

TUTORIAL #12

Session 1

Reinforcement Learning: Part 2

LECTURE OVERVIEW

01

Revision

What have we learned?

03

Q-Learning

What is it?

How do we use it?

02

Markov Decision Process

04

Hierarchical RL

Theory

REVISION

The intro to RL

The agent receives feedback in the form of rewards or penalties for each action it takes, allowing it to learn which actions are more likely to lead to positive outcomes.

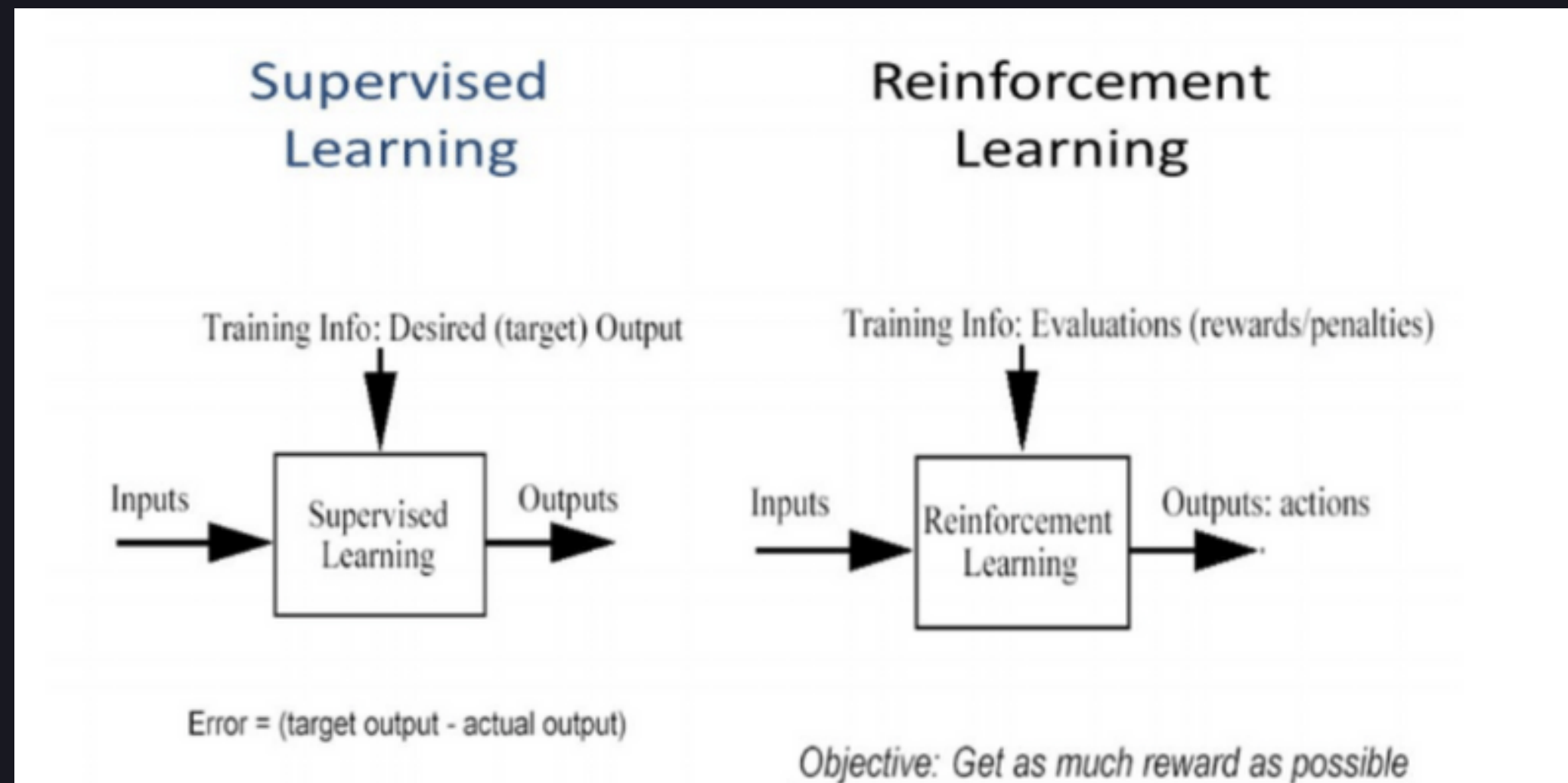
Reinforcement learning relies on agent learning to take actions to maximize rewards

This is embodied by policy, which is a function that determines what action should agent take in each state

Theory

REVISION

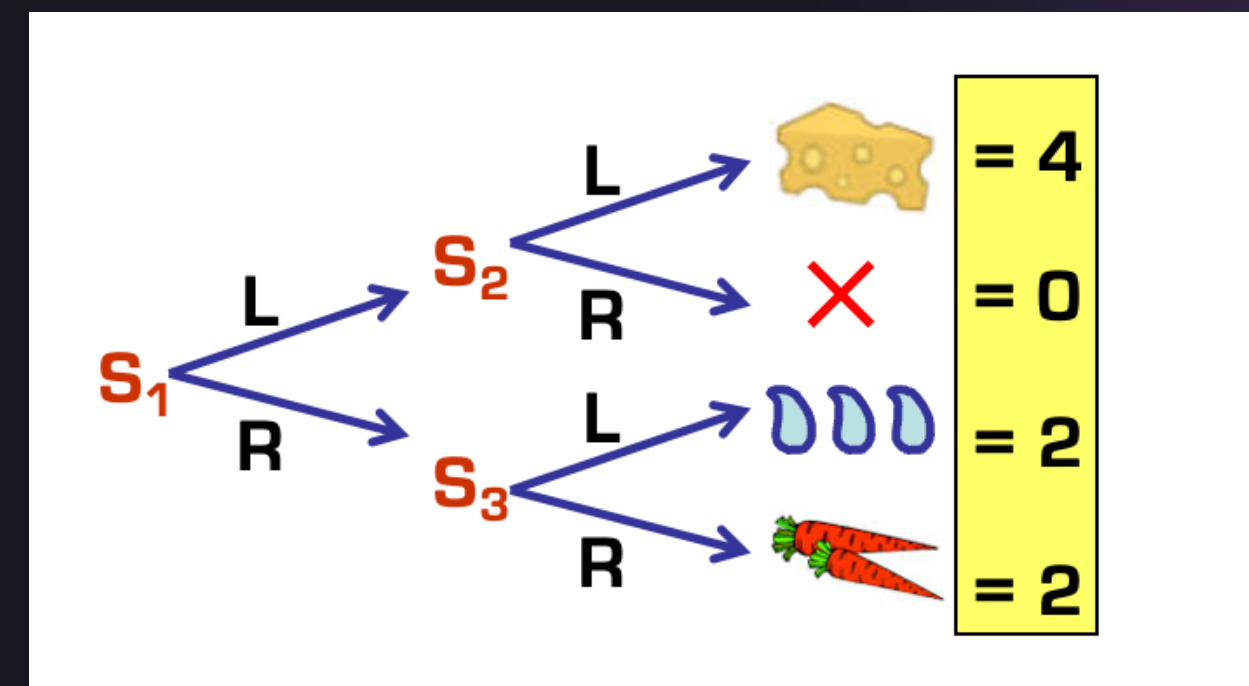
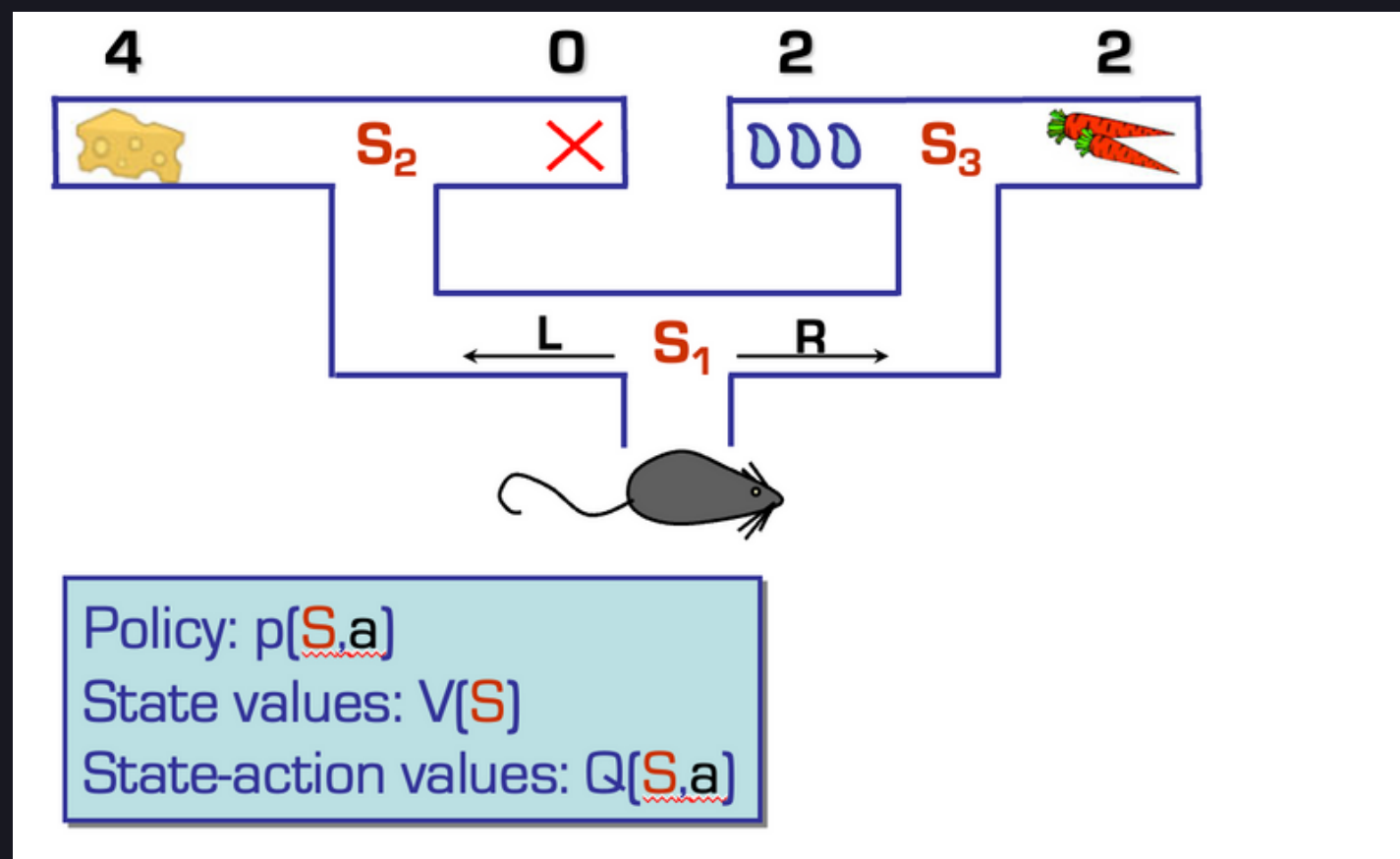
The intro to RL



Theory

REVISION

The intro to RL

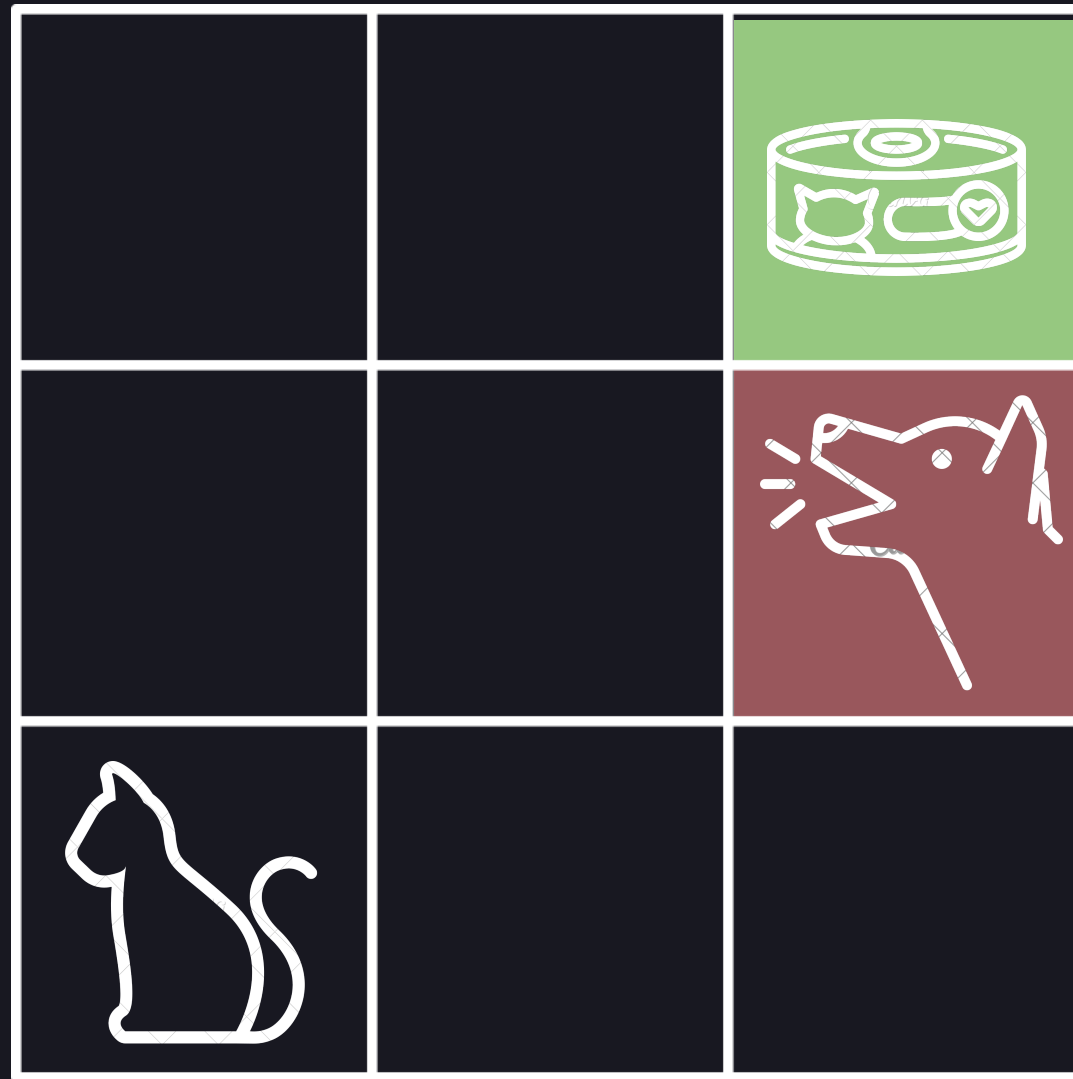


Theory

MARKOV DECISION PROCESS



Markov Decision Process: Example Game



Markov Decision Process: Formal Definition

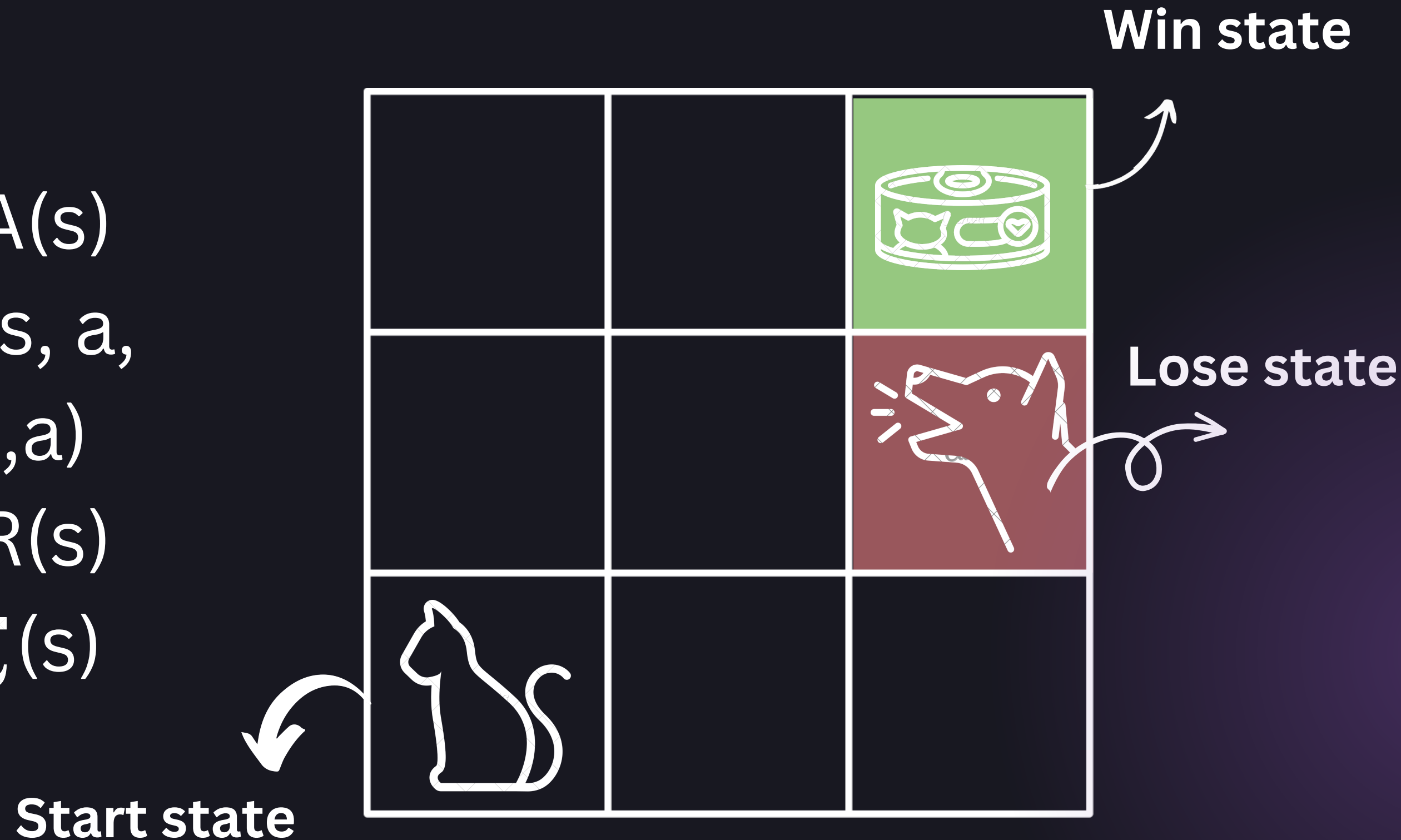
Tuple: (S, A, Pr, R)

- States: s
- Actions: $A(s)$
- Model: $T(s, a, s'), Pr(s'|s, a)$
- Reward: $R(s)$
- Policy: $\pi(s)$



States

- **States: s**
- Actions: $A(s)$
- Model: $T(s, a, s'), \text{Pr}(s'|s, a)$
- Reward: $R(s)$
- Policy: $\pi(s)$



Actions




- States: s
- **Actions: $A(s)$**
- Model: $T(s, a, s'), \text{Pr}(s'|s, a)$
- Reward: $R(s)$
- Policy: $\pi(s)$



Model




- States: s
- Actions: $A(s)$
- **Model: $T(s, a, s')$, $\Pr(s'|s, a)$**
- Reward: $R(s)$
- Policy: $\pi(s)$

For deterministic systems, the p is 0 or 1.

$p = 0$	$p = 0$	
$p = 1$	$p = 0$	
	$p = 0$	$p = 0$

Reward

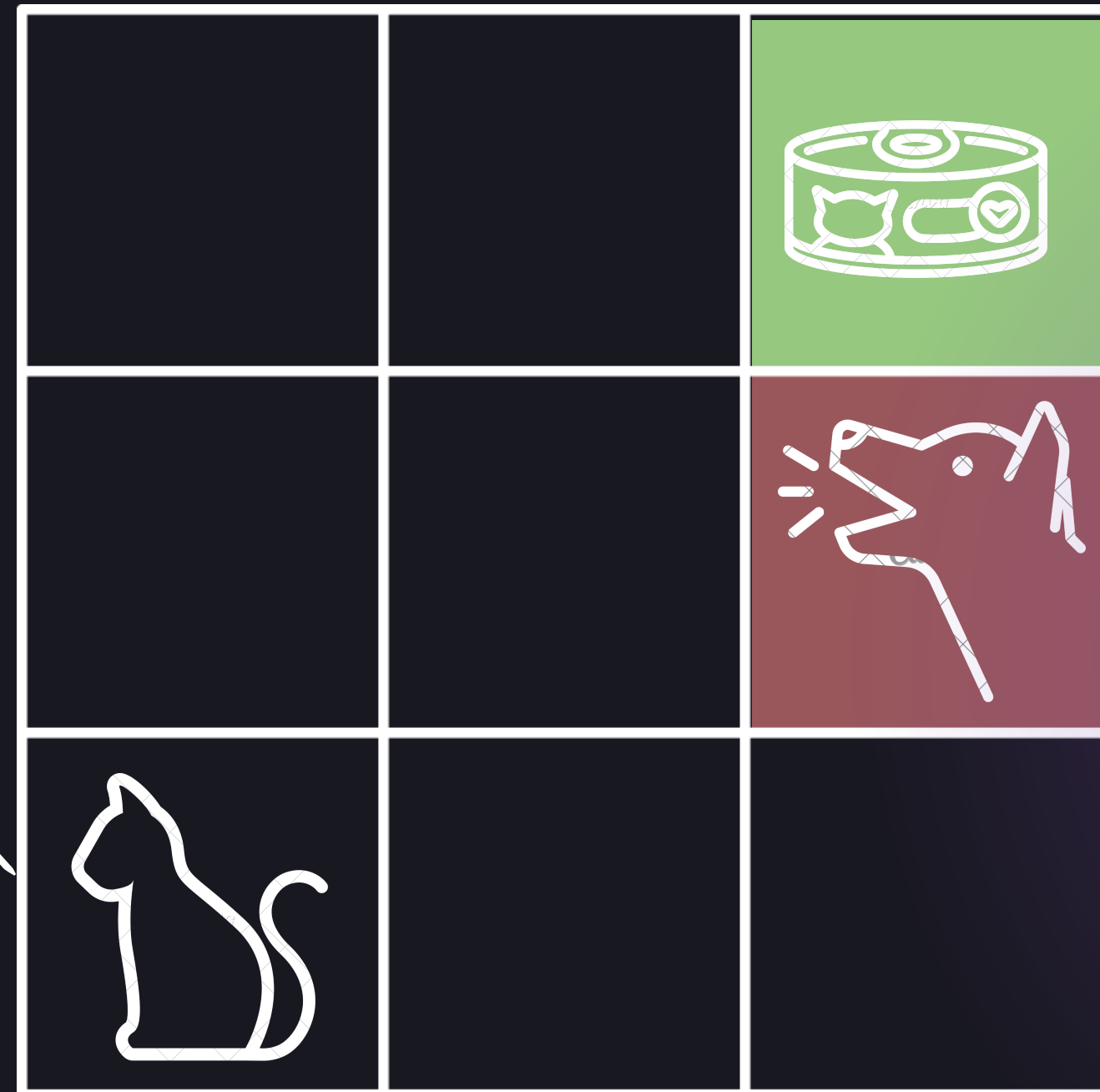
- States: s
- Actions: $A(s)$
- Model: $T(s, a, s')$, $\Pr(s'|s, a)$
- **Reward: $R(s)$**
- Policy: $\pi(s)$

$r = +2$		
$r = +1$		
	$r = -1$	$r = -2$

Policy

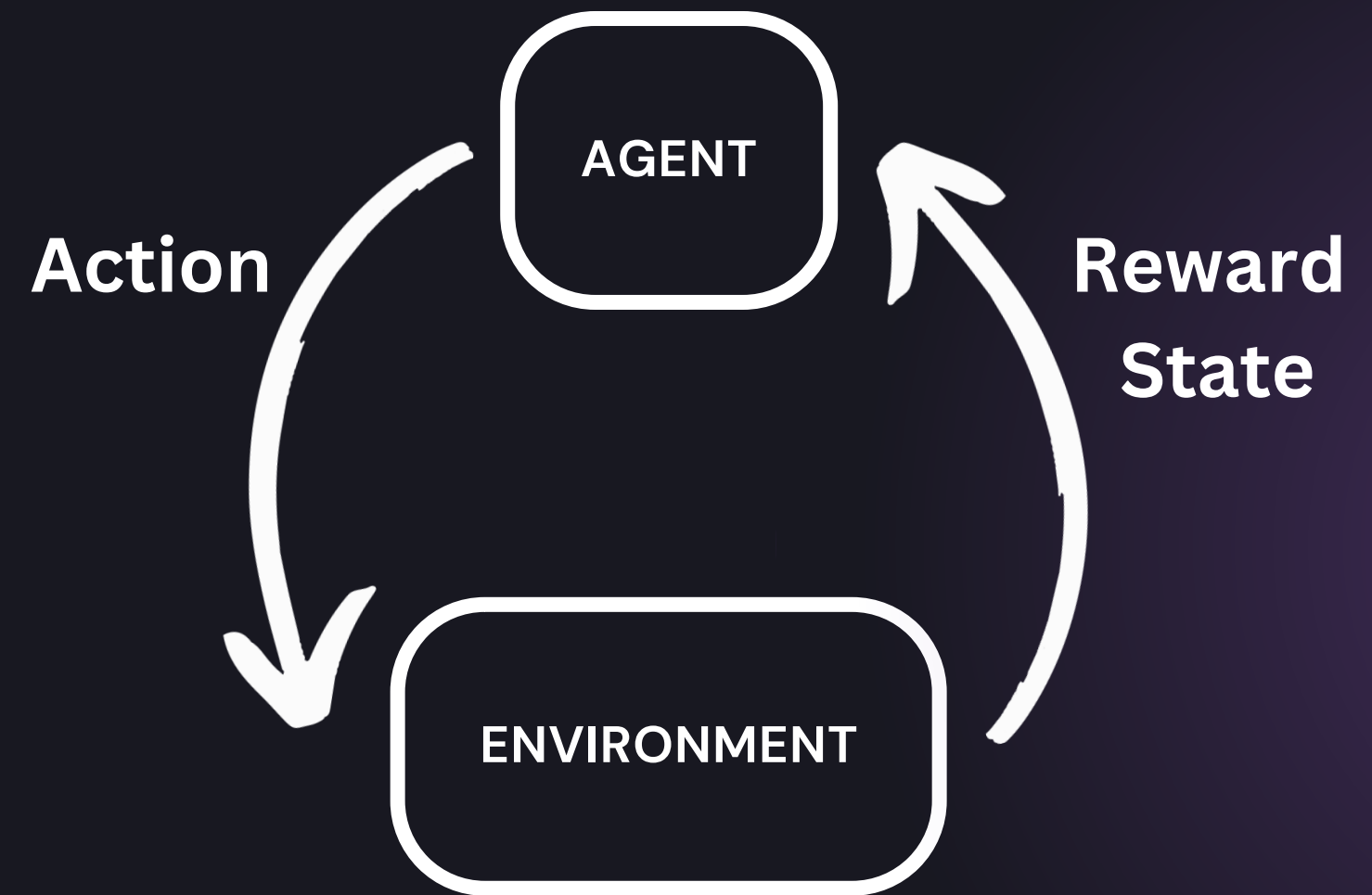
- States: s
- Actions: $A(s)$
- Model: $T(s, a, s')$,
 $\Pr(s'|s,a)$
- Reward: $R(s)$
- **Policy: $\pi(s)$**

What is my best next move?



MDP Key Points

1. We only care about the present.
2. The states and rules in the game do not change.



MDP Limitation

Requires precise
predicitons of P and R ->
Introducing Q-learning.



Theory

Q-LEARNING

Definition

A **model-free, gradient-free** algorithm that maximises the expected value of the total reward over any and all successive steps, including the current step. It uses Temporal Difference Learning (TD)

Model Free: it does not require a model of the environment to learn an optimal policy. Instead, the optimal policy is learnt by interacting with the environment. The Q-values are estimated using the reward function and using this we select the best action to take next.

Gradient Free: it does not use gradient descent or other optimization techniques to update its Q-values. Instead, Q-learning uses a simple iterative update rule based on the Bellman equation.

Theory

Q-LEARNING

Definition

TD: TD learning updates the Q-values incrementally as the agent interacts with the environment, i.e., with every time step.

Monte Carlo: Updates Q-values at the end of an episode. Where an episode is a series of steps that the agent takes to get to a reward/when max steps has expired.

Theory

Q-LEARNING

Quality Function

Each Q-value $Q(a,s)$ is a scalar that represents the cumulative reward (current + all future rewards) that an agent can get by taking an action a in state s and following a policy. This function is usually implemented as a look-up table

The Q-learning algorithm updates the Q-values for each state-action pair using the bellman equation.

Theory

Q-LEARNING

$Q(1,down)=4, Q(1,right)=6$
 $Q(2,left)=2, Q(2,right)=4, Q(2,down)=3$
 $Q(3,left)=5, Q(3,down)=7$
 $Q(4,right)=4, Q(4,up)=8$
 $Q(5,left)=5, Q(5,right)=6, Q(5,up)=8$

1	2	3
4	5	6

State/Action	1	2	3	4	5	6 (Final)
Up				8	8	
Down	4	3	7			
Left		2	5		5	
Right	6	4		4	6	

Theory

Q-LEARNING

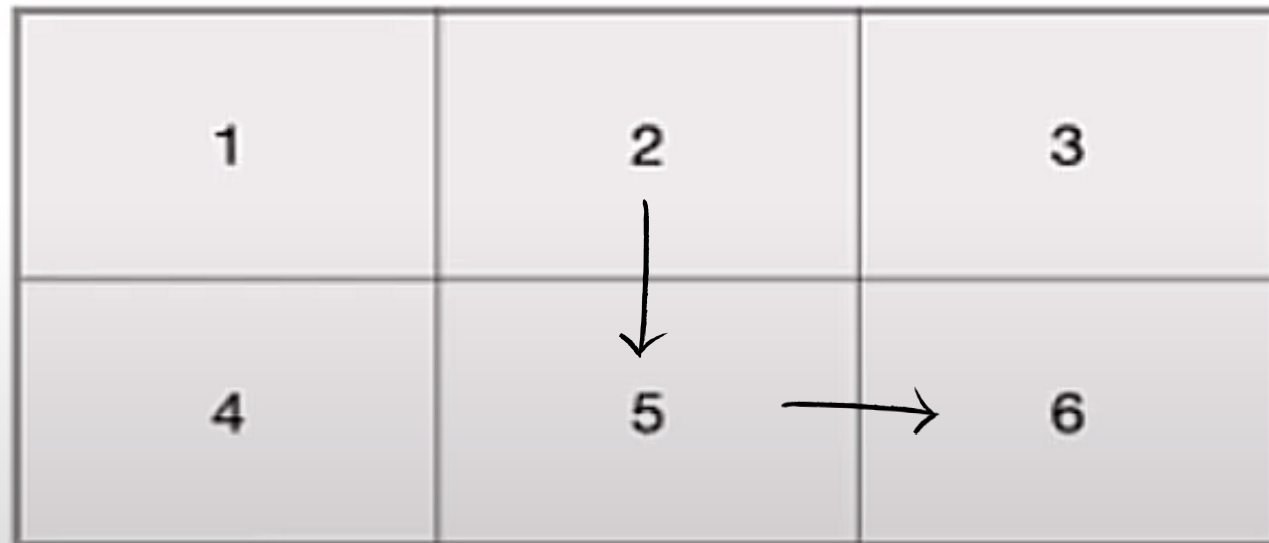
$Q(1, \text{down})=4$, $Q(1, \text{right})=6$

$Q(2, \text{left})=2$, $Q(2, \text{right})=4$, $Q(2, \text{down})=3$

$Q(3, \text{left})=5$, $Q(3, \text{down})=7$

$Q(4, \text{right})=4$, $Q(4, \text{up})=8$

$Q(5, \text{left})=5$, $Q(5, \text{right})=6$, $Q(5, \text{up})=8$



$\alpha = 0.25$; $\gamma = 0.75$

State/Action	1	2	3	4	5	6 (Final)
Up				8	8	
Down	4	3.6875	7			
Left		2	5		5	
Right	6	4		4	7.375	

Using $Q(s, a) = Q(s, a) + \alpha[R(s, a) + \gamma(Q'(s', a') - Q(s, a))]$

Say $R(s, a) = \begin{cases} 10, & \text{if } s = 6 \\ -1, & \text{otherwise} \end{cases}$ $Q'(s', a') = \text{maximum value of } s'$

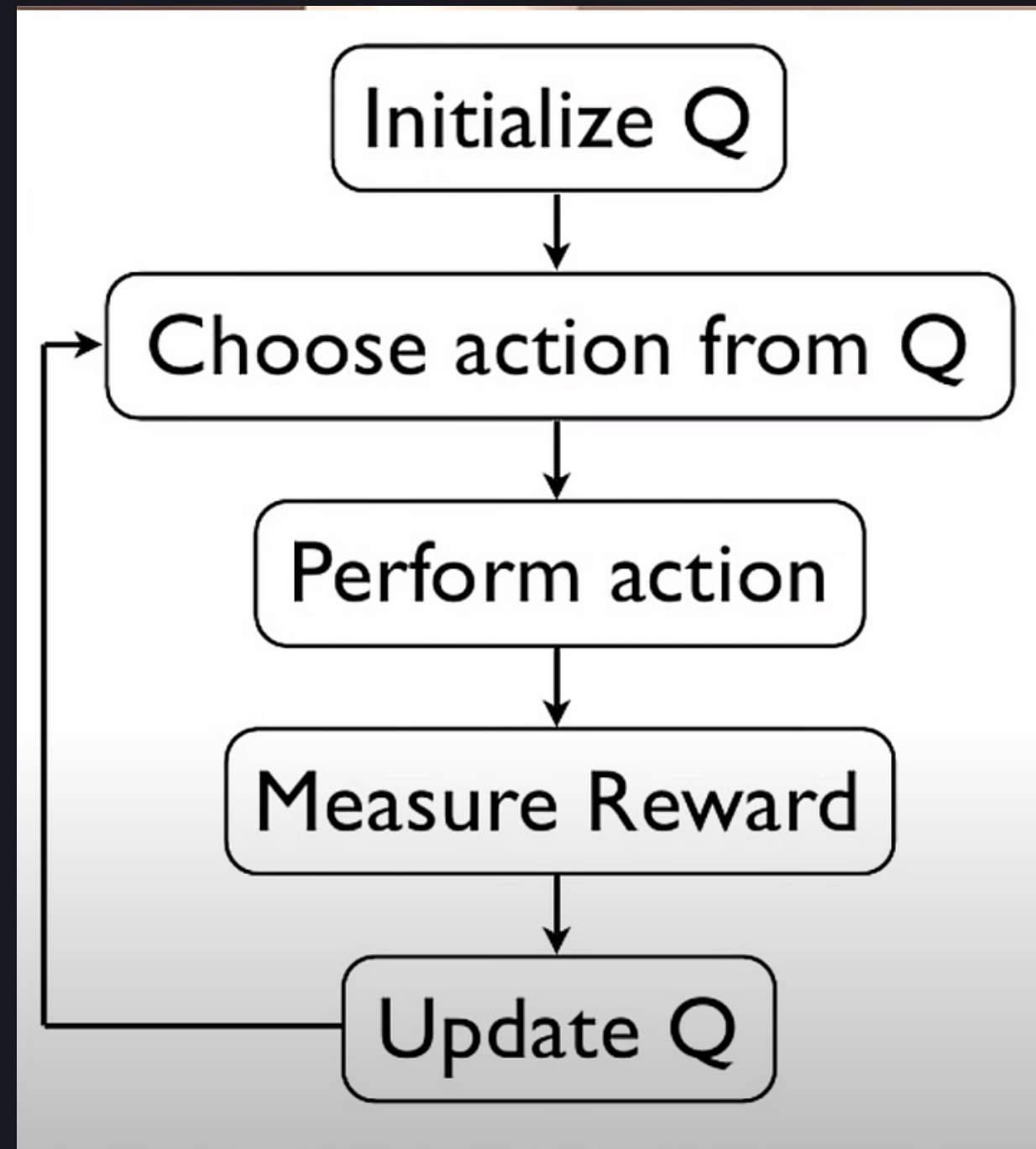
Then

$$Q(2, \text{down}) = 3 + 0.25 * [-1 + 0.75(8 - 3)] = 3.6875$$

$$Q(5, \text{right}) = 6 + 0.25 * [10 + 0.75(0 - 6)] = 7.375$$

Theory

Q-LEARNING



Theory

Q-LEARNING

Exploration vs Exploitation

Exploration: Gathering new information about the environment by performing actions that do not necessarily give highest reward. This is done by selecting actions with some level of randomness, rather than always selecting the action with the highest Q-value

Exploitation: using the knowledge or resources that have already been acquired in order to obtain the best possible outcomes. It involves exploiting known options and being confident about the outcome.

Theory

Q-LEARNING

When will it fail?

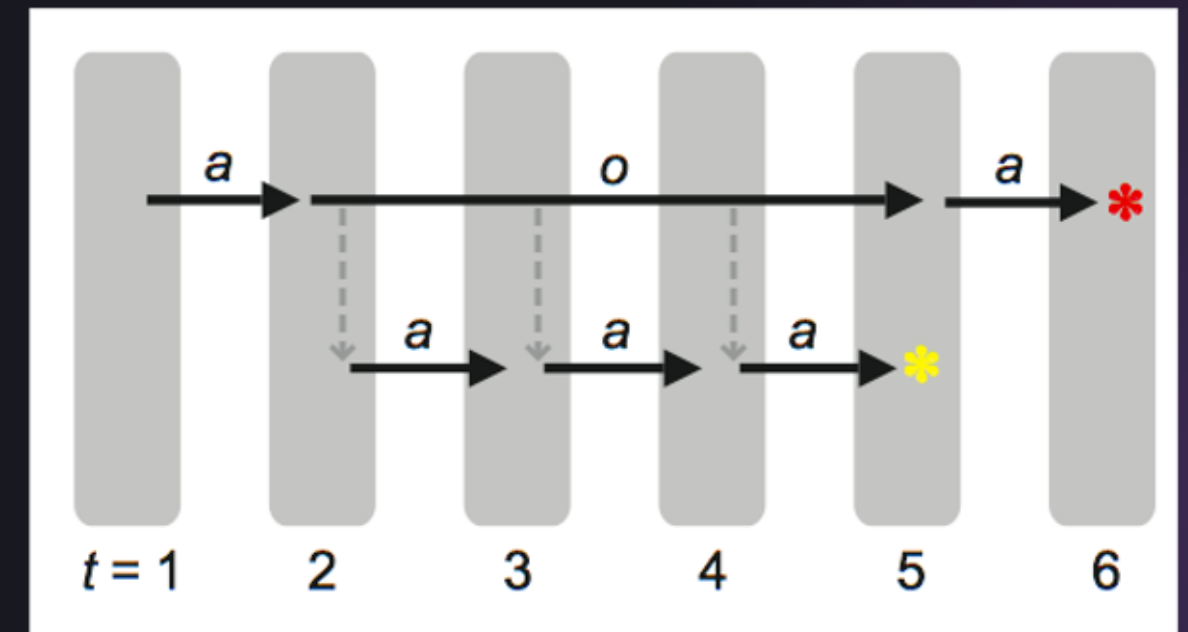
- 1.State space: Large / continuous / high dimensional
- 2.Action space: Large / continuous / high dimensional
- 3.Environment : Dynamic: transition probabilities and rewards change over time
- 4.Rewards: sparse rewards

Theory

HIERARCHICAL RL

What exactly is HRL

- Hierarchical reinforcement learning breaks down a complex task into smaller sub-tasks
- It uses a reward-based learning system to maximize cumulative reward over time
- The hierarchy involves multiple time scales for planning and executing long-term and short-term strategies



Theory

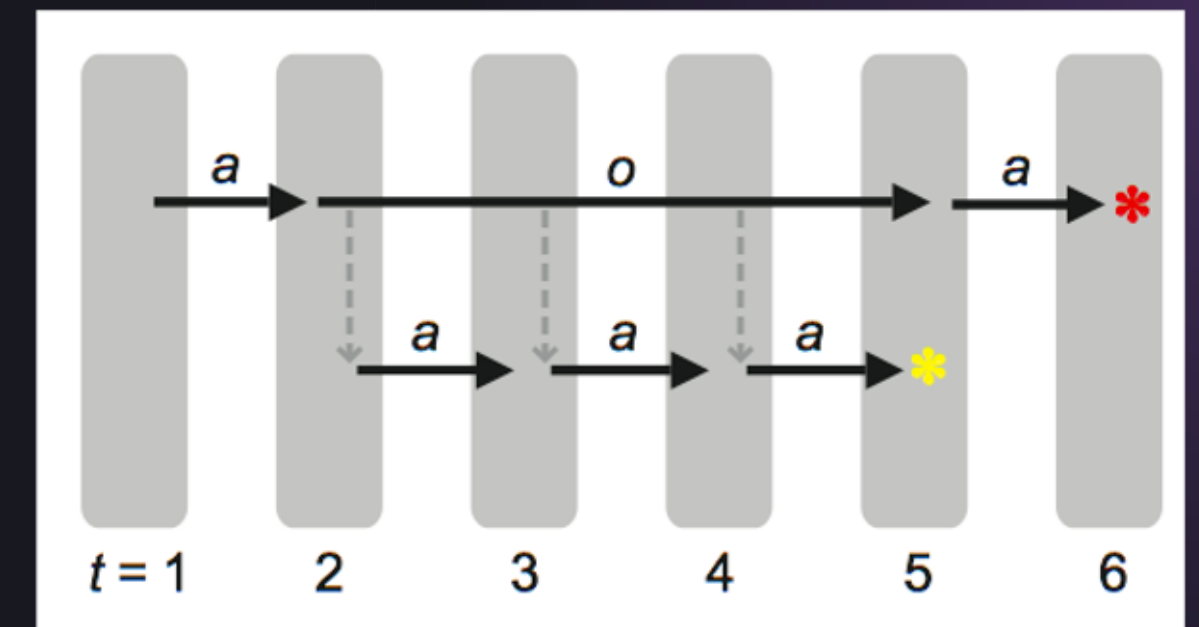
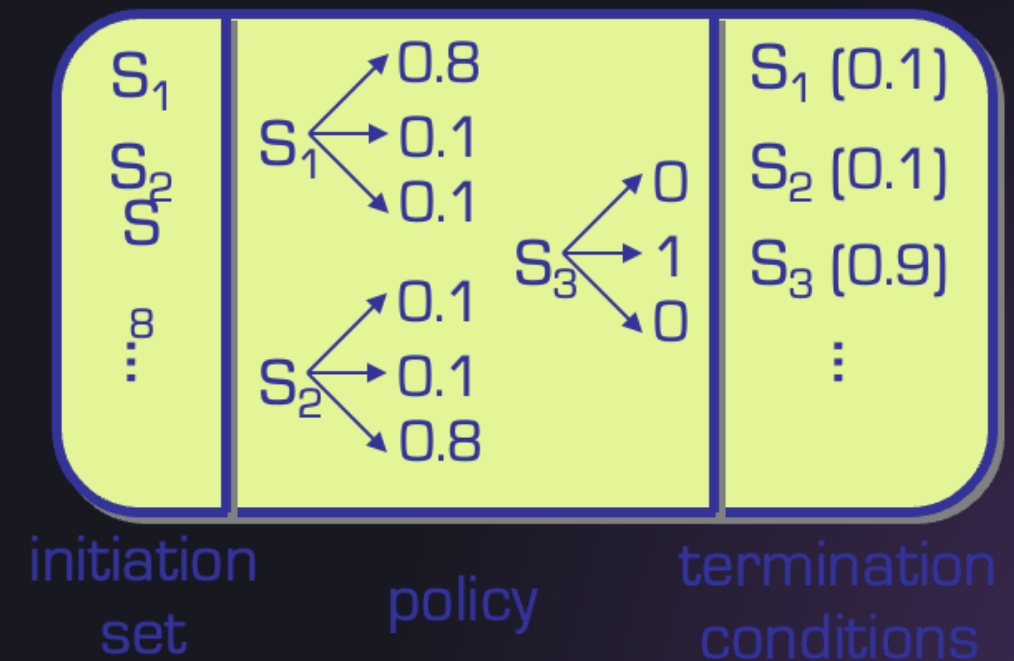
HIERARCHICAL RL

Takes advantage of options

In reinforcement learning, an option is a sub-policy or a sub-routine that an agent can choose to follow to achieve a particular goal.

Options are used in HRL to break down a complex task into smaller sub-tasks that the agent can learn and execute.

Each option is defined by initiation set, policy and termination condition.



Theory

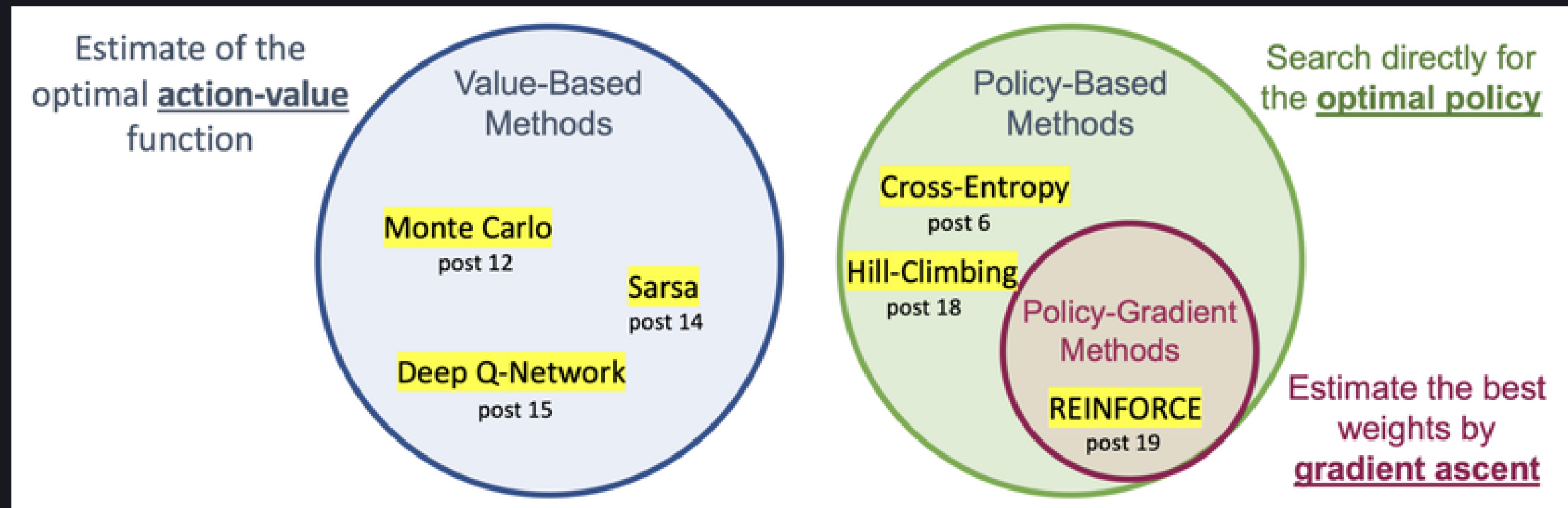
HIERARCHICAL RL

Actor-Critic

Actor-Critic

Estimates action
probability

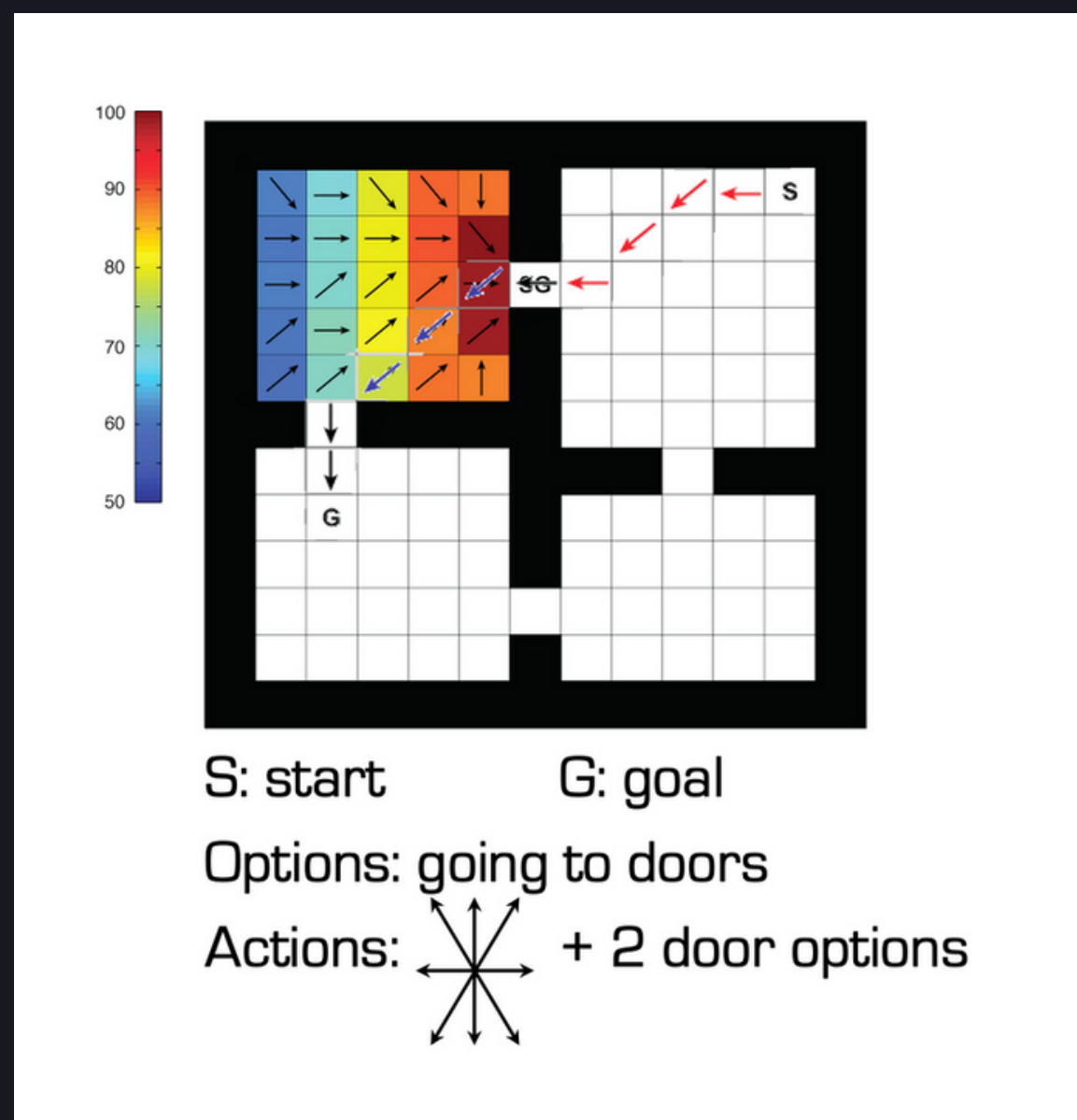
Estimates value
for action



Theory

HIERARCHICAL RL

Toy problem



Theory

HIERARCHICAL RL

Advantages and Disadvantages of RL

Advantage:

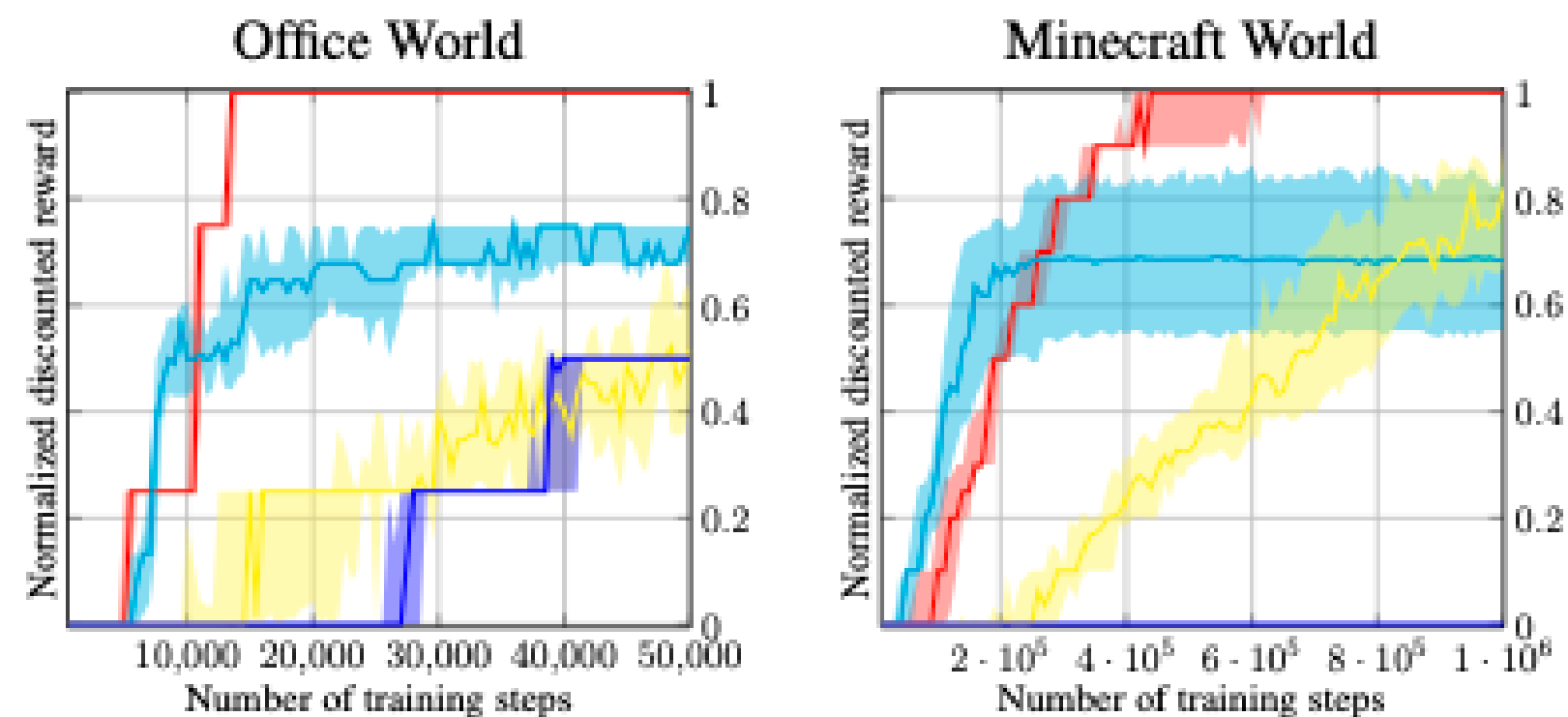
- Faster learning
- Transfer of knowledge from previous tasks

Disadvantage:

- Need 'right' options – how to learn them?
- Suboptimal behavior (“negative transfer”; habits)
- More complex learning/control structure

Theory

AND NEW METHODS ARE STILL BEING CREATED



(a) Performance over 4 tasks. (b) Performance over 10 tasks.

Legend: — Q-Learning — HRL — HRL-RM — QRM



SEE YOU NEXT TIME