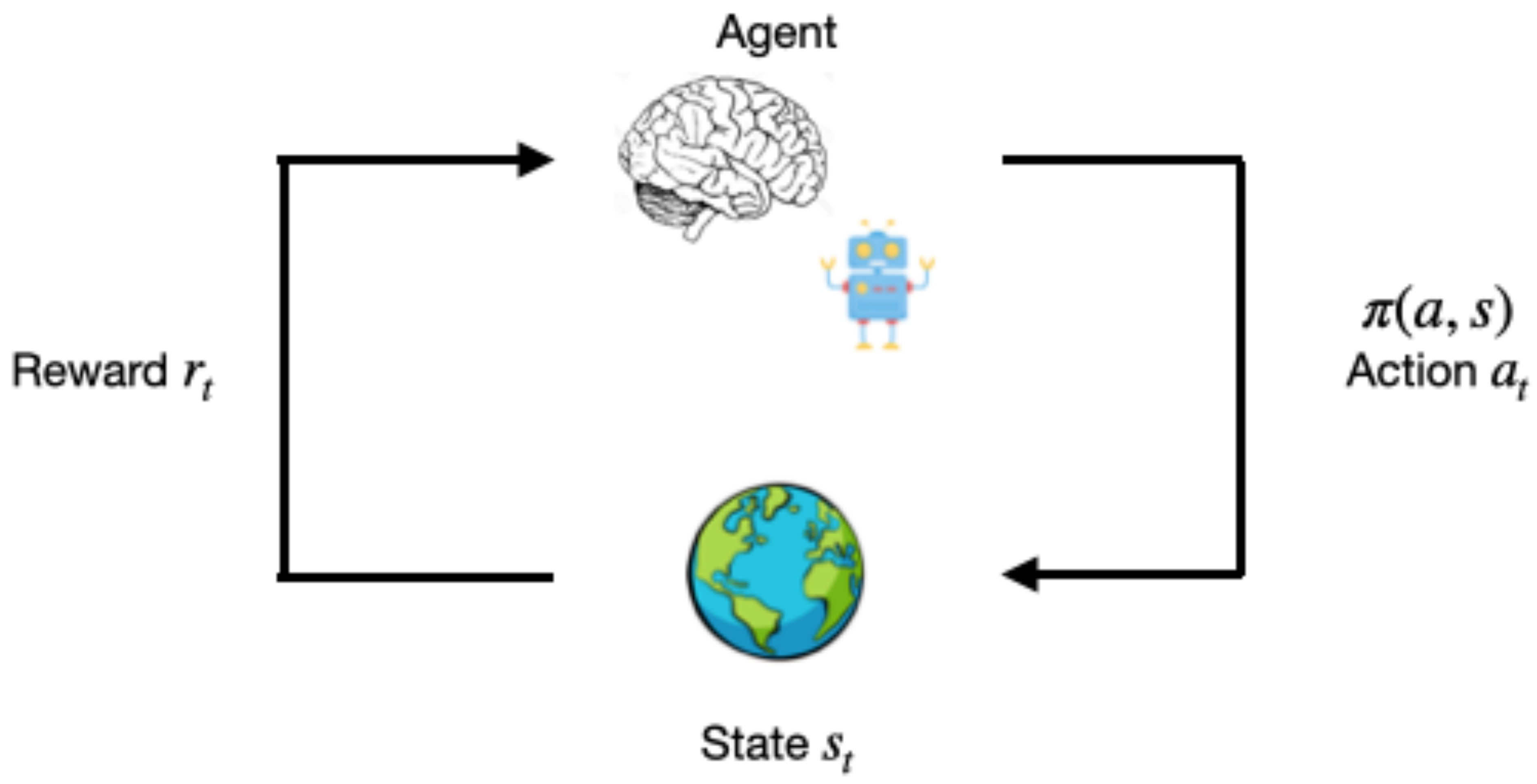
An introduction to Reinforcement Learning

12th of July 2022

Recap Q-Learning



TD Learning:

Prediction error

$$V(s_t) \leftarrow V(s_t) + \alpha \cdot (r + \gamma \cdot V(s_{t+1}) - V(s_t))$$

Learning rate Discount rate

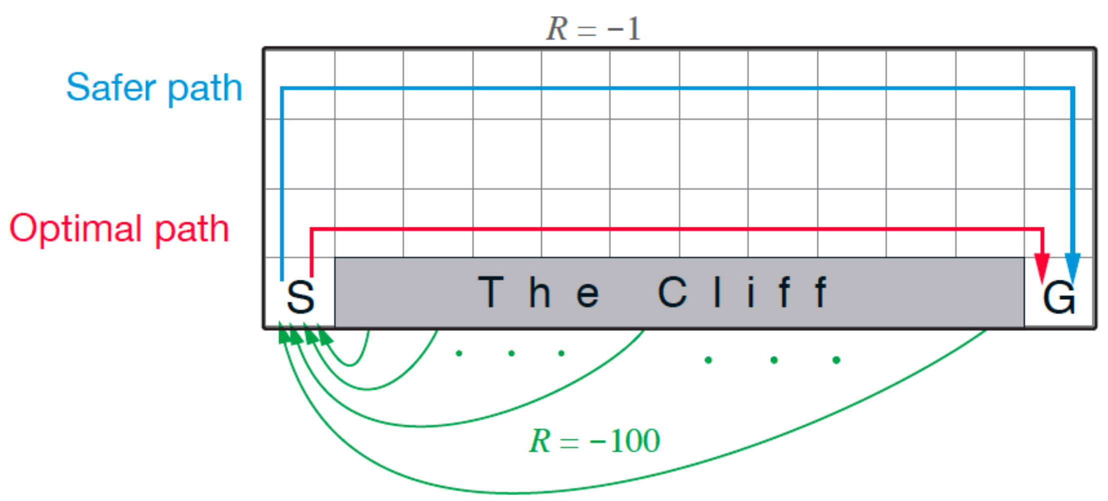
Q-Learning:

Prediction error

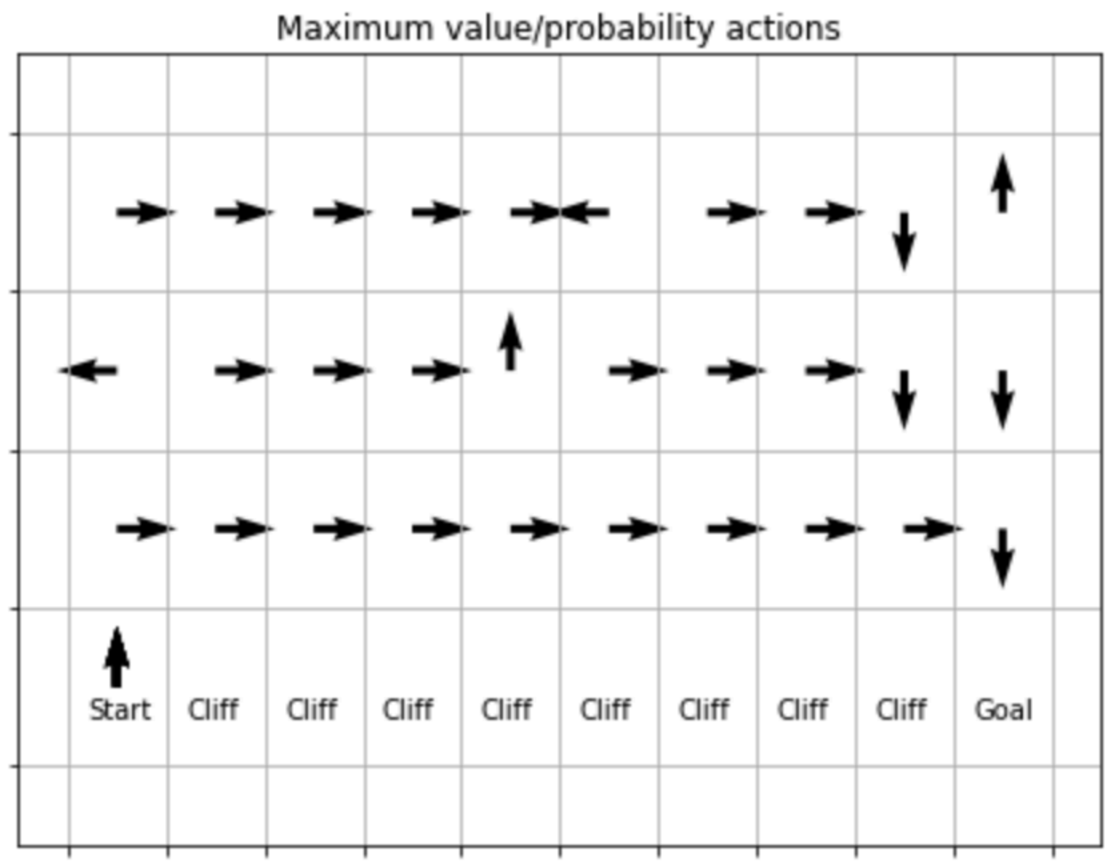
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot (r + \gamma \cdot max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

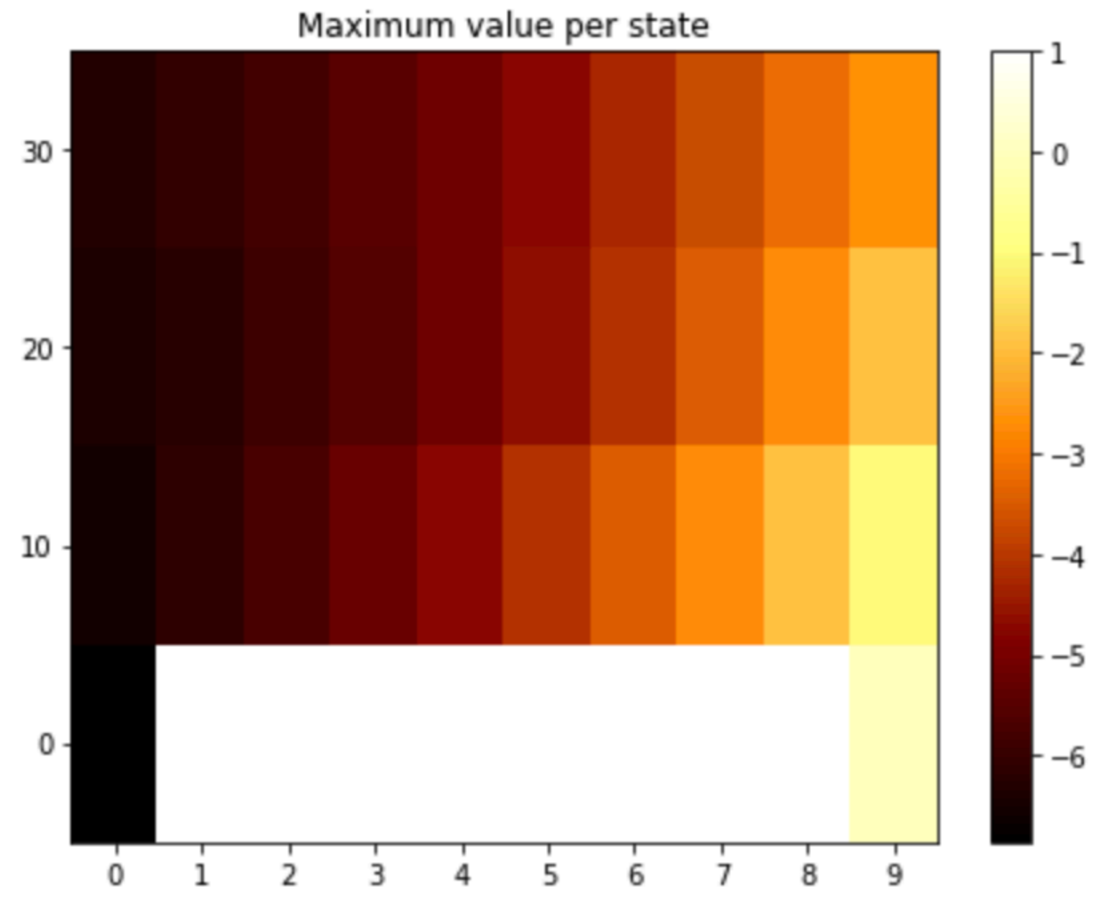
Learning rate Discount rate

Recap Q-Learning



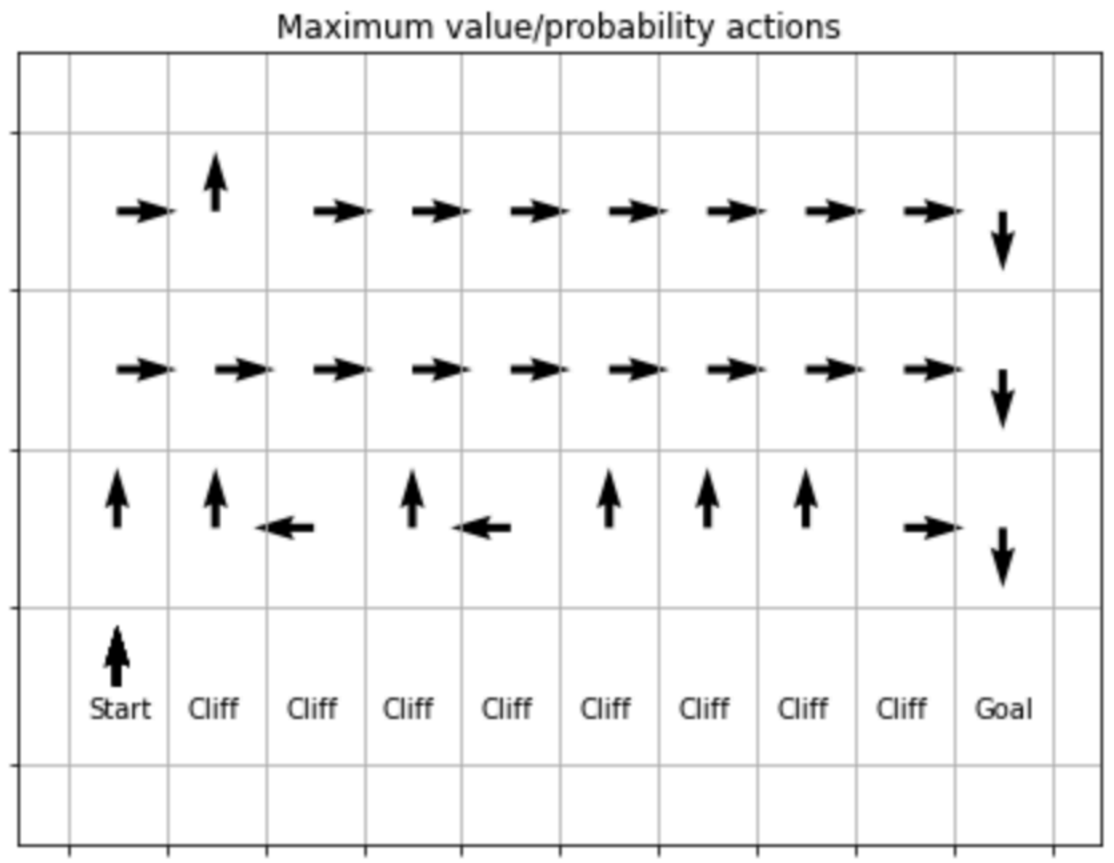
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big(r_t + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big)$$



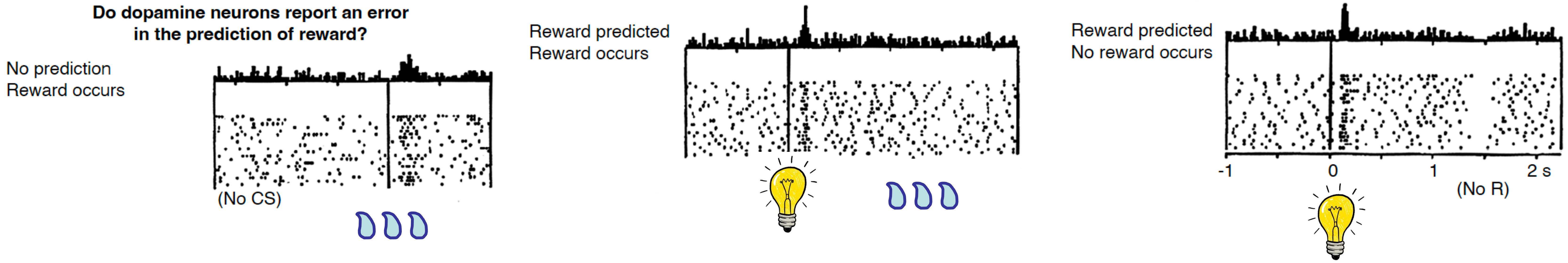


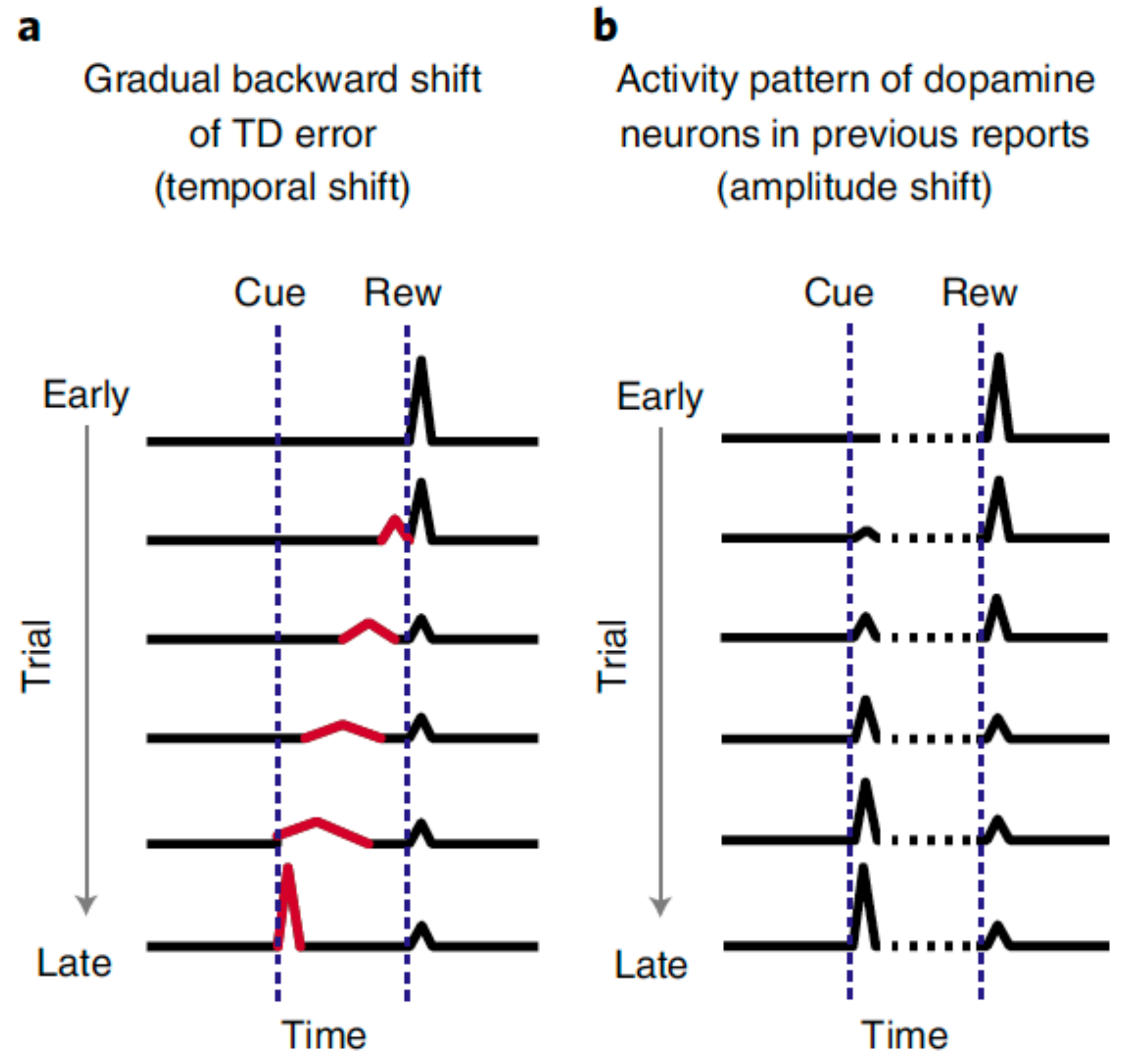
There are also alternatives:

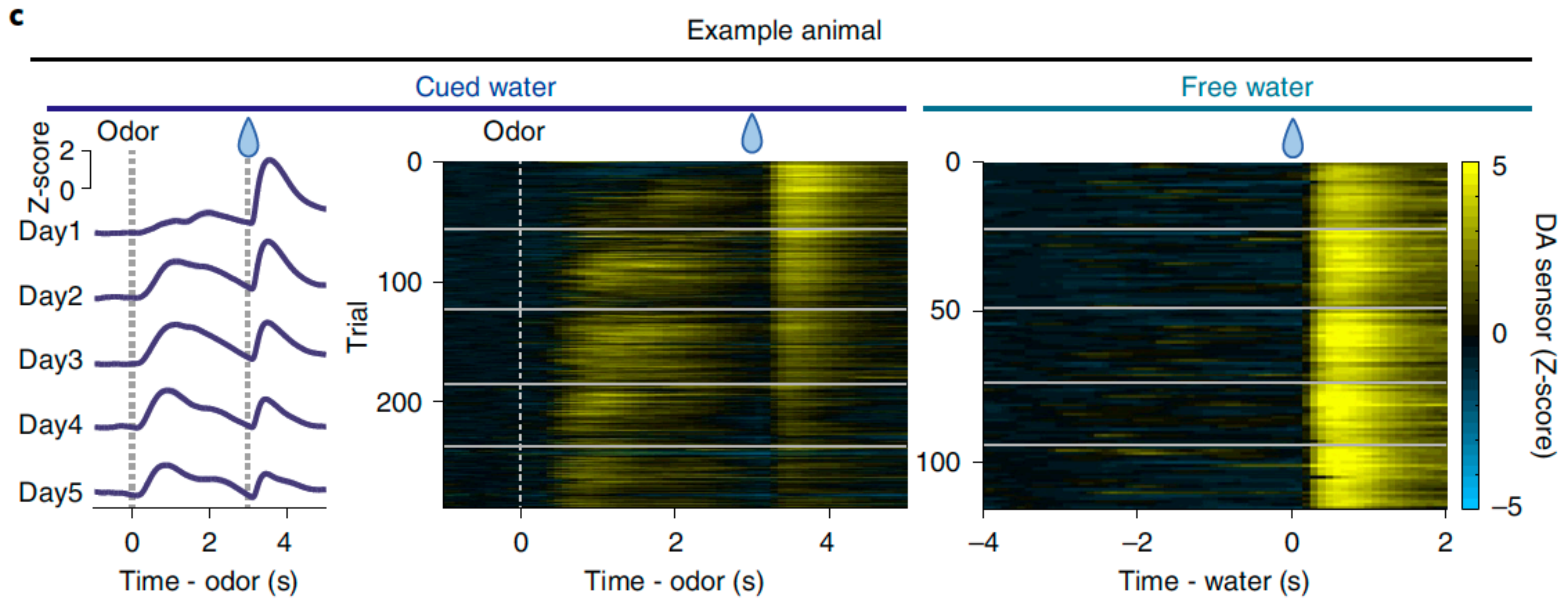
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big)$$



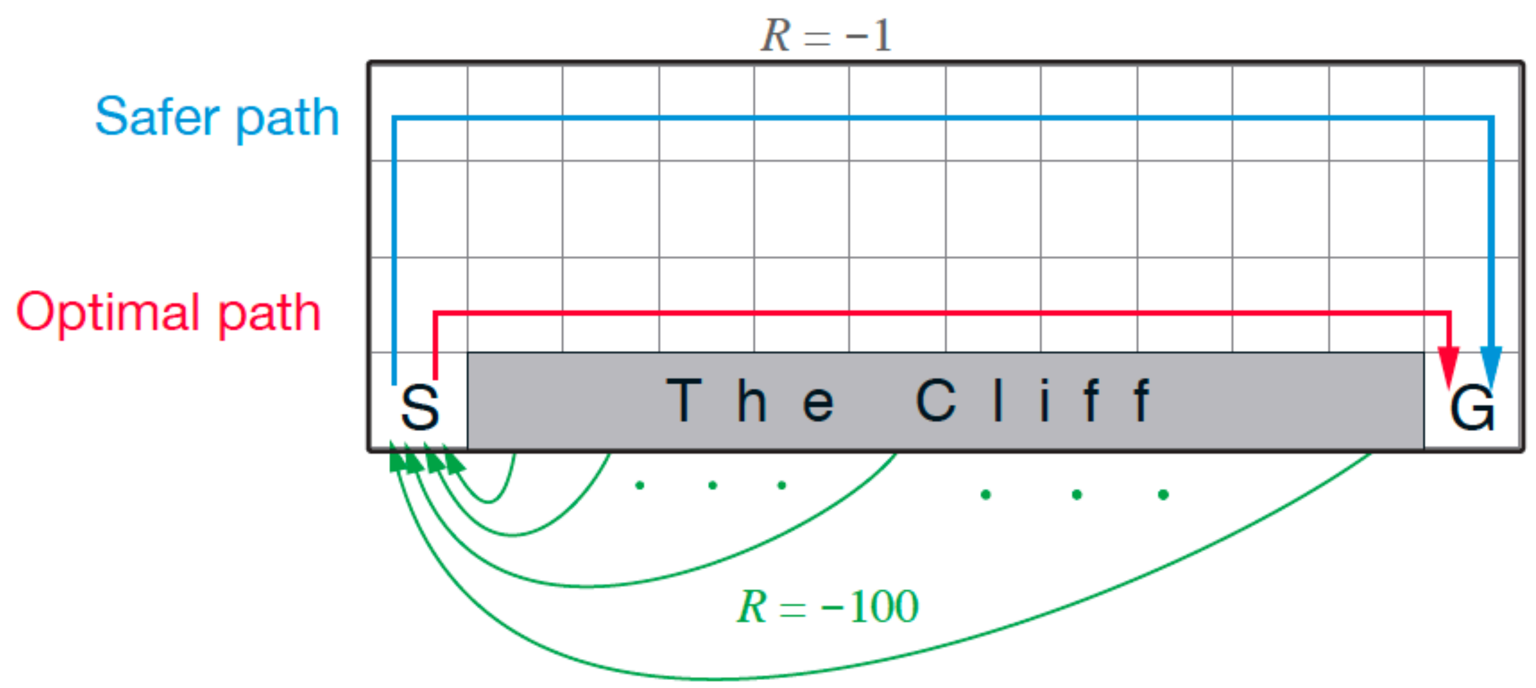
Q- (TD-) learning in action







Amo, …, Watabe-Uchida, Nature Neuroscience, 07 July 2022



MDPs and model-based RL

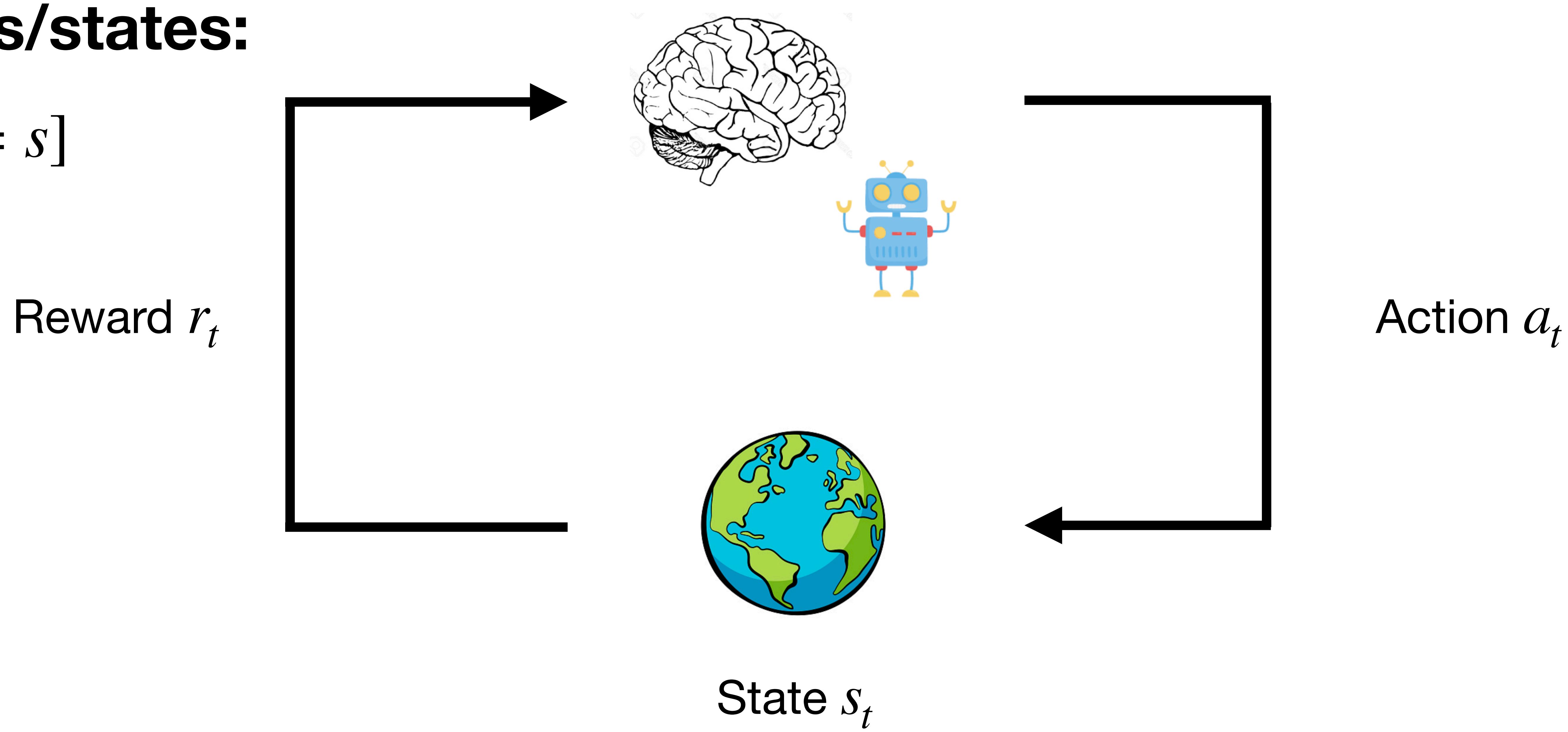
Basic setup: how do agents learn to act?

Based on a reward signal, agents learn **values of actions/states:**

$$V_\pi(s) = \mathbb{E}_\pi[R \mid s_0 = s]$$

Action is governed by a **policy**:

$$\pi(a, s) = P(a_t = a \mid s_t = s)$$



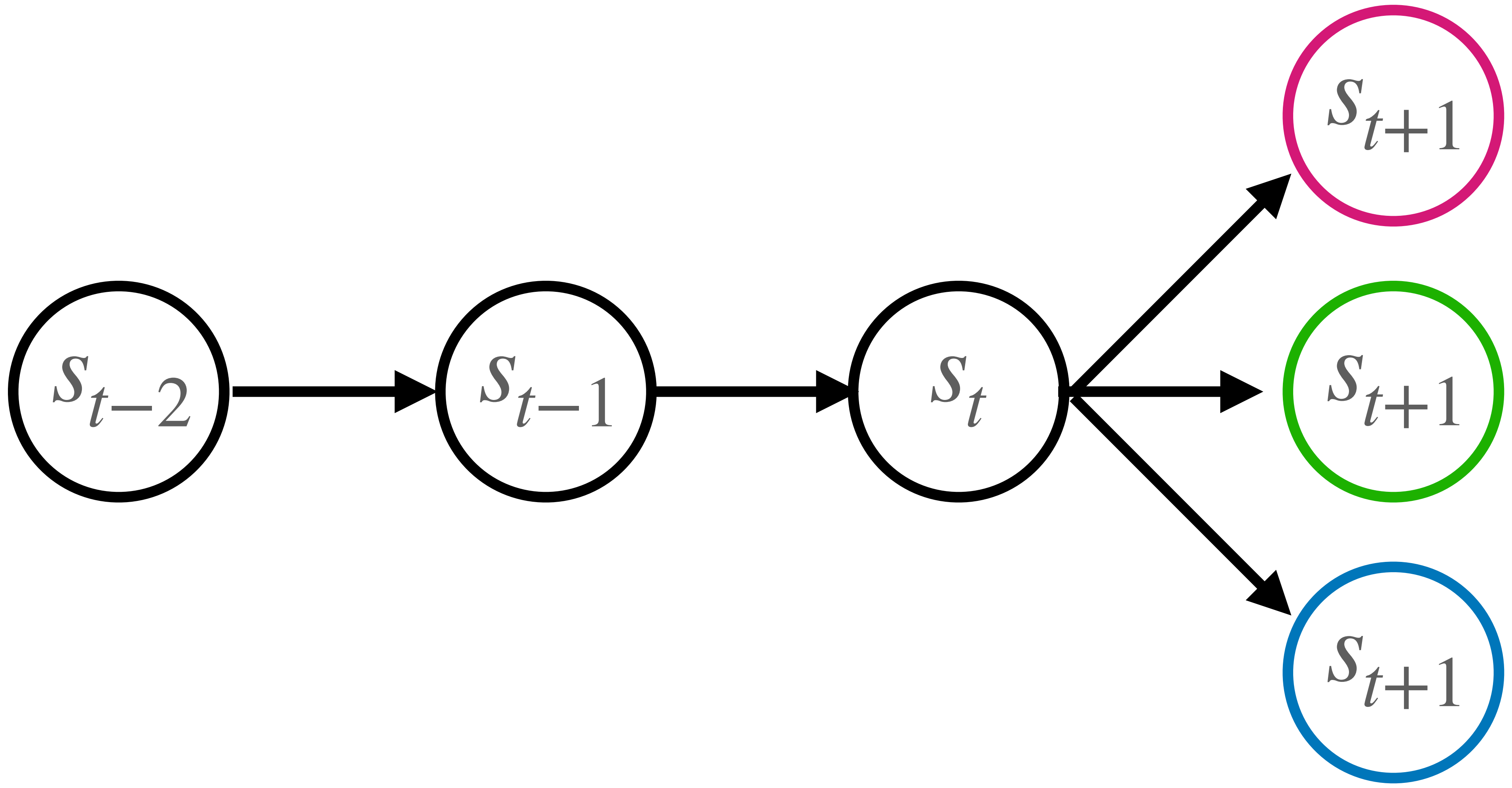
Agents can learn a **model of the environment** to make smarter decisions, e.g.:

$$P(s_{t+1} = s \mid s_t = s, a_t = a)$$

Markov Process ⟶ Markov Reward Process ⟶ **Markov Decision Process (MDP)**

Markov Process

Most RL problems are problems where agents face sequences of states:

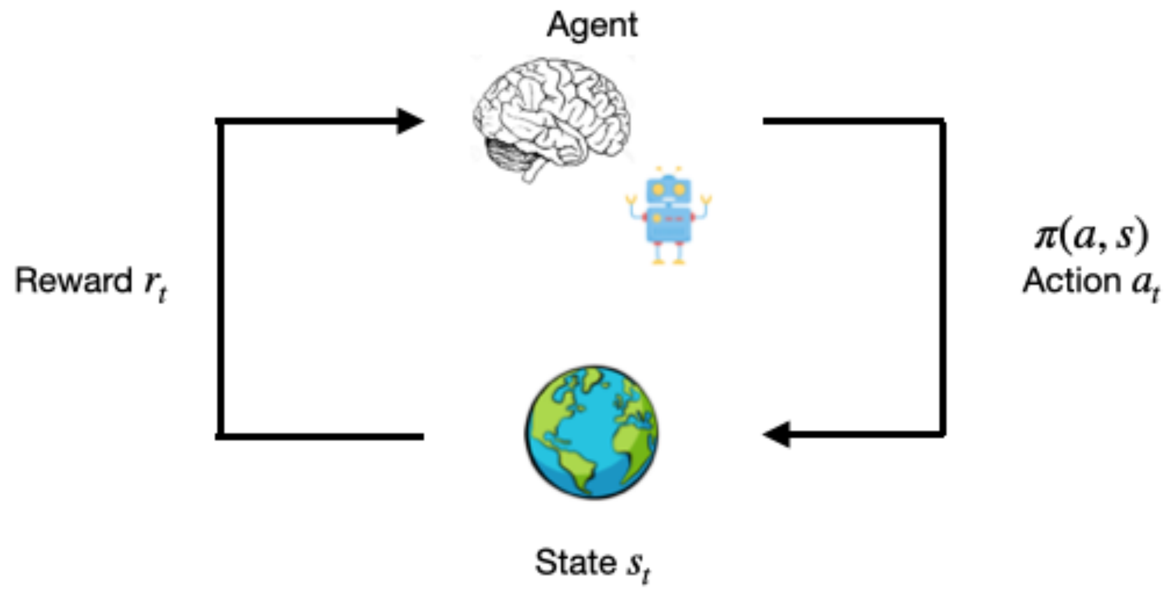


Fundamental property: **Markov property**

$$P(s_{t+1} = s \,|\, s_t, s_{t-1}, s_{t-2}, \ldots) = P(s_{t+1} = s \,|\, s_t)$$

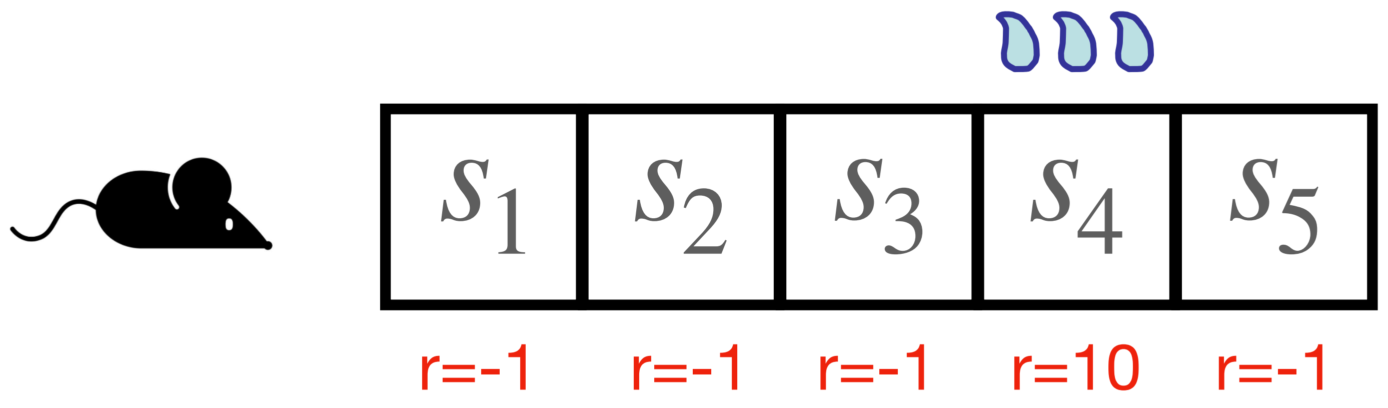
"The future is independent of the past given the present"

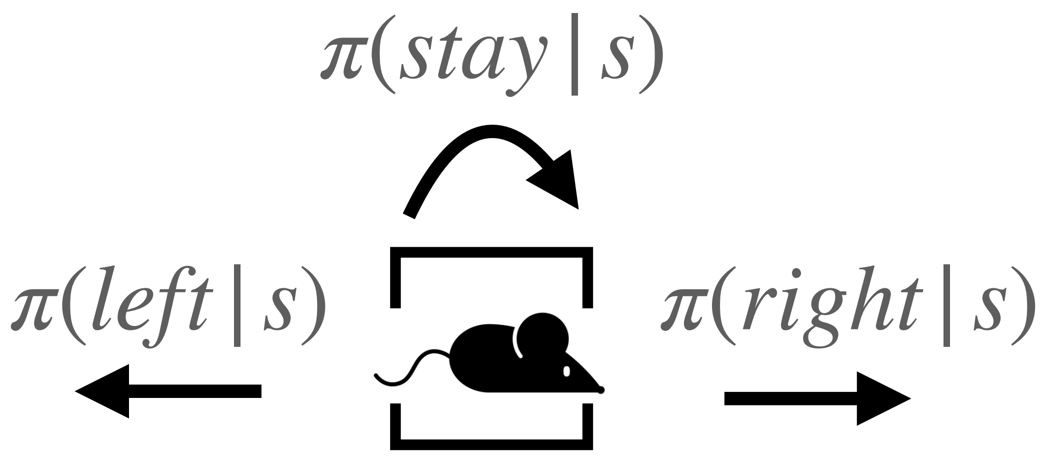
Markov Decision Process



A **Markov Decision Process** is defined based on

- A State Space S
- An **Action Space** A
- Transition Probabilities P
- A Reward Function $R_s = \mathbb{E}[r_t \mid s_t = s]$
- A Discount Factor $\gamma \in [0,1]$



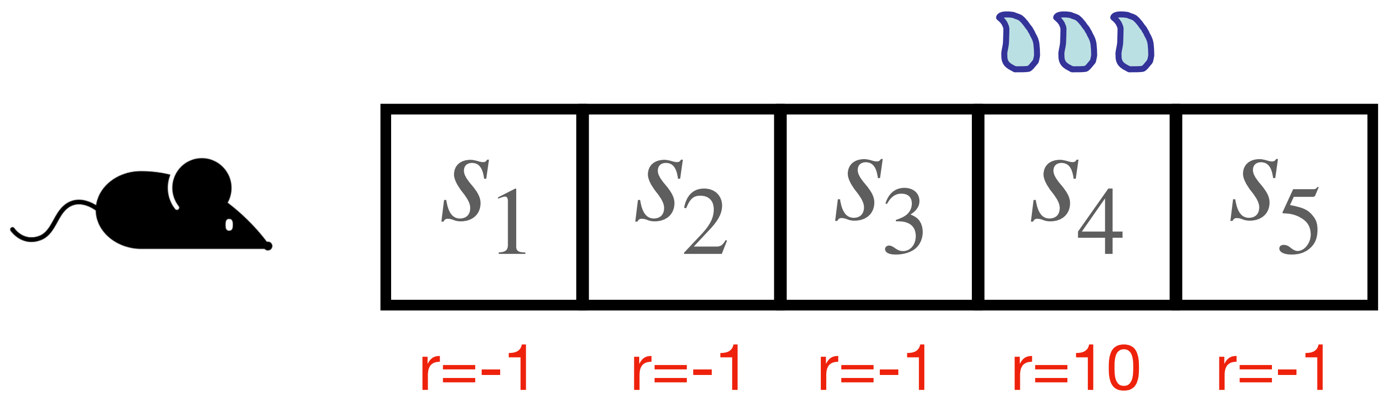


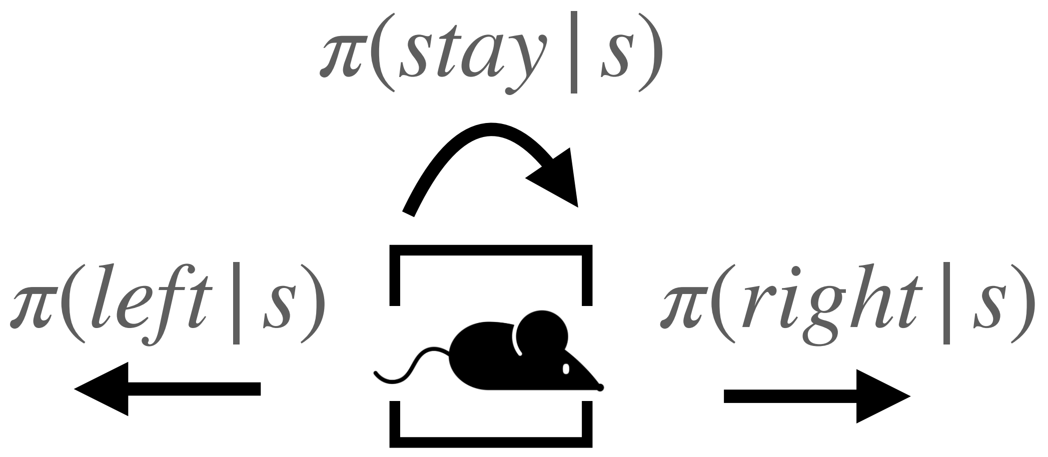
Actions are governed via a **policy**: $\pi(a,s) = P(a_t = a \mid s_t = s)$

MDPs basis for model-based RL

Allows to specify all environment dynamics for RL problem:

$$P(s', r \mid s, a) = P(s_{t+1} = s', r_{t+1} = r \mid s_t = s, a_t = a)$$





MDPs basis for model-based RL

$$P(s', r \mid s, a) = P(s_{t+1} = s', r_{t+1} = r \mid s_t = s, a_t = a)$$

How can we make use of such models of the world?

Planning and **action selection** (maybe later..)

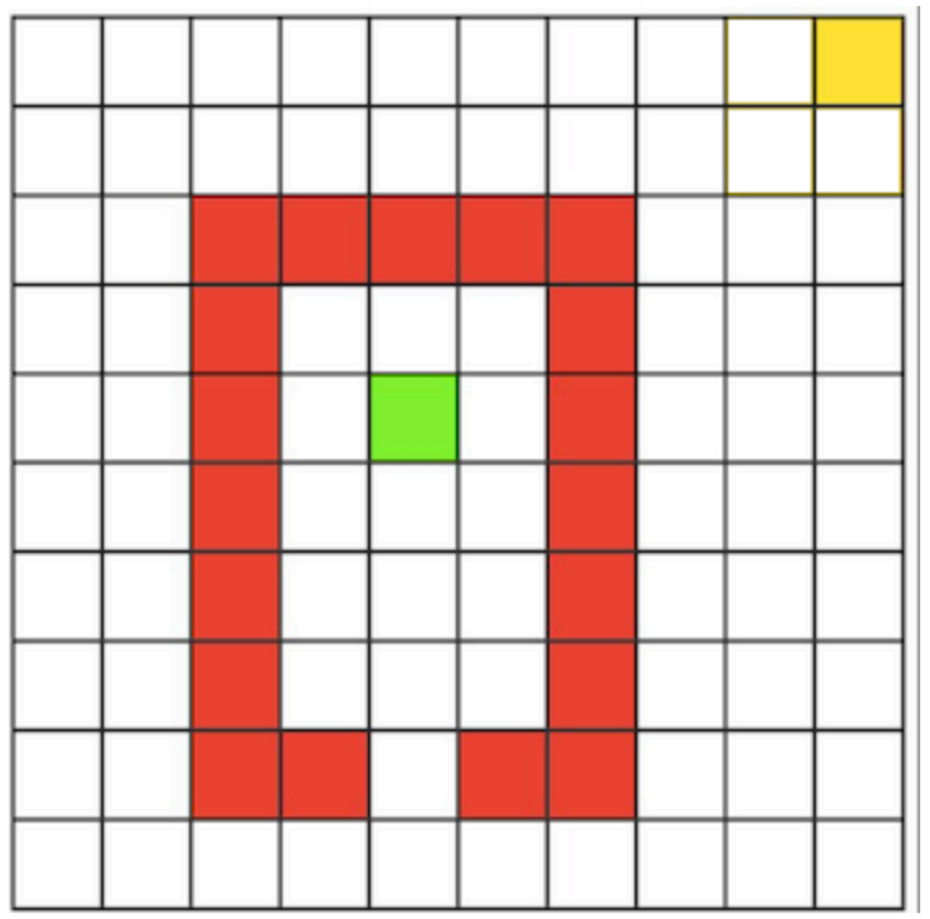
Learning

- Key idea: store experiences in world model $P(s', r \mid s, a)$
- Sample from this model to generate extra learning data
- This is called **DYNA-Q…**

Coding: DYNA-Q

https://github.com/schwartenbeckph/RL-Course/tree/main/2022_07_12

DYNA-Q



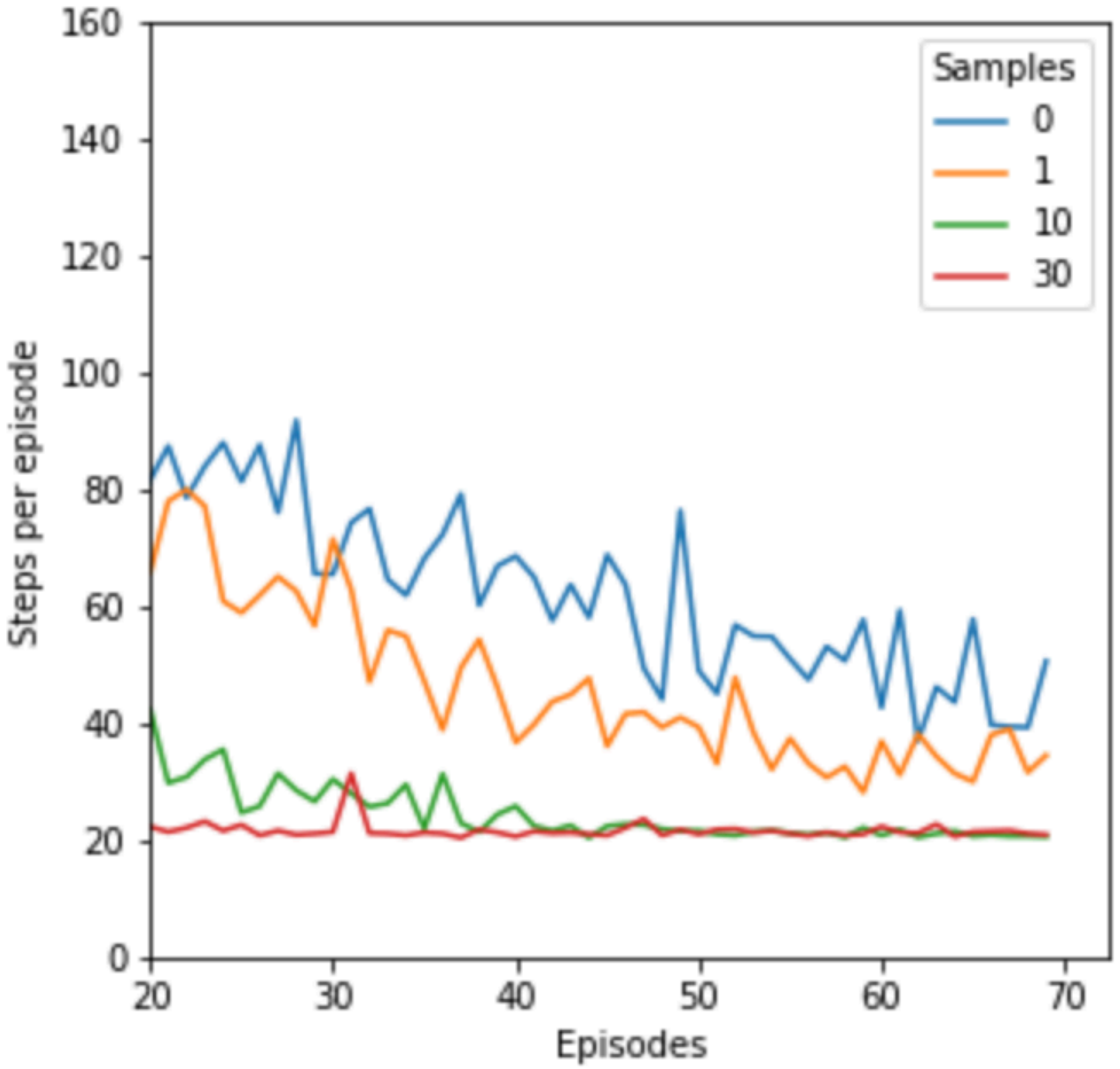
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big(r_t + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big)$$

$$P(s', r|s, a) = P(s_{t+1} = s', r_{t+1} = r|s_t = s, a_t = a)$$

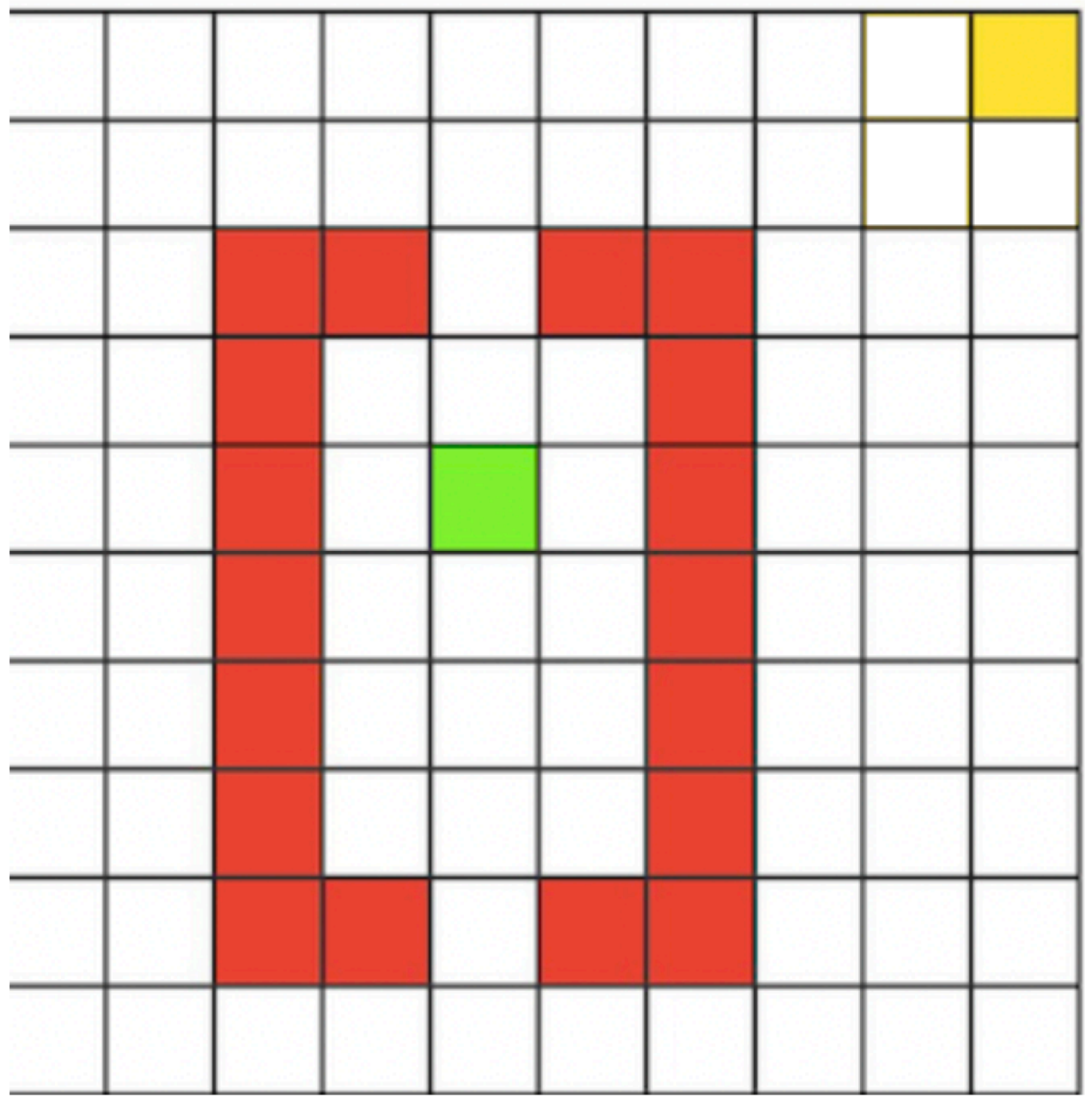
$$Model(S, A) \leftarrow R, S'$$

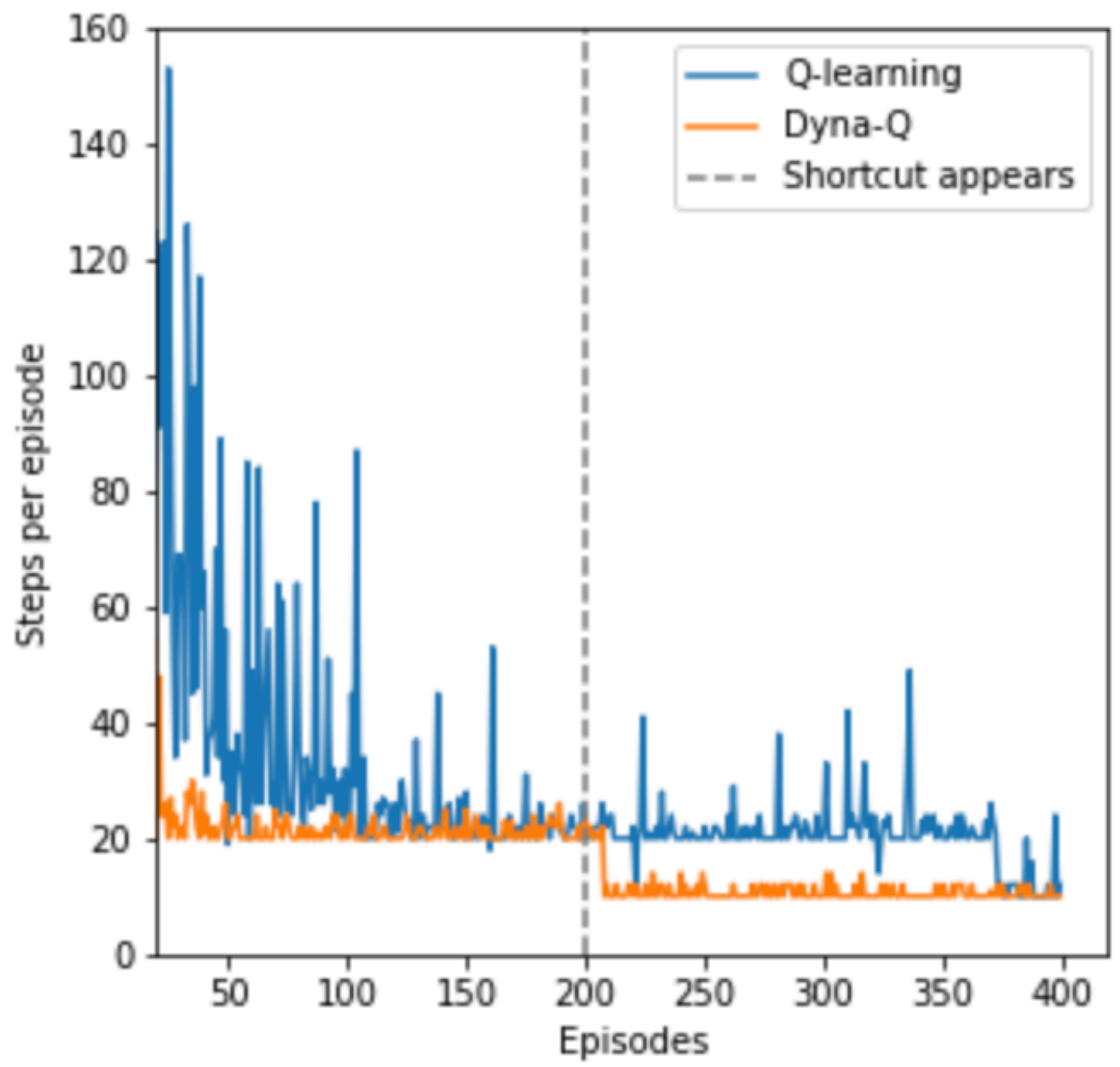
And during breaks ('offline rest'), they can sample from this experience and learn from these samples:

$S \leftarrow$ previously observed state
$A \leftarrow$ action previously taken in S
$R, S' \leftarrow Model(S, A)$
$Q(S, A) \leftarrow Q(S, A) + \alpha\left[R + \gamma \max_a Q(S', A) - Q(S, A)\right]$



DYNA-Q





Q-learning
Dyna-Q
Shortcut appears
Steps per episode
Episodes
0
20
40
60
80
100
120
140
160
50
100
150
200
250
300
350
400