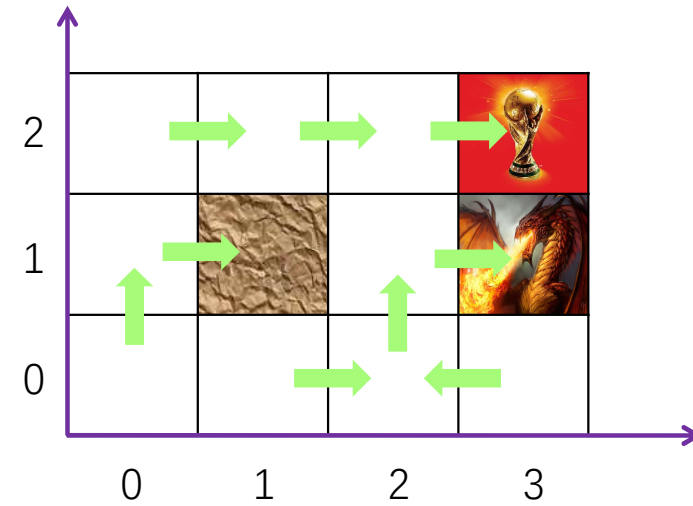


Value Function



A "good" policy π_1

$\gamma = 0.9$



A bad policy π_2

How to find V^{π_1} and V^{π_2} ?

Value Function

- Tower property

$$\mathbf{E}[X|Y] = \mathbf{E}[\mathbf{E}[X|Y, Z]|Y]$$

- A simpler version

$$\mathbf{E}[X] = \mathbf{E}[\mathbf{E}[X|Z]]$$

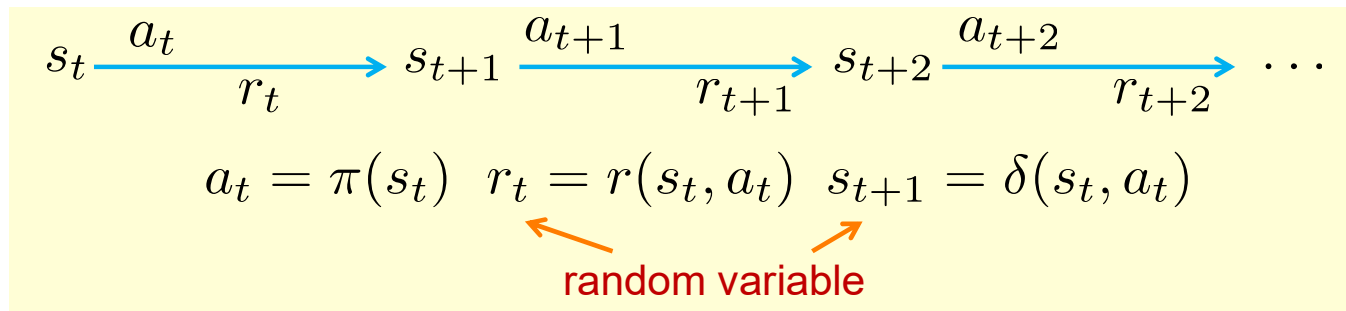
- Example: how to find the average height of the men in China?

$$\mathbf{E}[\text{height}] = \mathbf{E}[\mathbf{E}[\text{height}|\text{province}]] = \sum_{\text{province}} \mathbf{P}(\text{province})\mathbf{E}[\text{height}|\text{province}]$$

Value Function – Bellman Equation

- Starting from an arbitrary state s_t , the expected cumulative reward by following π is

$$V^\pi(s_t) := \mathbf{E}[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots | S_t = s_t] = \mathbf{E} \left[\sum_{i=0}^{\infty} \gamma^i R_{t+i} | S_t = s_t \right]$$



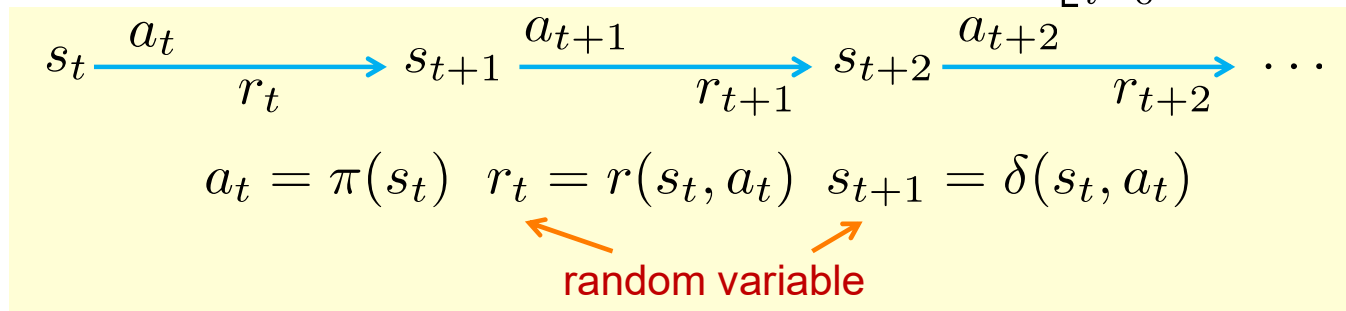
- Bellman Equation**

$$V^\pi(s) = \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s' | s, \pi(s)) V^\pi(s')$$

Value Function – Bellman Equation

- Starting from an arbitrary state s_t , the expected cumulative reward by following π is

$$V^\pi(s) := \mathbf{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots | s_t = s] = \mathbf{E} \left[\sum_{i=0}^{\infty} \gamma^i r_{t+i} | s_t = s \right]$$



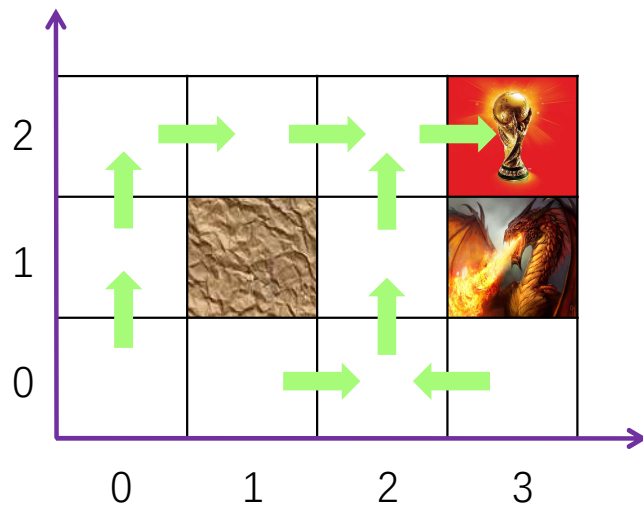
- Bellman Equation**

$$\begin{aligned}
 V^\pi(s) &= \mathbf{E}[r_t + \gamma (r_{t+1} + \gamma r_{t+2} + \dots) | S_t = s] \\
 &= \mathbf{E}[r(s, \pi(s))] + \gamma \mathbf{E}[\mathbf{E}[r_{t+1} + \gamma r_{t+2} + \dots | S_{t+1} = s', S_t = s] | S_t = s] \\
 &= \mathbf{E}[r(s, \pi(s))] + \gamma \mathbf{E}[\mathbf{E}[r_{t+1} + \gamma r_{t+2} + \dots | S_{t+1} = s'] | S_t = s] \\
 &= \mathbf{E}[r(s, \pi(s))] + \gamma \mathbf{E}[V^\pi(s') | S_t = s] \\
 &= \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s' | s, \pi(s)) V^\pi(s')
 \end{aligned}$$

Value Function – Bellman Equation

- Bellman Equation**

$$V^\pi(s) = \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s'|s, \pi(s)) V^\pi(s')$$



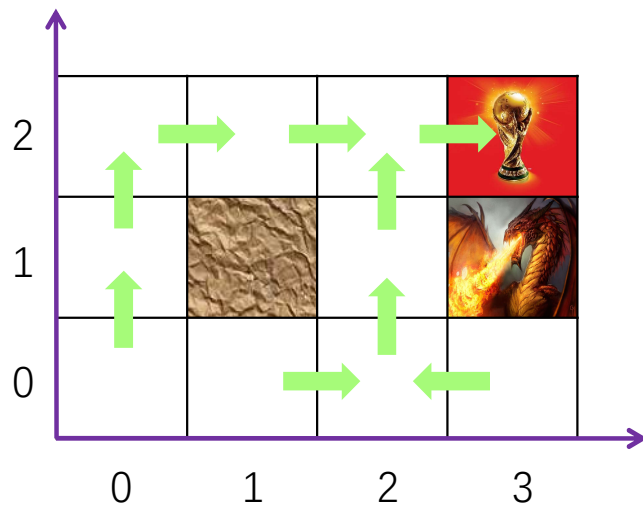
A “good” policy π_1 ?

$$\begin{pmatrix} V^{\pi_1}((0,0)) \\ V^{\pi_1}((1,0)) \\ V^{\pi_1}((2,0)) \\ V^{\pi_1}((3,0)) \\ V^{\pi_1}((0,1)) \\ V^{\pi_1}((1,1)) \\ V^{\pi_1}((2,1)) \\ V^{\pi_1}((3,1)) \\ V^{\pi_1}((0,2)) \\ V^{\pi_1}((1,2)) \\ V^{\pi_1}((2,2)) \\ V^{\pi_1}((3,2)) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -5 \\ 0 \\ 0 \\ -5 \\ 0 \\ 0 \\ 0 \\ 80 \\ 0 \end{pmatrix} + \gamma \begin{pmatrix} 0.15 & 0.05 & 0 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.05 & 0.1 & 0.05 & 0 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.8 & 0.15 & 0 & 0 & 0 & 0.05 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.2 & 0 & 0 & 0 & 0.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.15 & 0.05 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.05 & 0.15 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.05 & 0 & 0 & 0 & 0.15 & 0.8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} V^{\pi_1}((0,0)) \\ V^{\pi_1}((1,0)) \\ V^{\pi_1}((2,0)) \\ V^{\pi_1}((3,0)) \\ V^{\pi_1}((0,1)) \\ V^{\pi_1}((1,1)) \\ V^{\pi_1}((2,1)) \\ V^{\pi_1}((3,1)) \\ V^{\pi_1}((0,2)) \\ V^{\pi_1}((1,2)) \\ V^{\pi_1}((2,2)) \\ V^{\pi_1}((3,2)) \end{pmatrix}$$

Value Function – Bellman Equation

- Bellman Equation**

$$V^\pi(s) = \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s'|s, \pi(s)) V^\pi(s')$$



A “good” policy π_1

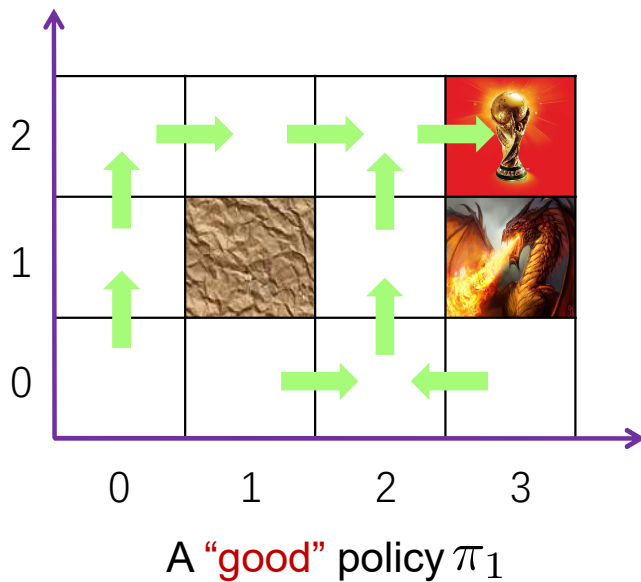
$$\begin{pmatrix} V^{\pi_1}((0,0)) \\ V^{\pi_1}((1,0)) \\ V^{\pi_1}((2,0)) \\ V^{\pi_1}((3,0)) \\ V^{\pi_1}((0,1)) \\ V^{\pi_1}((1,1)) \\ V^{\pi_1}((2,1)) \\ V^{\pi_1}((3,1)) \\ V^{\pi_1}((0,2)) \\ V^{\pi_1}((1,2)) \\ V^{\pi_1}((2,2)) \\ V^{\pi_1}((3,2)) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -5 \\ 0 \\ 0 \\ -5 \\ 0 \\ 0 \\ 0 \\ 80 \\ 0 \end{pmatrix} + \gamma \begin{pmatrix} 0.15 & 0.05 & 0 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.05 & 0.1 & 0.05 & 0 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.8 & 0.15 & 0 & 0 & 0 & 0.05 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.2 & 0 & 0 & 0 & 0.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.15 & 0.05 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.05 & 0.15 & 0.8 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.2 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.05 & 0 & 0 & 0 & 0.15 & 0.8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} V^{\pi_1}((0,0)) \\ V^{\pi_1}((1,0)) \\ V^{\pi_1}((2,0)) \\ V^{\pi_1}((3,0)) \\ V^{\pi_1}((0,1)) \\ V^{\pi_1}((1,1)) \\ V^{\pi_1}((2,1)) \\ V^{\pi_1}((3,1)) \\ V^{\pi_1}((0,2)) \\ V^{\pi_1}((1,2)) \\ V^{\pi_1}((2,2)) \\ V^{\pi_1}((3,2)) \end{pmatrix}$$

$$V = R + \gamma TV$$

Value Function – Bellman Equation

- Bellman Equation**

$$V^{\pi}(s) = \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s'|s, \pi(s))V^{\pi}(s')$$



$$V = R + \gamma TV$$

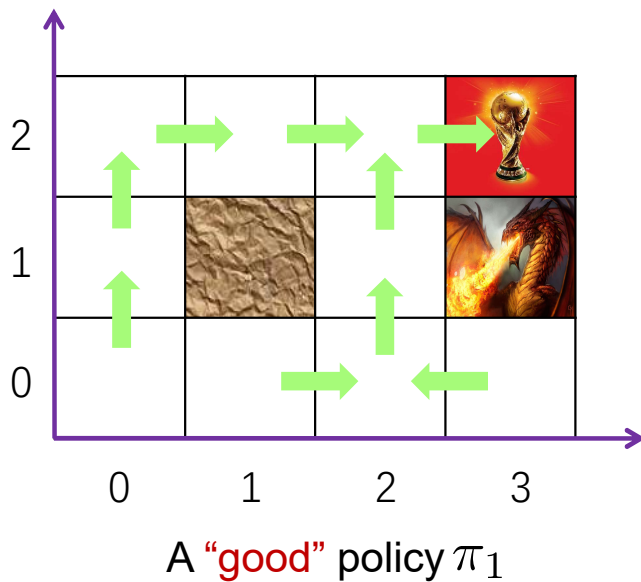
↓

$$V = (I - \gamma T)^{-1} R$$

Value Function – Bellman Equation

- Bellman Equation**

$$V^\pi(s) = \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s'|s, \pi(s)) V^\pi(s')$$



$$V = R + \gamma TV$$

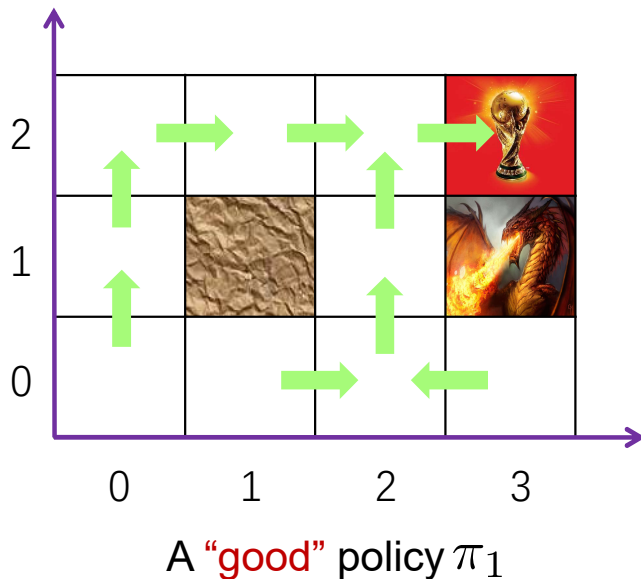
$$V = (I - \gamma T)^{-1} R$$

invertible?

Value Function – Bellman Equation

- Bellman Equation**

$$V^{\pi}(s) = \mathbf{E}[r(s, \pi(s))] + \gamma \sum_{s'} \mathbf{P}(s'|s, \pi(s)) V^{\pi}(s')$$



Theorem: For a finite MDP, Bellman's equation admits a unique solution that is given by

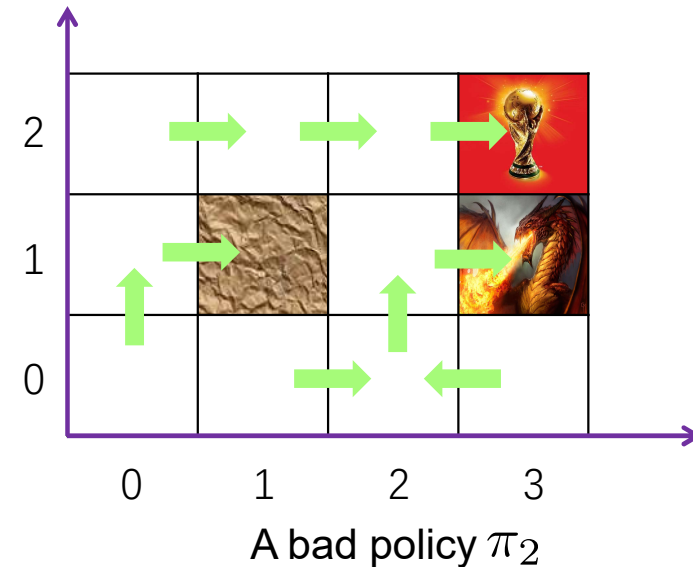
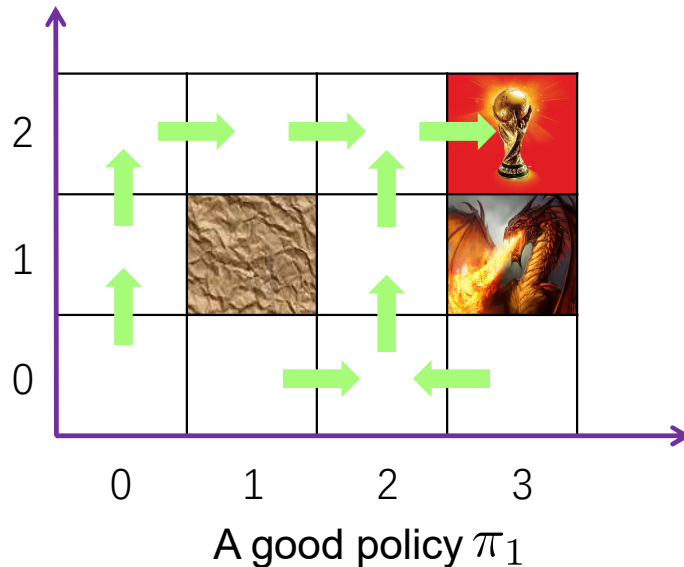
$$V = (I - \gamma T)^{-1} R$$

- The vector R and matrix T depend on the policy

The Learning Task Revisited

- The learning task for RL scenarios is to learn an **optimal policy** in the sense that

$$\pi^* := \operatorname{argmax}_{\pi} V^{\pi}(s), \quad \forall s.$$



- For π_1 and π_2 , we have

$$V^{\pi_1}(s) \geq V^{\pi_2}(s), \quad \forall s.$$

- Indeed, π_1 is the optimal policy.

The Q Function

- Learning the **optimal policy** is challenging
- An alternative approach to find the optimal policy indirectly is by computing the state-action value function (Q function)

$$Q(s, a) = \mathbf{E}[r(s, a)] + \gamma \sum_{s'} \mathbf{P}(s'|s, a) V^*(s')$$

$Q(s, a)$ is the expected accumulated reward by performing the action a first and then following the optimal policy

- The definition of the optimal policy implies that

$$\pi^*(s) = \operatorname{argmax}_a Q(s, a)$$

- Notice that

$$V^*(s) = \max_a Q(s, a)$$

- All together, we have

$$Q(s, a) = \mathbf{E}[r(s, a)] + \gamma \sum_{s'} \left[\mathbf{P}(s'|s, a) \max_{a'} Q(s', a') \right]$$

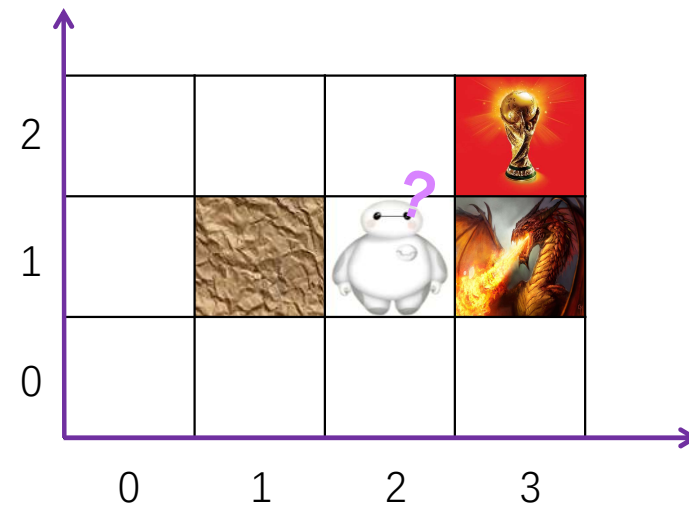
Bellman Equations



Quiz

- The learning task for RL scenarios is to learn an **optimal policy** in the sense that

$$\pi^* := \operatorname{argmax}_{\pi} V^{\pi}(s), \quad \forall s.$$

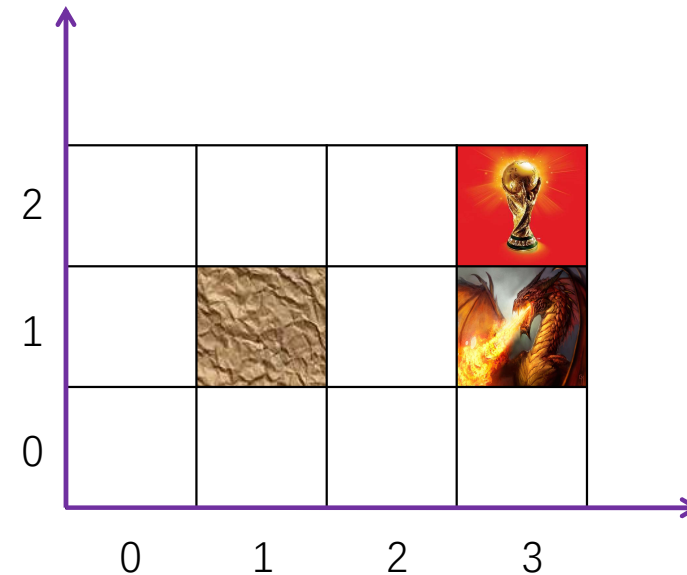


What are the best actions at states (3,0) and (2,1), i.e., $\pi^*((3,0))$ and $\pi^*((2,1))$?

Planning Algorithms

Planning

- Planning: we assume that the agent has perfect knowledge of the environment; thus, to find the optimal policy, there is no need for the agent to actually perform actions and interact with the environment



Known

$P(s'|s, a)$: state transition

$P(r|s, a)$: reward

Value Iteration

- Value iteration aims to find the optimal value function and thus the optimal policy

Initialize $V(s)$ to arbitrary values

while termination conditions does not hold

For $s \in \mathcal{S}$

For $a \in \mathcal{A}$

$$Q(s, a) \leftarrow \mathbf{E}[r(s, a)] + \gamma \sum_{s'} \mathbf{P}(s'|s, a) V(s')$$

$$V(s) \leftarrow \max_a Q(s, a)$$

Value Iteration

- Value iteration aims to find the optimal value function and thus the optimal policy



Example

$$V \leftarrow 0$$

$$Q((0,0), \text{up}) \leftarrow 0 + 0.9 \times (0.8 \times V((0,1)) + 0.1 \times V((0,0)) + 0.05 \times V((0,0)) + 0.05 \times V((1,0))) = 0$$

$$Q((0,0), \text{down}) \leftarrow 0 + 0.9 \times (0.95 \times V((0,0)) + 0.05 \times V((1,0))) = 0$$

$$Q((0,0), \text{left}) \leftarrow 0 + 0.9 \times (0.95 \times V((0,0)) + 0.05 \times V((0,1))) = 0$$

$$Q((0,0), \text{right}) \leftarrow 0 + 0.9 \times (0.8 \times V((1,0)) + 0.15 \times V((0,0)) + 0.05 \times V((0,1))) = 0$$

$$V((0,0)) \leftarrow \max\{Q((0,0), \text{up}), Q((0,0), \text{down}), Q((0,0), \text{left}), Q((0,0), \text{right})\} = 0$$

Value Iteration

- Value iteration aims to find the optimal value function and thus the optimal policy



Example

$$V \leftarrow 0$$

$$Q((0,0), \text{up}) \leftarrow 0 + 0.9 \times (0.8 \times V((0,1)) + 0.1 \times V((0,0)) + 0.05 \times V((0,0)) + 0.05 \times V((1,0))) = 0$$

$$Q((0,0), \text{down}) \leftarrow 0 + 0.9 \times (0.95 \times V((0,0)) + 0.05 \times V((1,0))) = 0$$

$$Q((0,0), \text{left}) \leftarrow 0 + 0.9 \times (0.95 \times V((0,0)) + 0.05 \times V((0,1))) = 0$$

$$Q((0,0), \text{right}) \leftarrow 0 + 0.9 \times (0.8 \times V((1,0)) + 0.15 \times V((0,0)) + 0.05 \times V((0,1))) = 0$$

$$V((0,0)) \leftarrow \max\{Q((0,0), \text{up}), Q((0,0), \text{down}), Q((0,0), \text{left}), Q((0,0), \text{right})\} = 0$$

Nothing happens

Value Iteration

- Value iteration aims to find the optimal value function and thus the optimal policy



Example

$$V \leftarrow 0$$

$$Q((2, 2), \text{up}) \leftarrow 5 + 0.9 \times (0.9 \times V((2, 2)) + 0.05 \times V((1, 2)) + 0.05 \times V((3, 2))) = 5$$

$$Q((2, 2), \text{down}) \leftarrow 5 + 0.9 \times (0.8 \times V((2, 1)) + 0.1 \times V((2, 2)) + 0.05 \times V((1, 2)) + 0.05 \times V((3, 2))) = 5$$

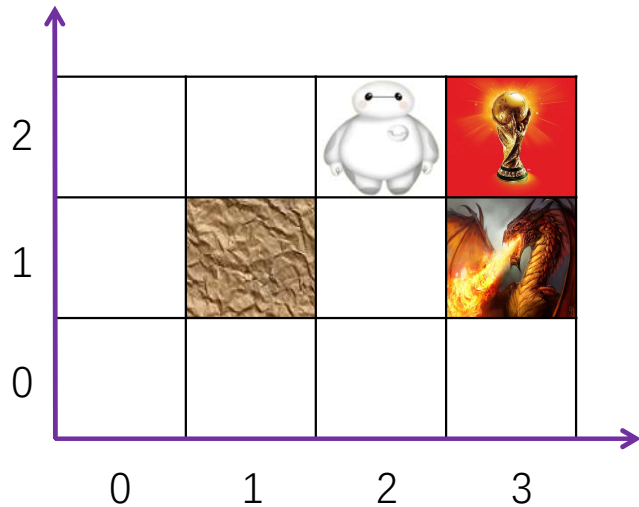
$$Q((2, 2), \text{left}) \leftarrow 0 + 0.9 \times (0.8 \times V((1, 2)) + 0.15 \times V((2, 2)) + 0.05 \times V((2, 1))) = 0$$

$$Q((2, 2), \text{right}) \leftarrow 80 + 0.9 \times (0.8 \times V((3, 2)) + 0.15 \times V((2, 2)) + 0.05 \times V((2, 1))) = 80$$

$$V((2, 2)) \leftarrow \max\{Q((2, 2), \text{up}), Q((2, 2), \text{down}), Q((2, 2), \text{left}), Q((2, 2), \text{right})\} = 80$$

Value Iteration

- Value iteration aims to find the optimal value function and thus the optimal policy



Value Iteration

- Value iteration aims to find the optimal value function and thus the optimal policy

Theorem: For any initial value V , the sequence generated by the value iteration algorithm converges to V^* .

- The key to the proof is the contraction mapping theorem

Policy Iteration

- Policy iteration improves the policy directly

Initialize $\pi \leftarrow \pi_2, \pi' \neq \pi_2$

while($\pi \neq \pi'$)

$$V \leftarrow (I - \gamma T^\pi)^{-1} R^\pi$$

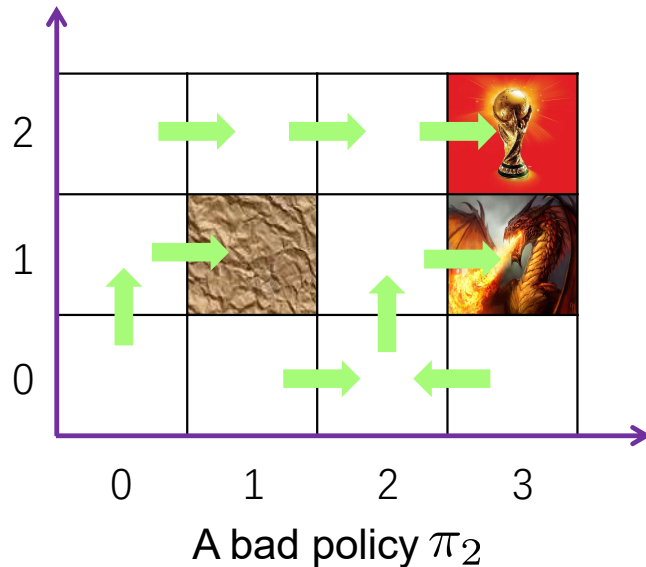
$$\pi' \leftarrow \pi$$

For $s \in \mathcal{S}$

$$\pi(s) \leftarrow \operatorname{argmax}_a \mathbf{E}[r(s, a)] + \gamma \sum_{s'} \mathbf{P}(s'|s, a) V(s')$$

Policy Iteration

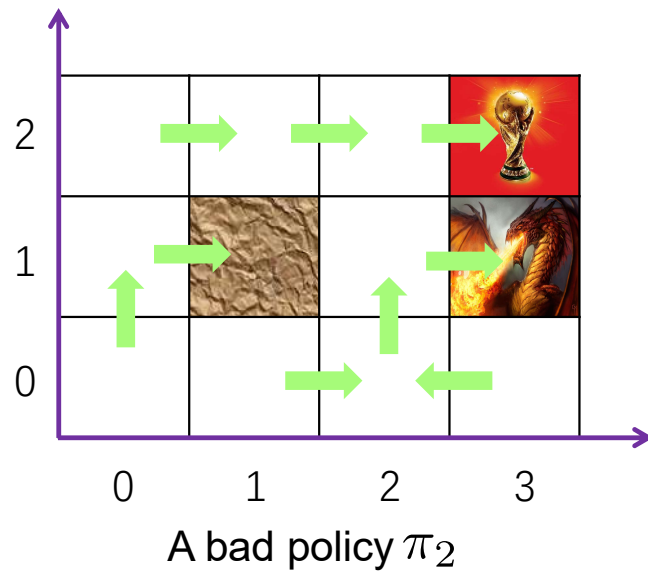
- Policy iteration improves the policy directly



Initialize $\pi \leftarrow \pi_2, \pi' \neq \pi_2$
while($\pi \neq \pi'$)
 1^{st} $V \leftarrow (I - \gamma T^\pi)^{-1} R^\pi$
 $\pi' \leftarrow \pi$
 For $s \in \mathcal{S}$
 $\pi(s) \leftarrow \operatorname{argmax}_a \mathbf{E}[r(s, a)] + \gamma \sum_{s'} \mathbf{P}(s'|s, a) V(s')$

Policy Iteration

- Policy iteration improves the policy directly



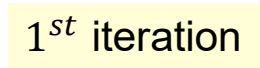
1st iteration

$$V^\pi = \begin{pmatrix} V^\pi(0,0) \\ V^\pi(1,0) \\ V^\pi(2,0) \\ V^\pi(3,0) \\ V^\pi(0,1) \\ V^\pi(2,1) \\ V^\pi(3,1) \\ V^\pi(0,2) \\ V^\pi(1,2) \\ V^\pi(2,2) \\ V^\pi(3,2) \end{pmatrix}$$

11 states in total

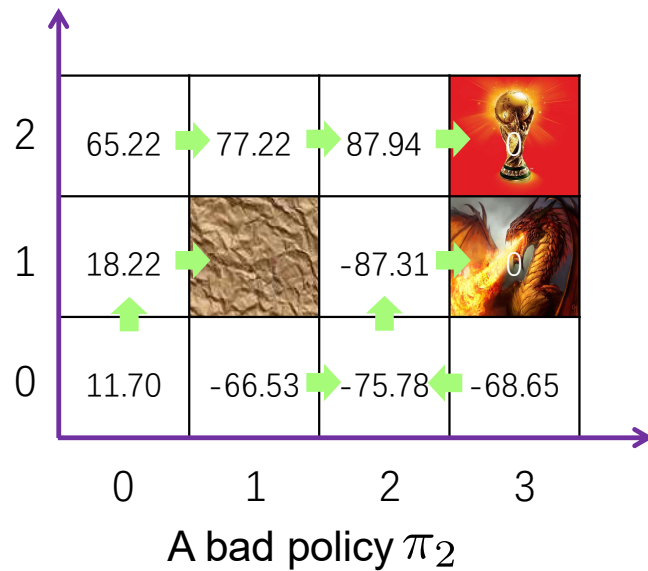
- Policy iteration improves the policy directly

- Policy iteration improves the policy directly



Policy Iteration

- Policy iteration improves the policy directly



1st iteration

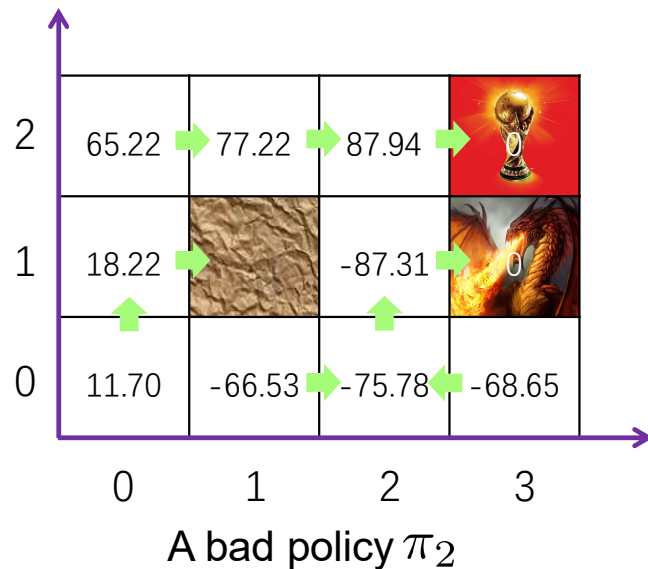
$$V \leftarrow (I - \gamma T^\pi)^{-1} R^\pi =$$

$$\begin{pmatrix} 11.70 \\ -66.53 \\ -75.78 \\ -68.85 \\ 18.22 \\ -87.31 \\ 0 \\ 65.22 \\ 77.22 \\ 87.94 \\ 0 \end{pmatrix}$$

Policy Iteration

- Policy iteration improves the policy directly

1st iteration: update the policy



$$\begin{aligned} Q((0,0), \text{up}) &= \mathbf{E}[r((0,0), \text{up})] + 0.9 \times (0.8 \times V((0,1)) + 0.15 \times V((0,0)) + 0.05 \times V((1,0))) \\ &= 0 + 0.9 \times (0.8 \times 18.22 + 0.15 \times 11.70 + 0.05 \times -66.53) \\ &= 11.70 \end{aligned}$$

$$\begin{aligned} Q((0,0), \text{down}) &= \mathbf{E}[r((0,0), \text{down})] + 0.9 \times (0.95 \times V((0,0)) + 0.05 \times V((1,0))) \\ &= 0 + 0.9 \times (0.95 \times 11.70 + 0.05 \times (-66.53)) \\ &= 7.01 \end{aligned}$$

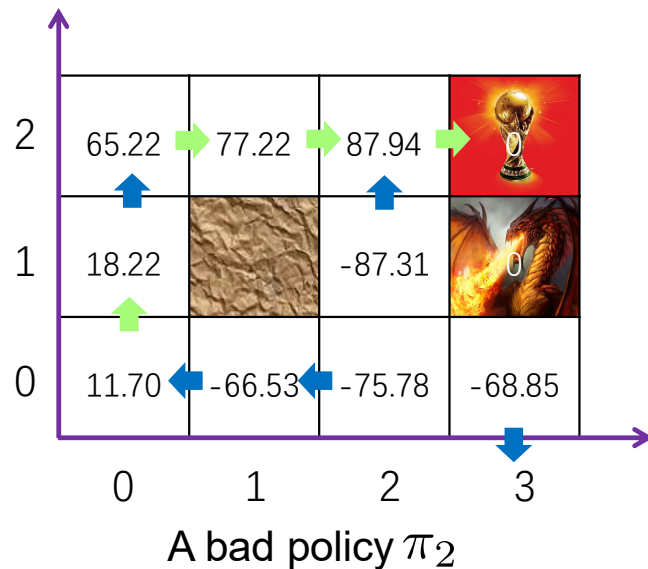
$$\begin{aligned} Q((0,0), \text{left}) &= \mathbf{E}[r((0,0), \text{left})] + 0.9 \times (0.95 \times V((0,0)) + 0.05 \times V((0,1))) \\ &= 0 + 0.9 \times (0.95 \times 11.70 + 0.05 \times 18.22) \\ &= 10.82 \end{aligned}$$

$$\begin{aligned} Q((0,0), \text{right}) &= \mathbf{E}[r((0,0), \text{right})] + 0.9 \times (0.8 \times V((1,0)) + 0.15 \times V((0,0)) + 0.05 \times V((0,1))) \\ &= 0 + 0.9 \times (0.8 \times (-66.53) + 0.15 \times 11.70 + 0.05 \times 18.22) \\ &= -45.50 \end{aligned}$$

$$\begin{aligned} \pi((0,0)) &= \operatorname{argmax}_{\{\text{up}, \text{down}, \text{left}, \text{right}\}} \{Q((0,0), \text{up}), Q((0,0), \text{down}), Q((0,0), \text{left}), Q((0,0), \text{right})\} \\ &= \text{up} \end{aligned}$$

Policy Iteration

- Policy iteration improves the policy directly



1st iteration: update the policy

$\pi((0, 0)) = \text{up}$

$\pi((1, 0)) = \text{left}$

$\pi((2, 0)) = \text{left}$

$\pi((3, 0)) = \text{down}$

$\pi((0, 1)) = \text{up}$

$\pi((2, 1)) = \text{up}$

$\pi((3, 1)) = \text{END}$

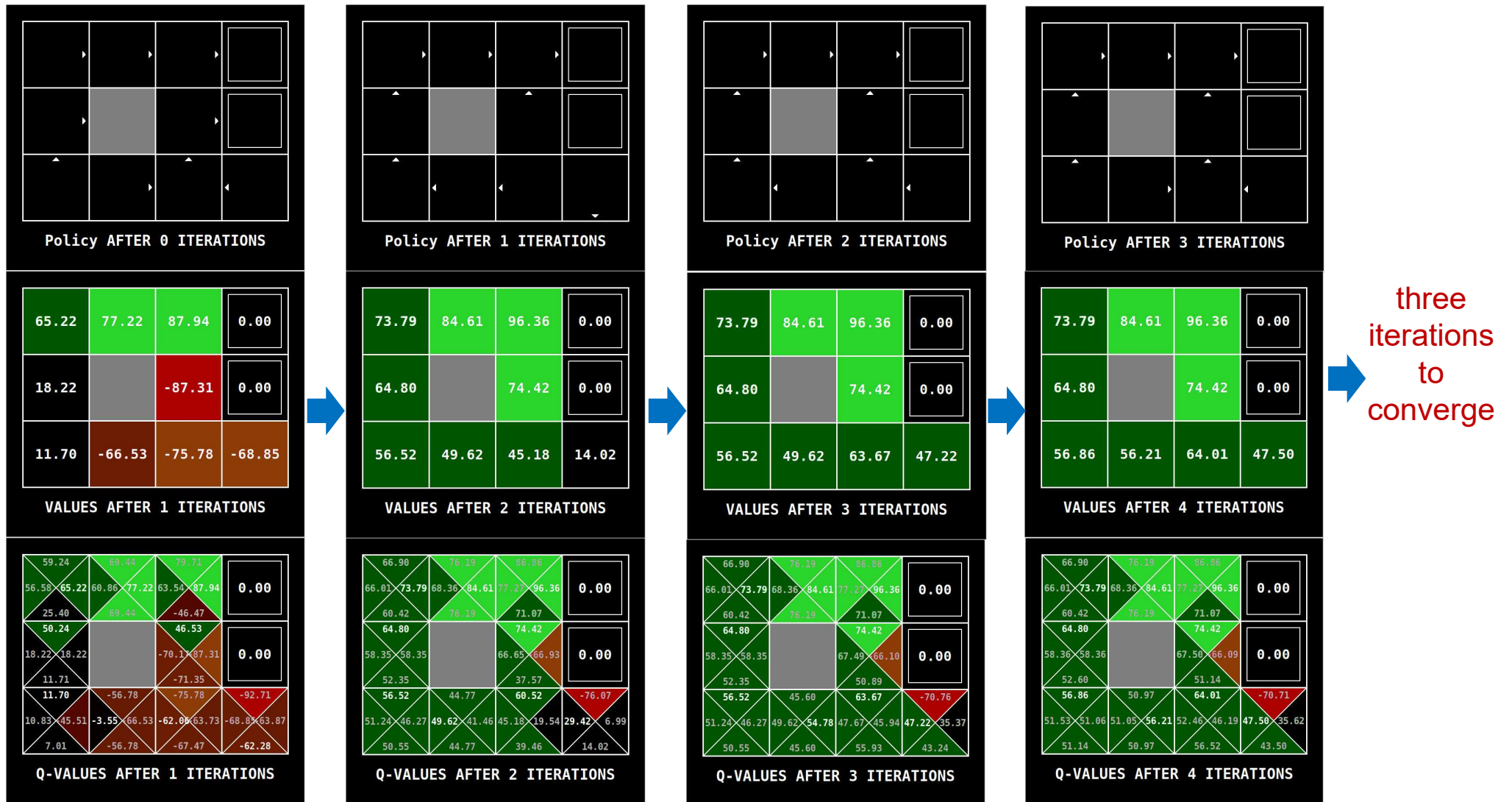
$\pi((0, 2)) = \text{right}$

$\pi((1, 2)) = \text{right}$

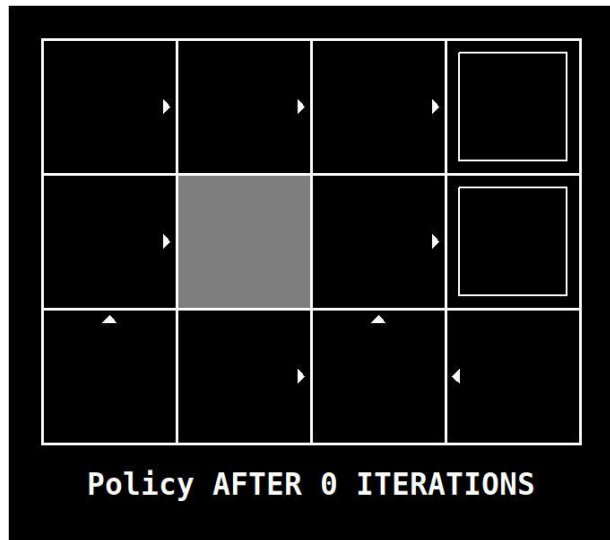
$\pi((2, 2)) = \text{right}$

$\pi((3, 2)) = \text{END}$

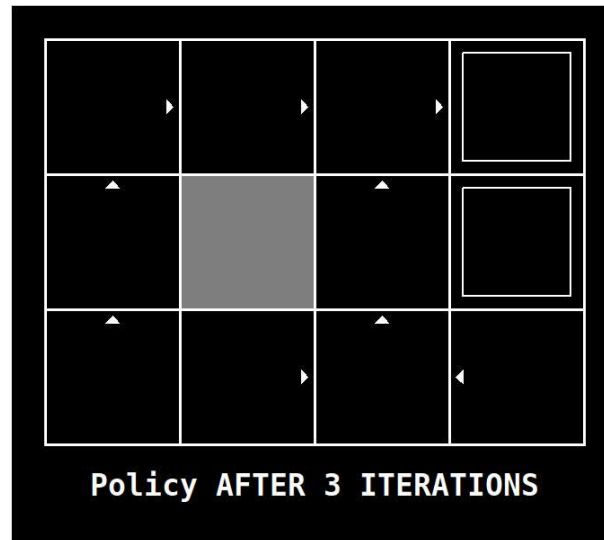
Policy Iteration



Policy Iteration



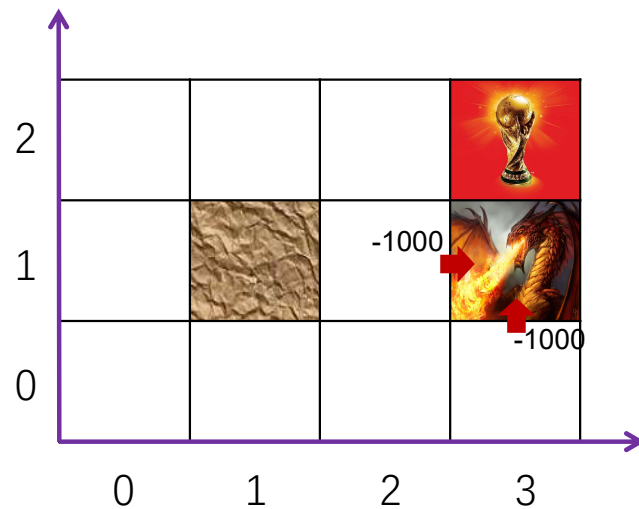
The optimal policy



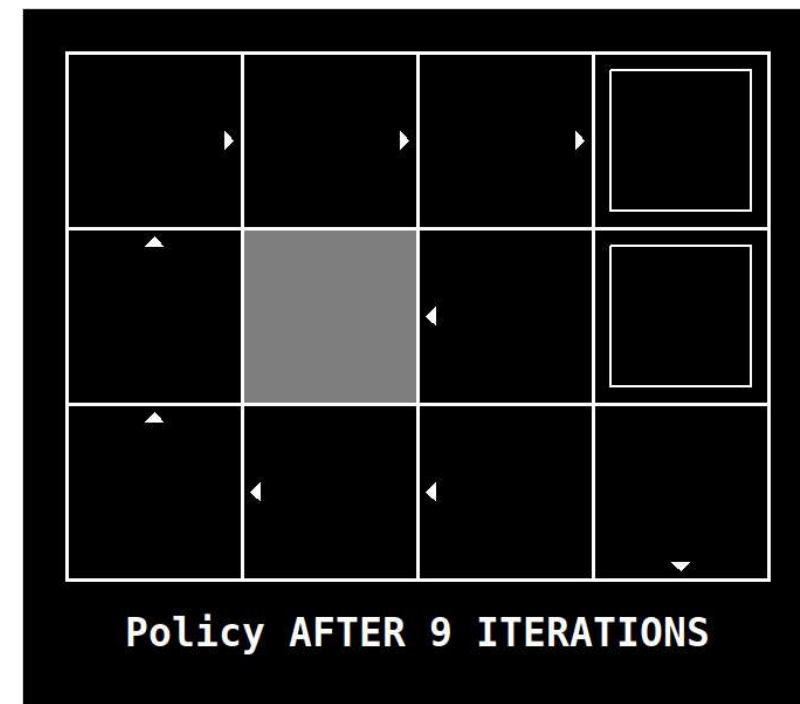
Is this an **always** winning policy?

Policy Iteration

- What if the reward for getting into (3,1) is -1000?



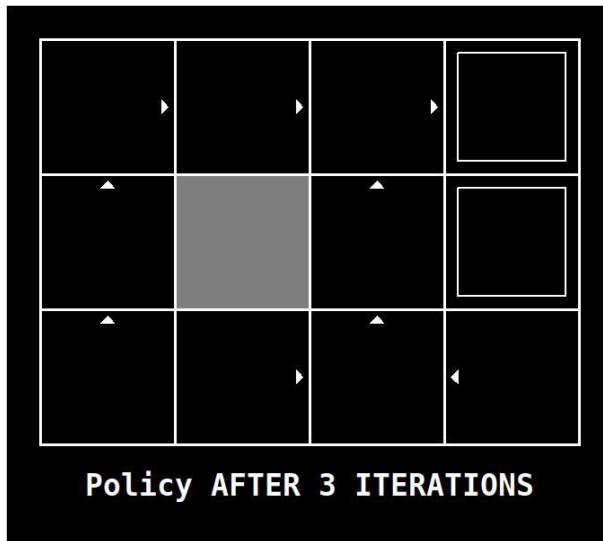
The optimal policy



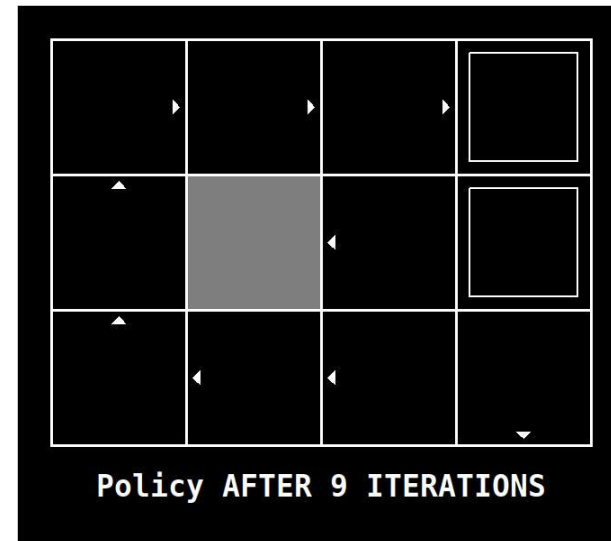
This is an **always** winning policy (why?).

Policy Iteration

A mostly winning policy



A never lose policy

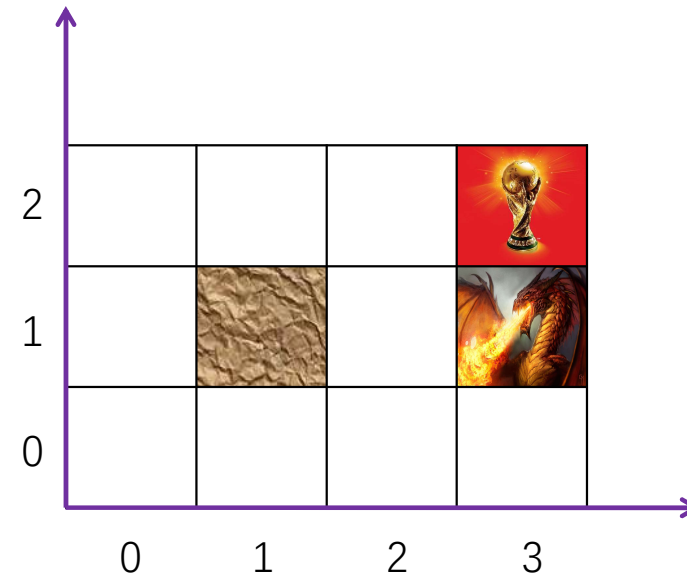


- For the same task, different reward strategies lead to different optimal policy
- According to your preference, you need to carefully design your reward strategy

Learning Algorithms

Learning

- Learning: as the environment model, i.e., the transition and reward, is unknown, the agent may need to learn them based on the training information.



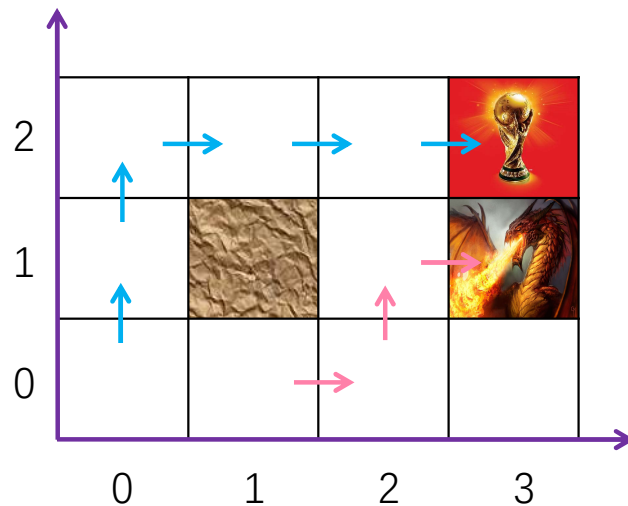
Unknown

$P(s'|s, a)$: state transition

$P(r|s, a)$: reward

Learning

- Learning: as the environment model, i.e., the transition and reward probabilities, is unknown, the agent may need to learn them based on the training information.
 - Model-free approach: the agent learns the optimal policy directly, e.g., Q-learning
 - Model-based approach: the agent first learns the environment model and then the optimal policy

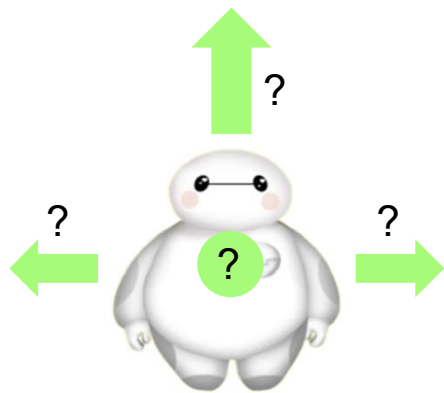


Examples of training data

$(0,0) \xrightarrow[0]{\text{up}} (0,1) \xrightarrow[0]{\text{up}} (0,2) \xrightarrow[0]{\text{right}} (1,2) \xrightarrow[0]{\text{right}} (2,3) \xrightarrow[100]{\text{right}} (3,2)$

$(1,0) \xrightarrow[0]{\text{right}} (2,0) \xrightarrow[0]{\text{up}} (2,1) \xrightarrow[-100]{\text{right}} (3,1)$

Nondeterministic Rewards and Actions



Unknown

$P(s'|s, a)$: state transition

$P(r|s, a)$: reward


How to find the optimal policy without the state transition and reward probabilities?

The Q-learning Algorithm

Recall the Q-learning algorithm for the **deterministic** environment

- Initialize the matrix \hat{Q} to zero
- Observe the current state s
- Do forever:
 - **Pick and perform** an action a
 - Receive immediate reward r
 - Observe the new state s'
 - Update

What if r and s' become random variables?

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$


- $s \leftarrow s'$

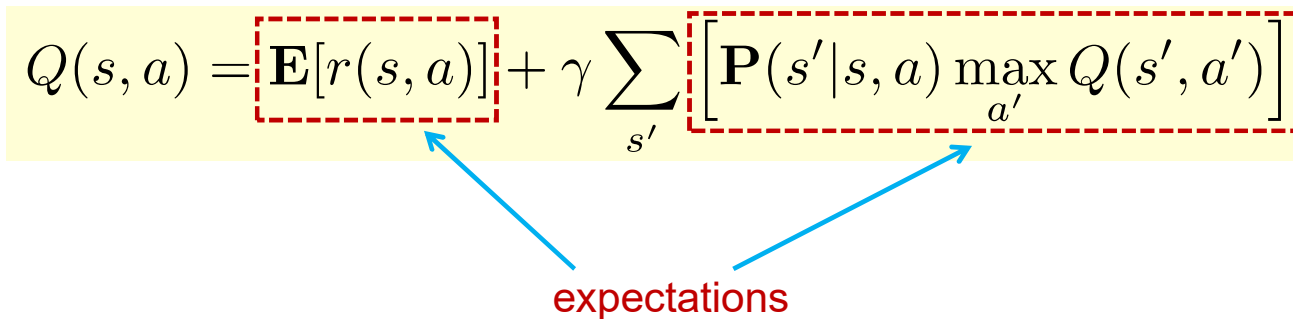
A sufficient condition for $\hat{Q}(s, a)$ to converge is to visit each state-action pair **infinitely often**

The Q-learning Algorithm

- For the stochastic environment, we replace the random variables with their **expectations** in the definition of Q values.

$$Q(s, a) = \mathbf{E}[r(s, a)] + \gamma \sum_{s'} \left[\mathbf{P}(s'|s, a) \max_{a'} Q(s', a') \right]$$

expectations



Q: How to find the expectation of a random variable X ?

A: Keep sampling and recording its running average

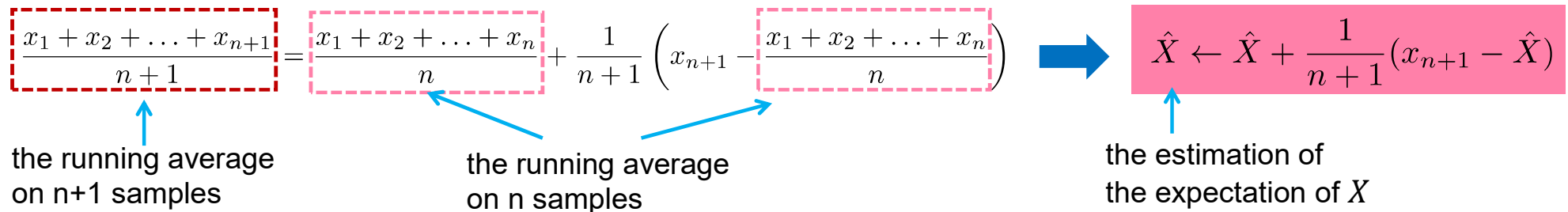
$$\frac{x_1 + x_2 + \dots + x_{n+1}}{n+1} = \frac{x_1 + x_2 + \dots + x_n}{n} + \frac{1}{n+1} \left(x_{n+1} - \frac{x_1 + x_2 + \dots + x_n}{n} \right)$$

the running average on $n+1$ samples

the running average on n samples

$$\hat{X} \leftarrow \hat{X} + \frac{1}{n+1} (x_{n+1} - \hat{X})$$

the estimation of the expectation of X



The Q-learning Algorithm

Initialize \hat{Q} arbitrarily

For all episodes

Initialize s

[Alpaydin 2014](#), Chapter 18

Repeat

Choose a using policy derived from Q , e.g., ϵ -greedy

Take action a , observe r and s'

Update $\hat{Q}(s, a)$:

$$\alpha_n = \frac{1}{1 + n((s, a))}$$

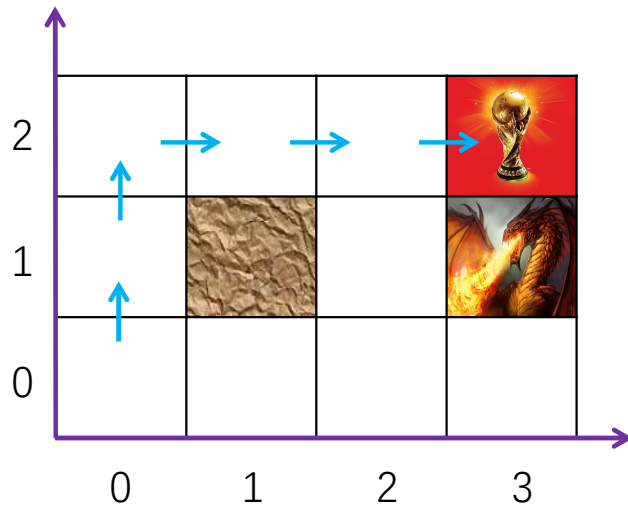

$$\hat{Q}(s, a) \leftarrow \hat{Q}(s, a) + \alpha_n (r + \gamma \max_{a'} \hat{Q}(s', a') - \hat{Q}(s, a))$$

$$s \leftarrow s'$$

Until s is goal state

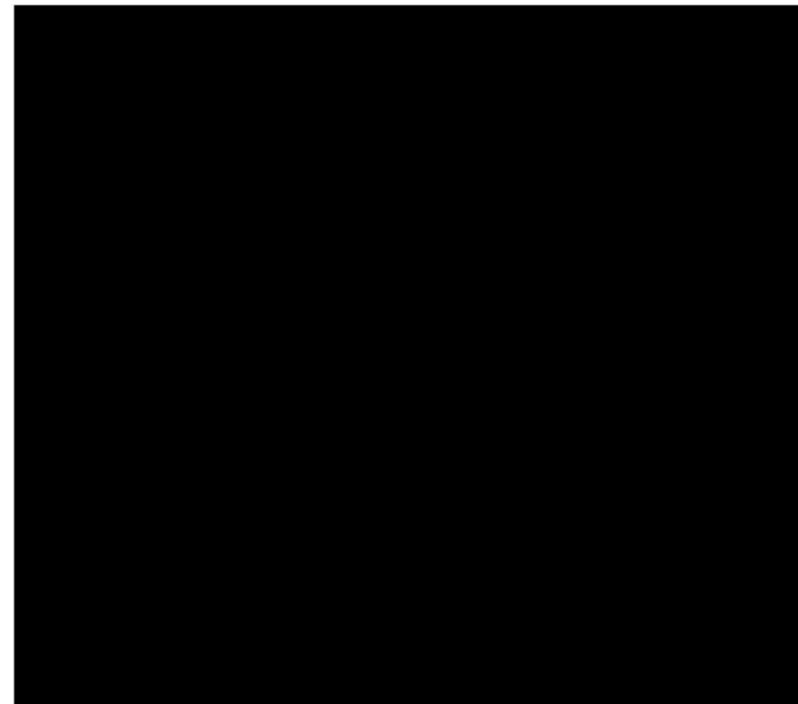
There are other ways to select α_n to guarantee that \hat{Q} converges to its optimal value. [Mitchell 1997](#), Chapter 13

The Q-learning Algorithm



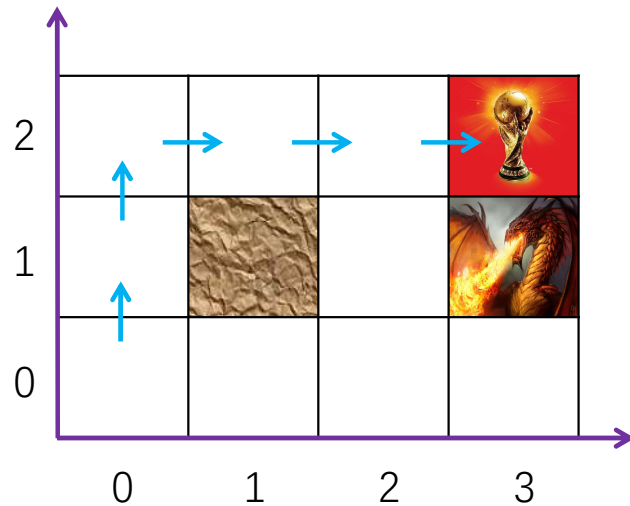
$(0,0) \xrightarrow[0]{\text{up}} (0,1) \xrightarrow[0]{\text{up}} (0,2) \xrightarrow[0]{\text{right}} (1,2) \xrightarrow[0]{\text{right}} (2,3) \xrightarrow[100]{\text{right}} (3,2)$

- an example episode
- the initial state in each episode could NOT be fixed (why?)



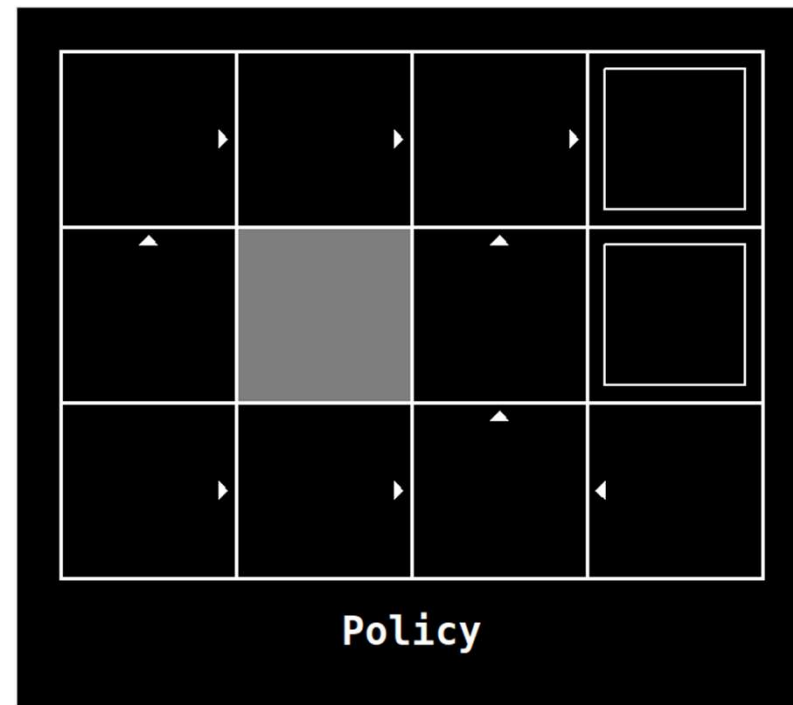
$$\epsilon = 0.3$$

The Q-learning Algorithm



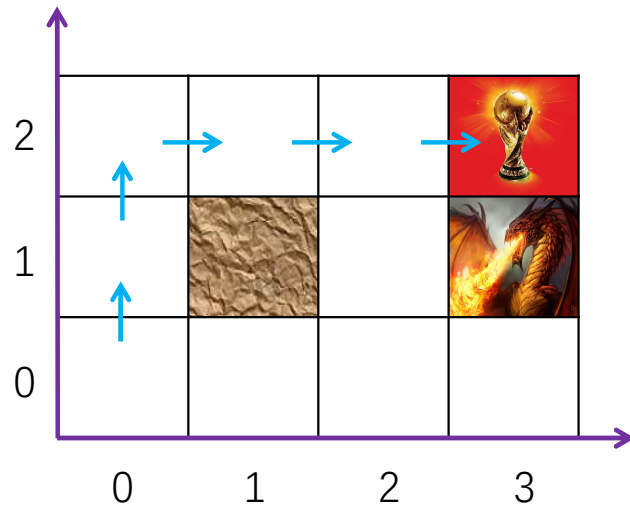
$(0,0) \xrightarrow[0]{\text{up}} (0,1) \xrightarrow[0]{\text{up}} (0,2) \xrightarrow[0]{\text{right}} (1,2) \xrightarrow[0]{\text{right}} (2,3) \xrightarrow[100]{\text{right}} (3,2)$

- an example episode
- the initial state in each episode could NOT be fixed (why?)



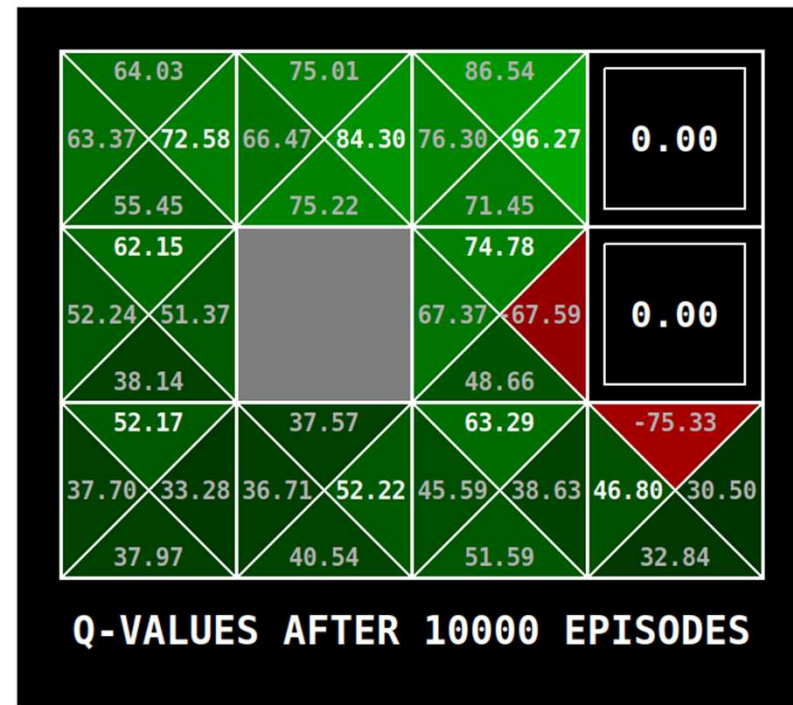
$\epsilon = 0.3$

The Q-learning Algorithm



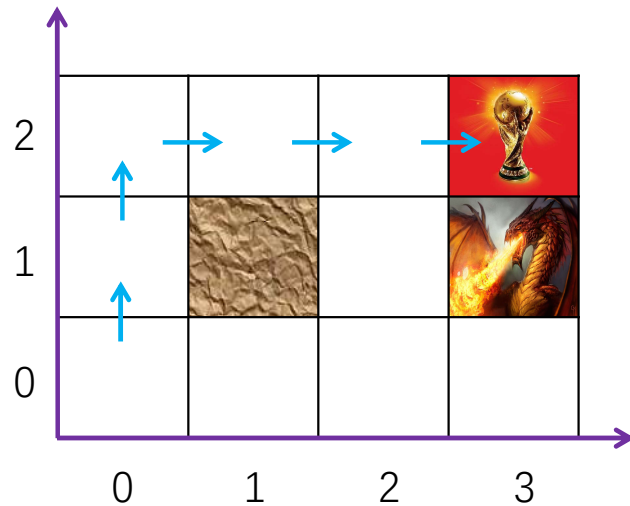
$(0,0) \xrightarrow[0]{\text{up}} (0,1) \xrightarrow[0]{\text{up}} (0,2) \xrightarrow[0]{\text{right}} (1,2) \xrightarrow[0]{\text{right}} (2,3) \xrightarrow[100]{\text{right}} (3,2)$

- an example episode
- the initial state in each episode could NOT be fixed (why?)



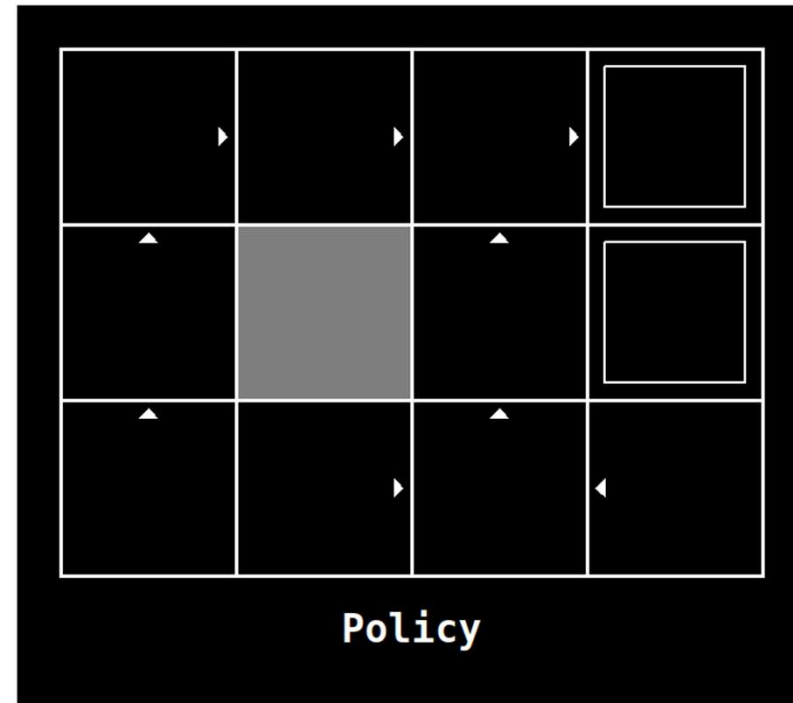
$$\epsilon = 0.3$$

The Q-learning Algorithm



$(0,0) \xrightarrow[0]{\text{up}} (0,1) \xrightarrow[0]{\text{up}} (0,2) \xrightarrow[0]{\text{right}} (1,2) \xrightarrow[0]{\text{right}} (2,3) \xrightarrow[100]{\text{right}} (3,2)$

- an example episode
- the initial state in each episode could NOT be fixed (why?)



$$\epsilon = 0.3$$

Questions



SARSA

Initialize $\hat{Q} \leftarrow 0$

For all episodes

Initialize s

Choose a using policy derived from Q , e.g., ϵ -greedy

Repeat

 Take action a , observe r and s'

 Choose a' using policy derived from Q , e.g., ϵ -greedy

 Update $\hat{Q}(s, a)$:

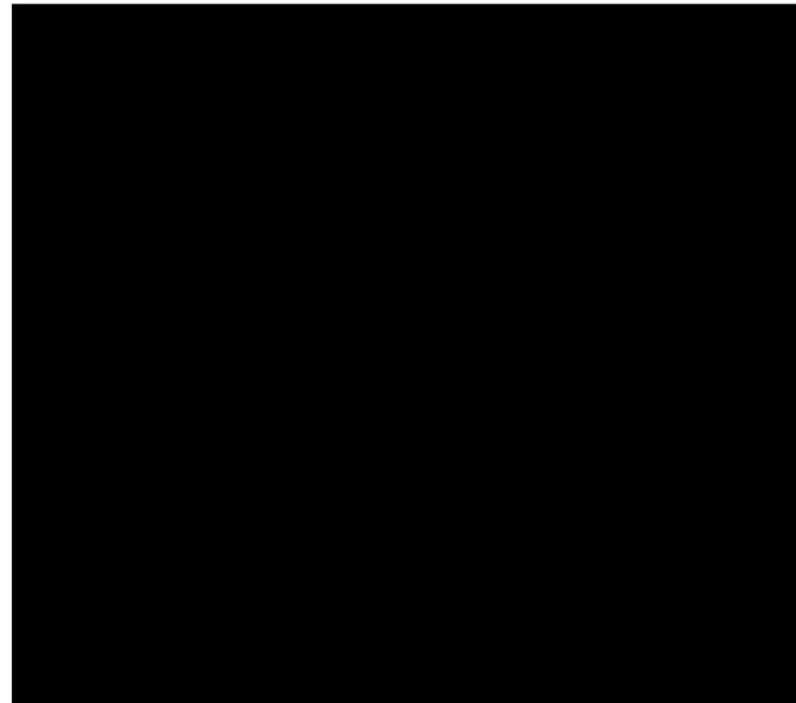
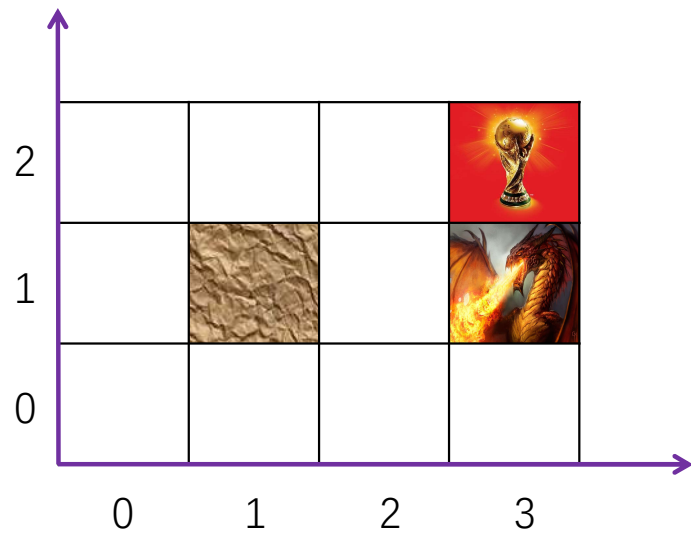
$$\alpha_n = \frac{1}{1 + n((s, a))}$$

$$\hat{Q}(s, a) \leftarrow \hat{Q}(s, a) + \alpha_n(r + \gamma\hat{Q}(s', a') - \hat{Q}(s, a))$$

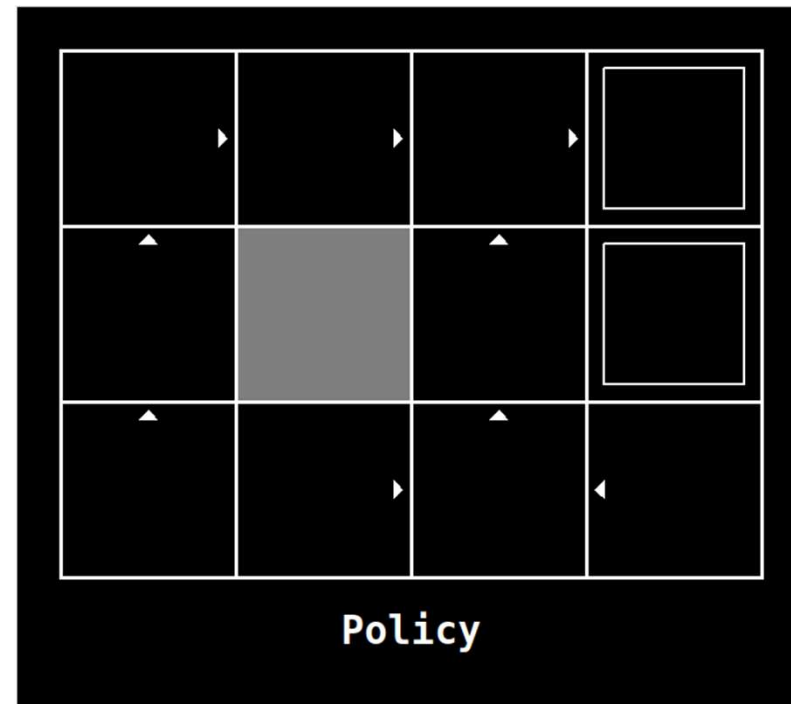
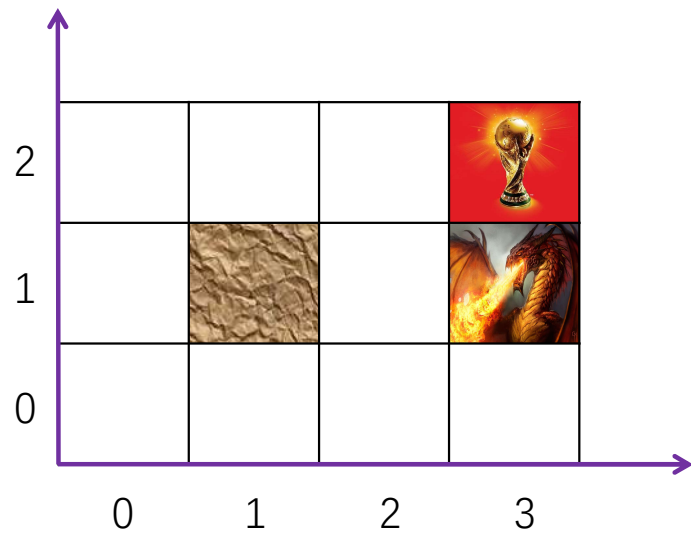
$s \leftarrow s', a \leftarrow a'$

Until s is goal state

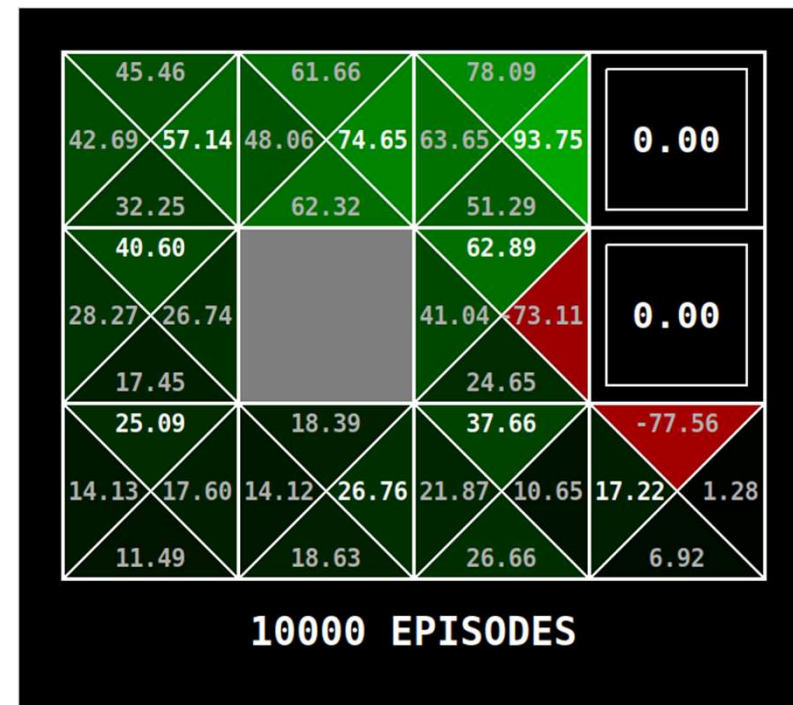
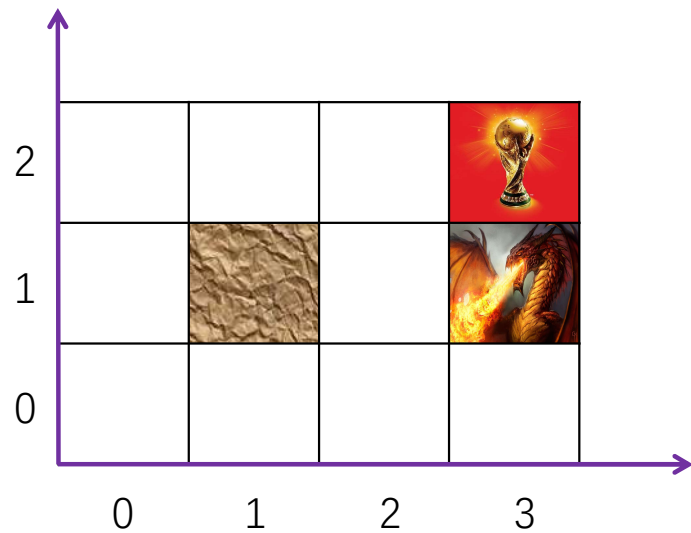
SARSA



SARSA



SARSA



SARSA

