

An introduction to Reinforcement Learning

05th of July 2022

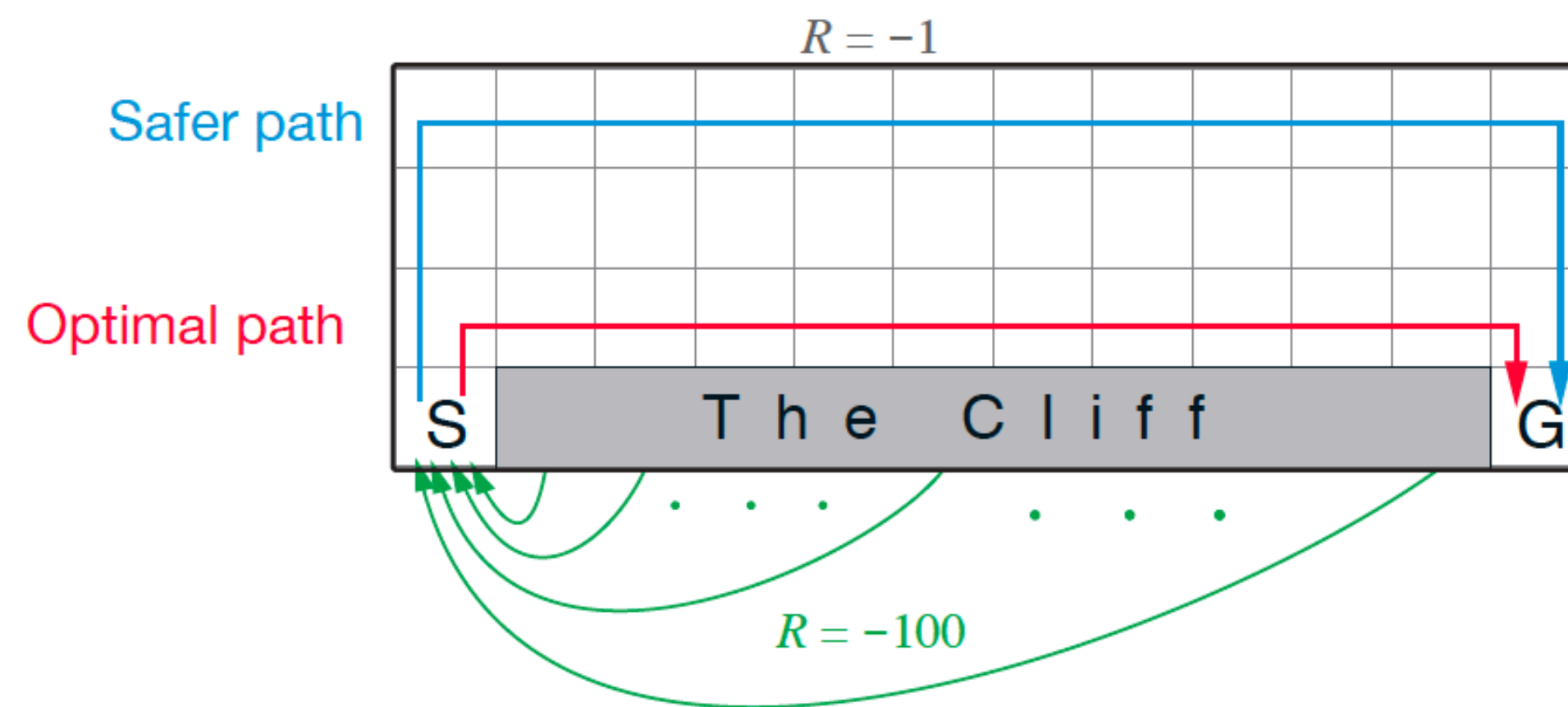
Q-Learning

Limitation of multi-armed bandit problems

Your current action does not influence what happens next!!

How can we solve sequential problems?

The textbook problem:
‘Cliff-World’



The rules:

- Agent has to move from start (S) to goal (G)
- Reaching the goal results in a positive reward of +10
- Falling off the cliff results in a negative reward of -100
- Any other state results in a negative reward of -1

What's the problem the agent has to solve here??

Note the subtle introduction of the concept of **‘transition probabilities’** here
- implicit, later: explicit

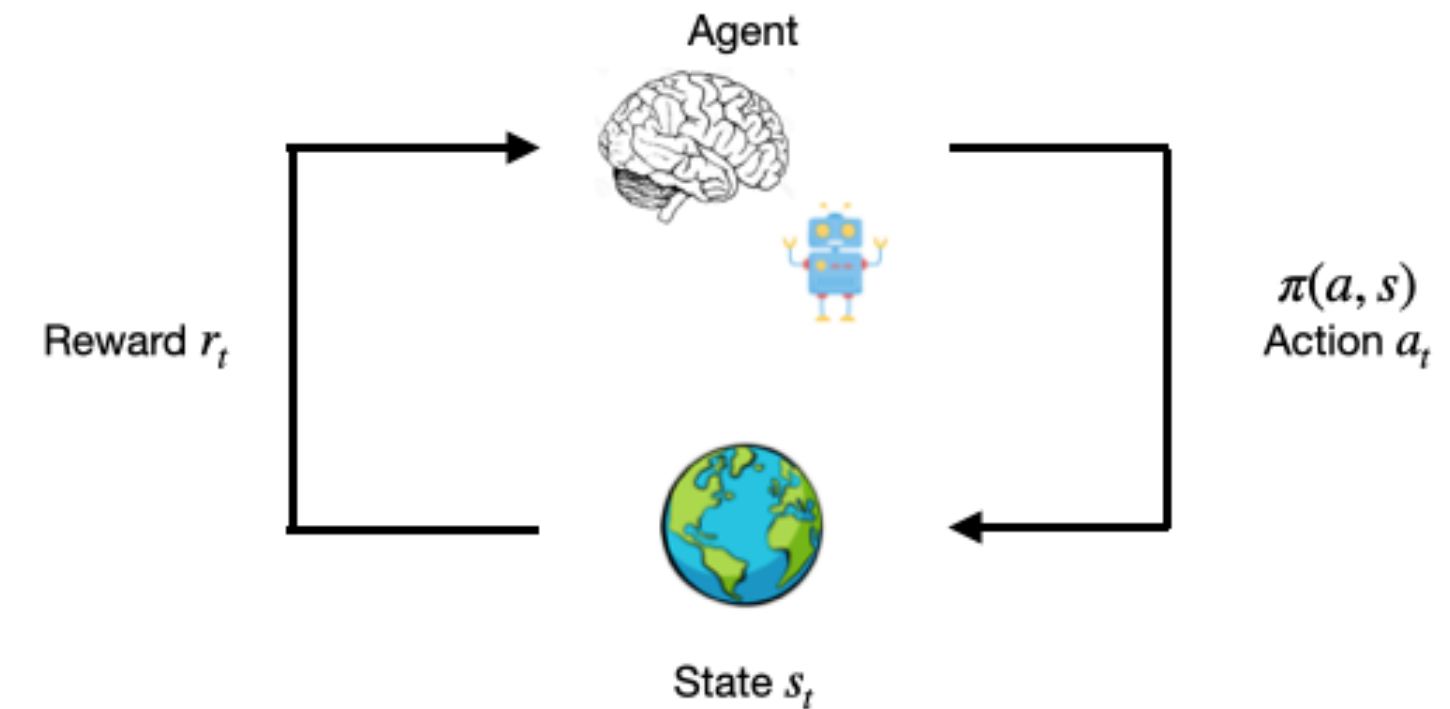
From classical to instrumental learning

TD Learning:

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

Learning rate Discount rate

Prediction error



Q-Learning:

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

Learning rate Discount rate

Prediction error

What's the difference between $V(s_t)$ and $Q(s_t, a_t)$?

What's is $\max_a Q(s_t, a_t)$ doing?

Note that this is just an update rule - doesn't tell us how to select an action!

Coding: Q-Learning

https://github.com/schwartenbeckph/RL-Course/tree/main/2022_07_05

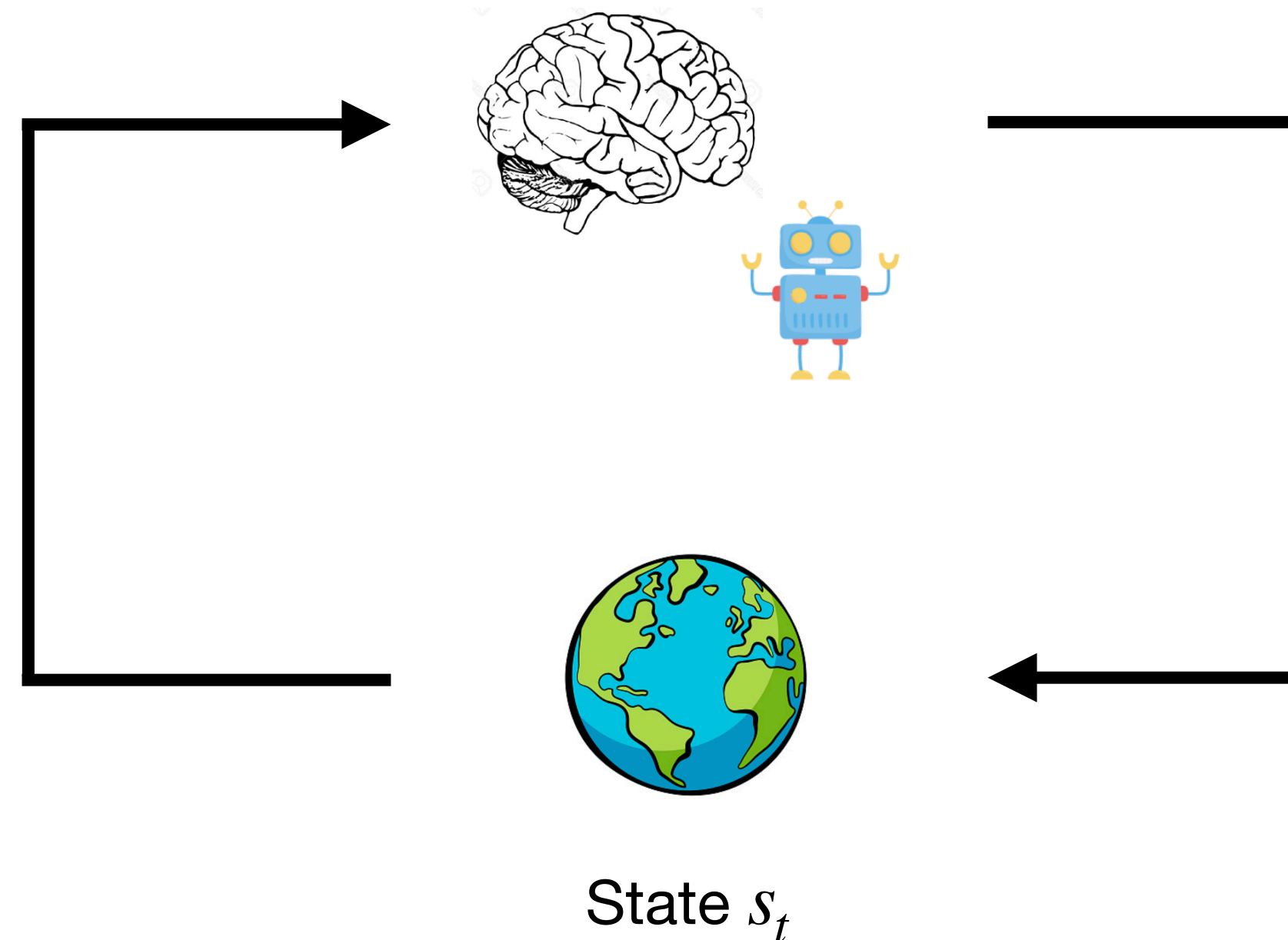
MDPs and model-based RL

Basic setup: how do agents learn to act?

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

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

Reward r_t



Action is governed by a **policy**:

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

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

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

Markov Process



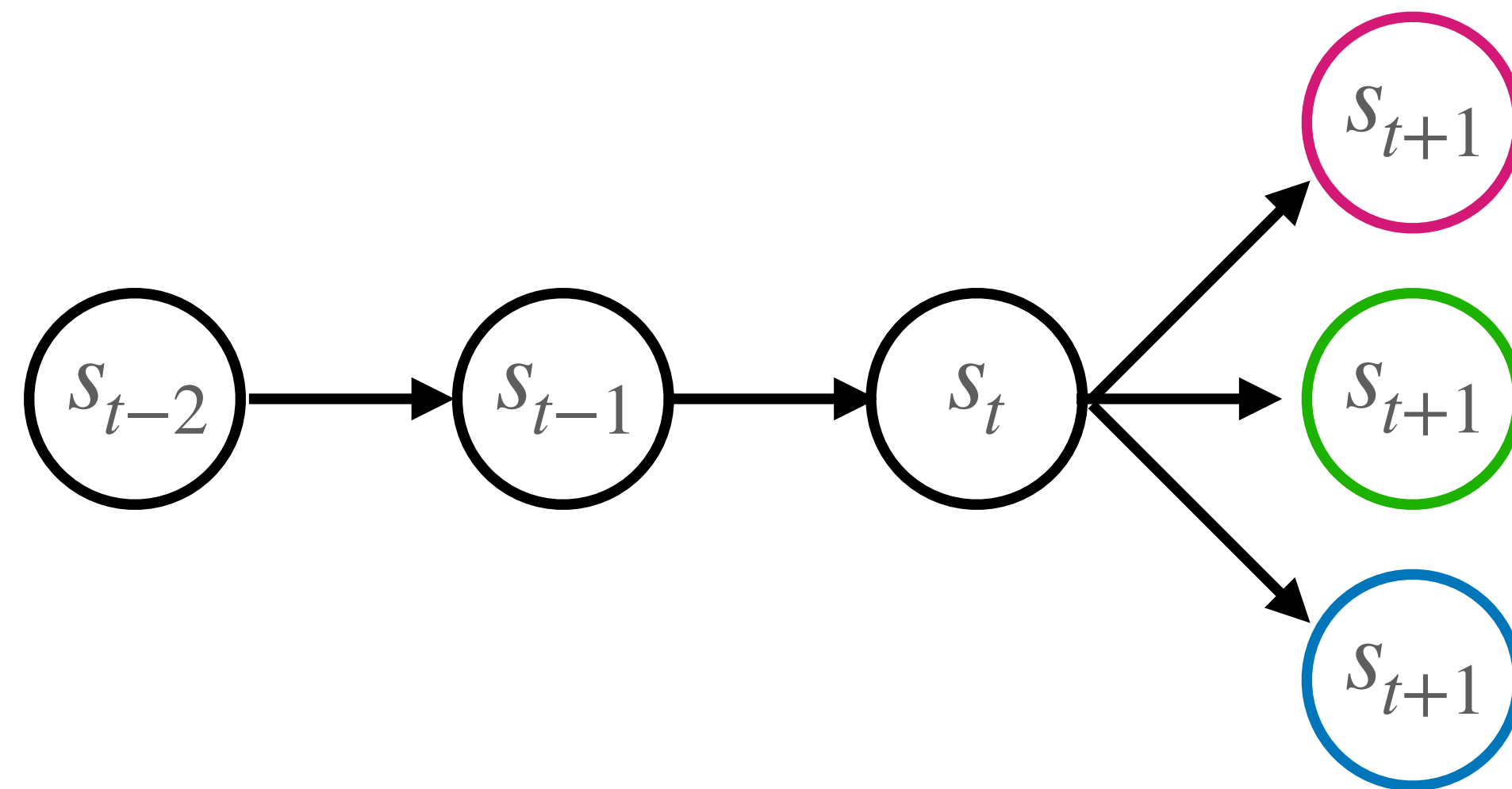
Markov Reward Process



Markov Decision Process (MDP)

Markov Process

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

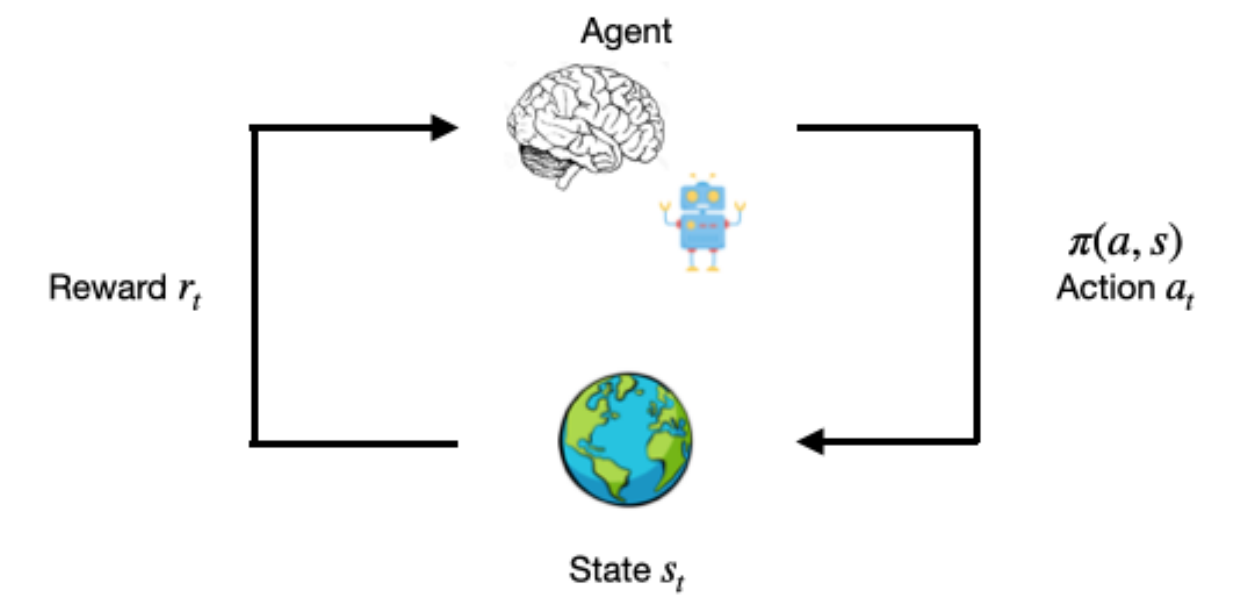


Fundamental property: **Markov property**

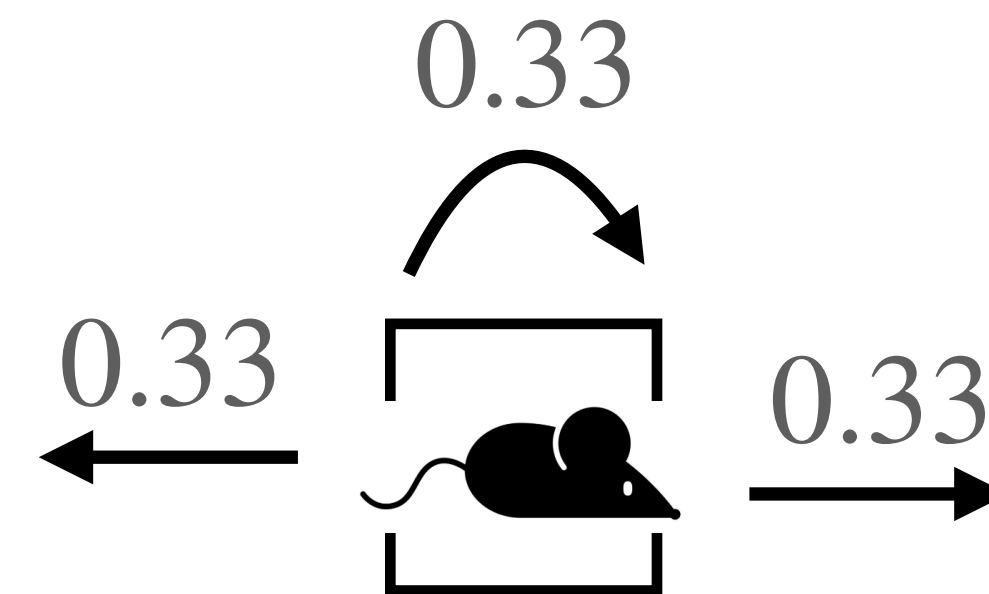
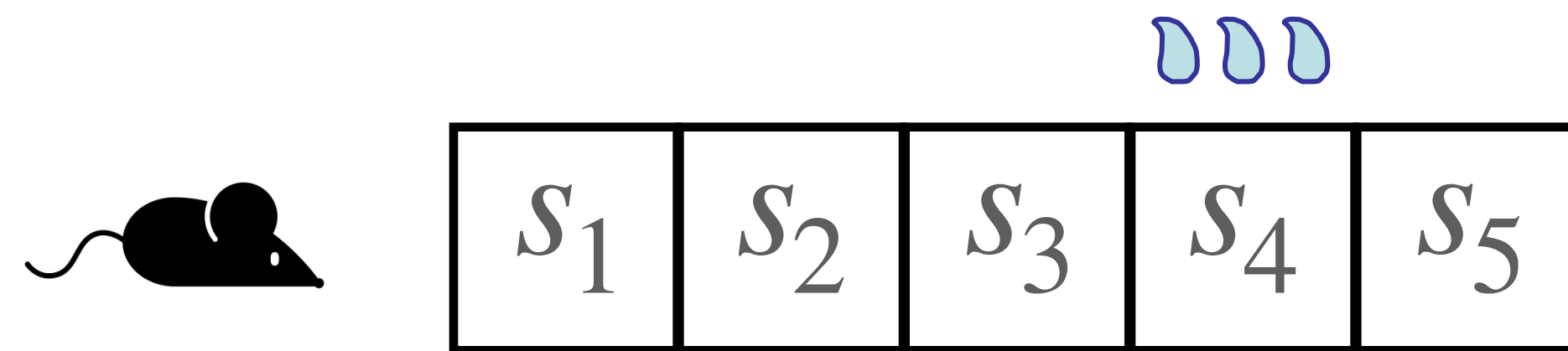
$$P(s_{t+1} = s \mid s_t, s_{t-1}, s_{t-2}, \dots) = P(s_{t+1} = s \mid s_t)$$

“The future is independent of the past given the present”

Markov Process

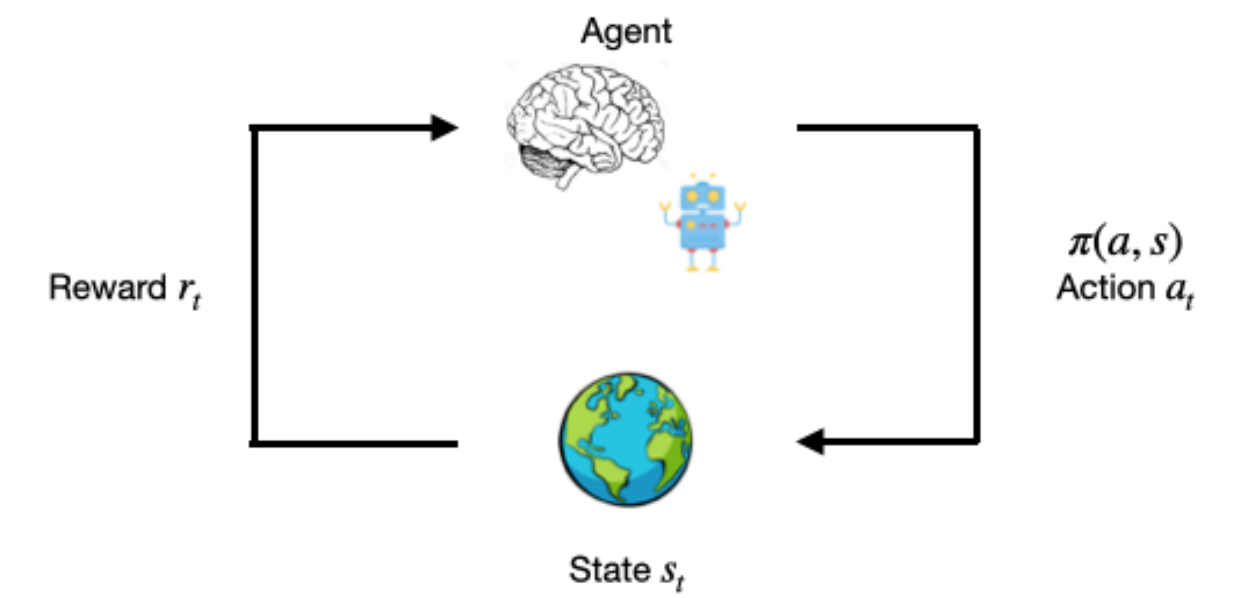


Let's assume a super simple problem:

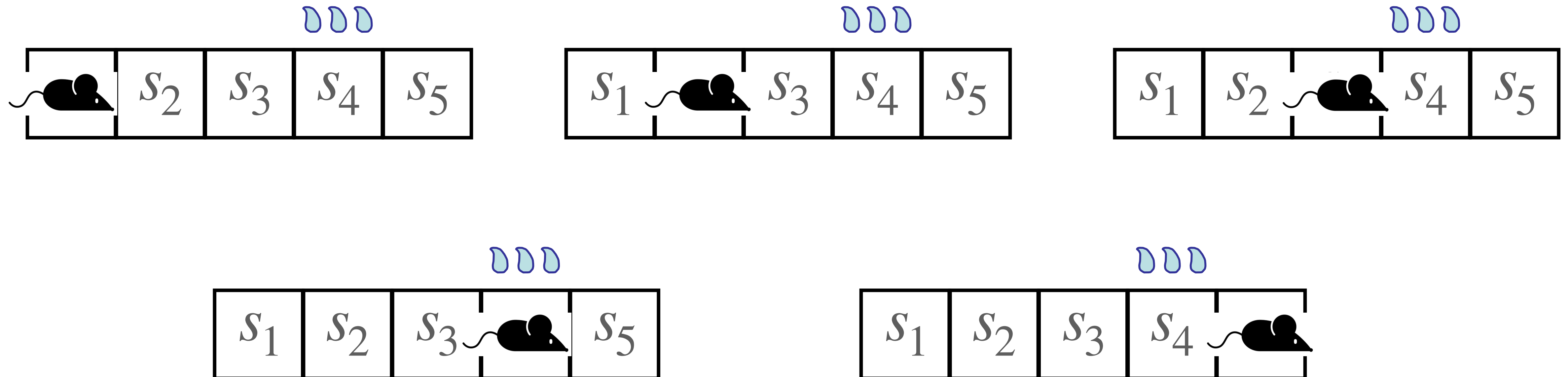


To define a **Markov Process**, we need to define a **state space** and **transition probabilities**

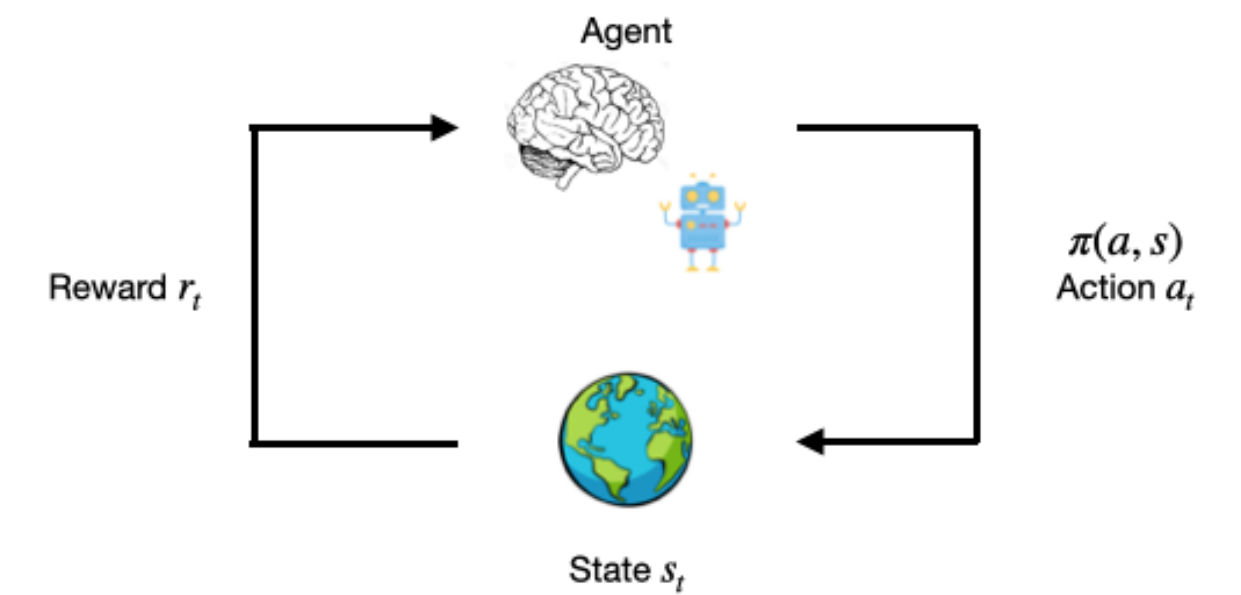
Markov Process



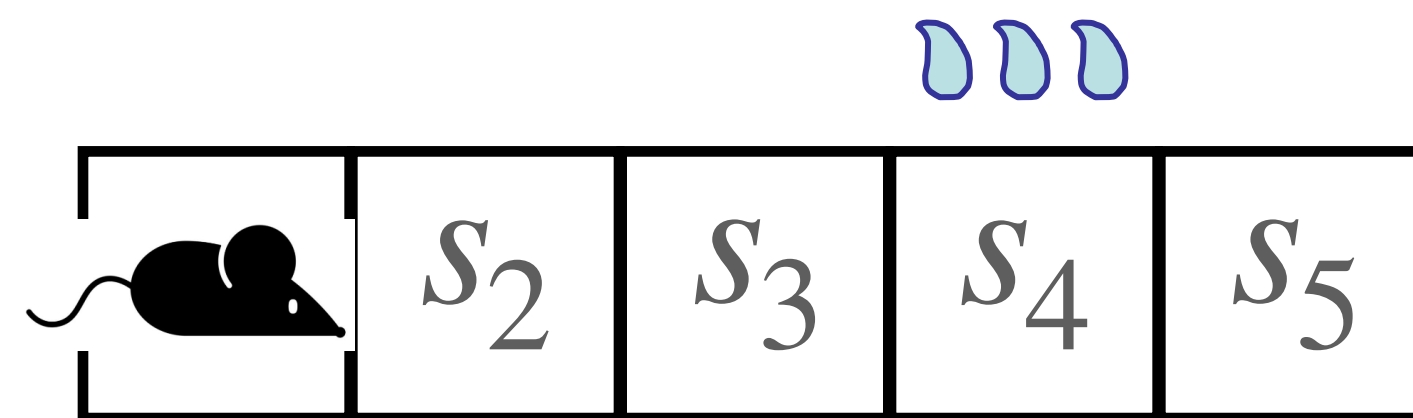
We can now define a **state space S** :



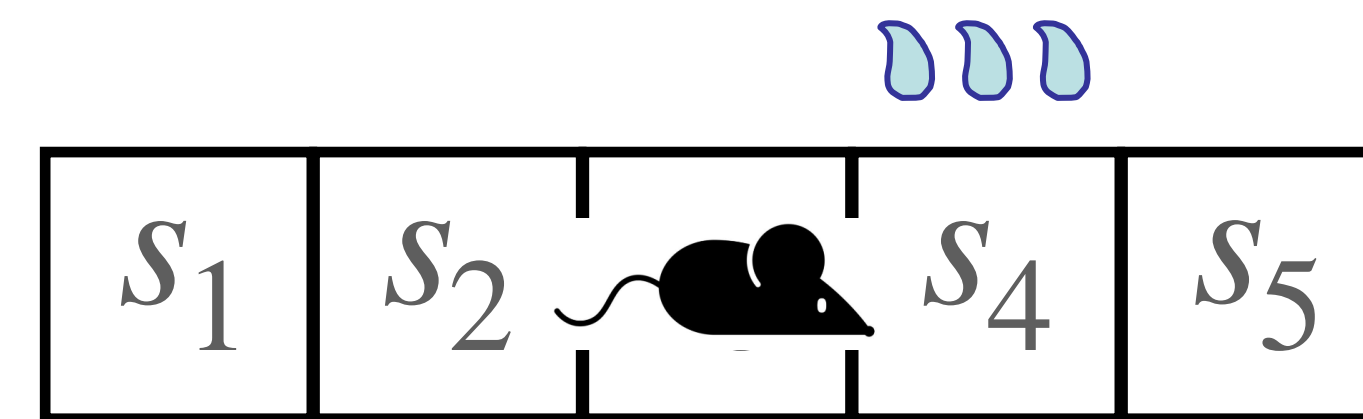
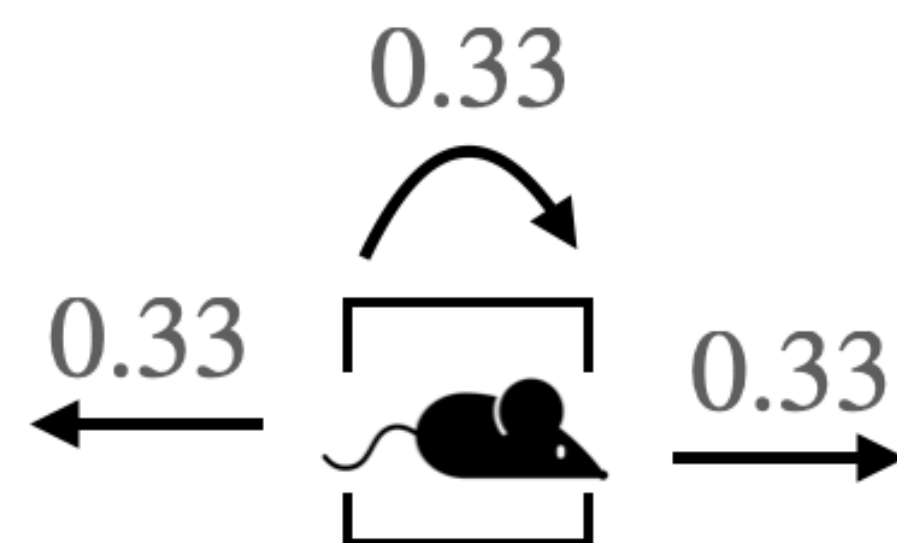
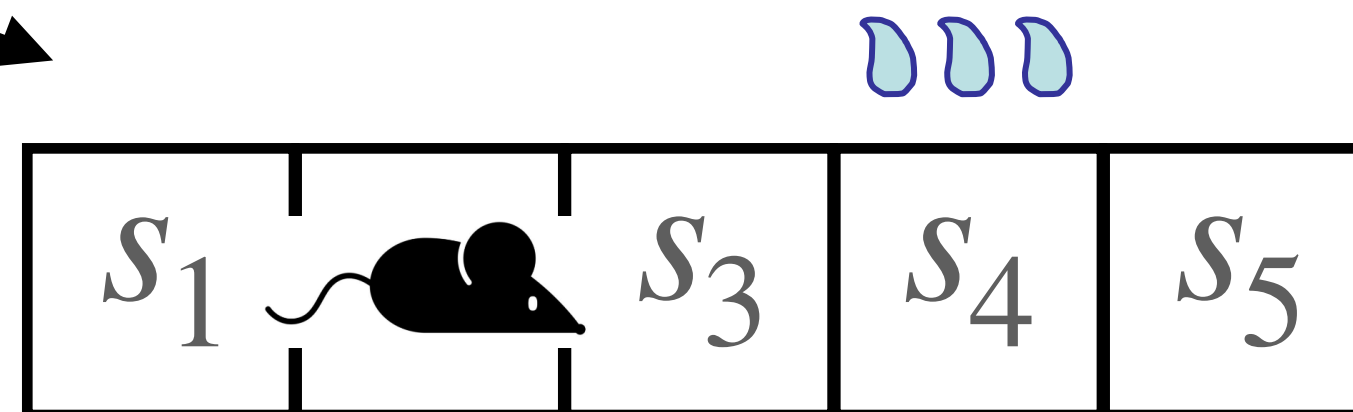
Markov Process



Does this problem have the **Markov Property**?

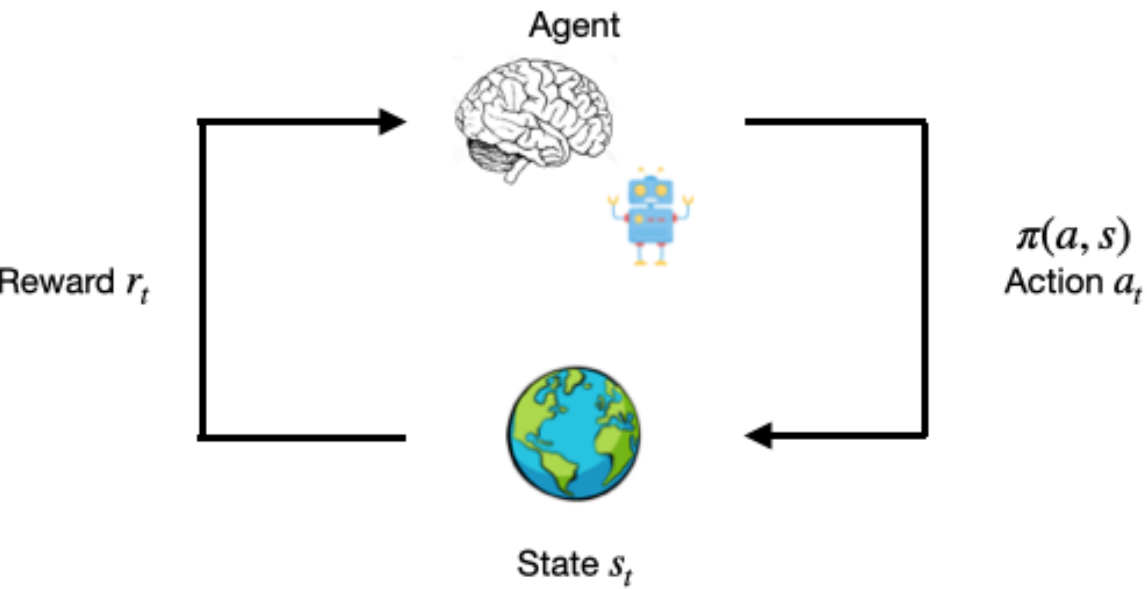


$$P(s_{t+1} = s \mid s_t, s_{t-1}, s_{t-2}, \dots) \stackrel{???}{=} P(s_{t+1} = s \mid s_t)$$

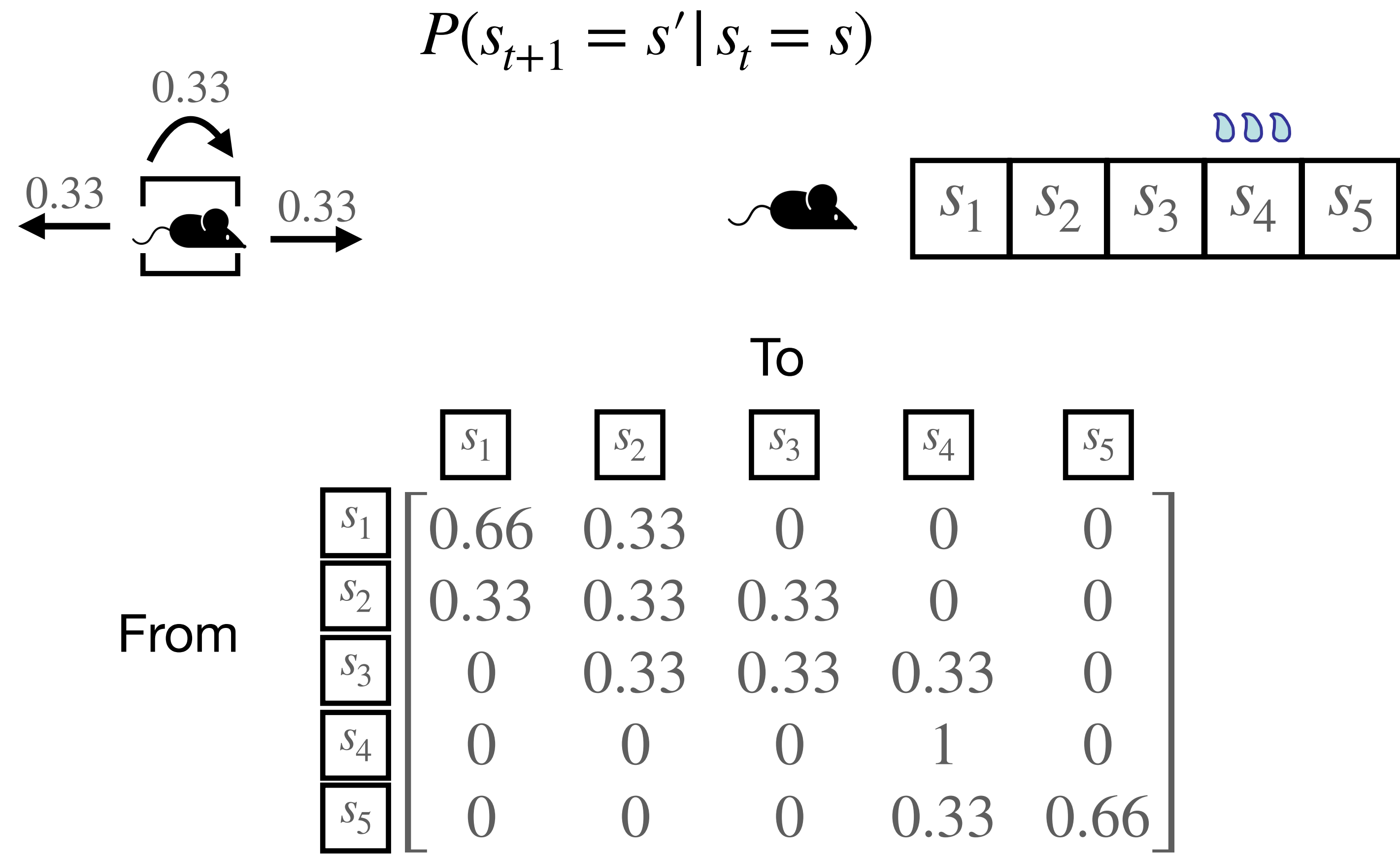


???

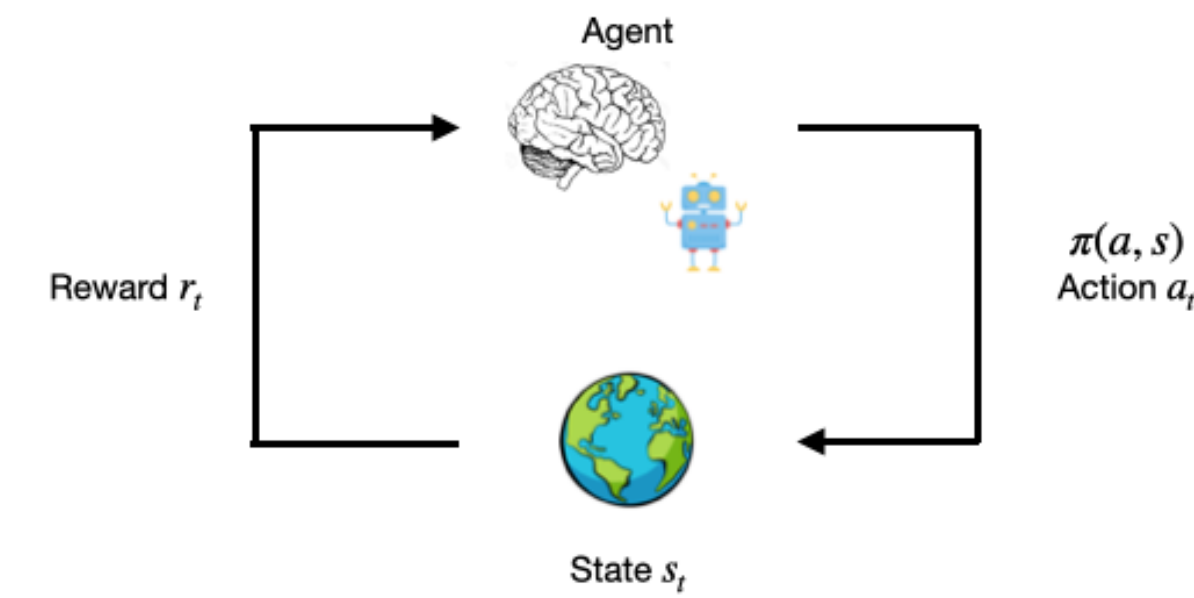
Markov Process



This allows us to define **transition probabilities** P :



Markov Process

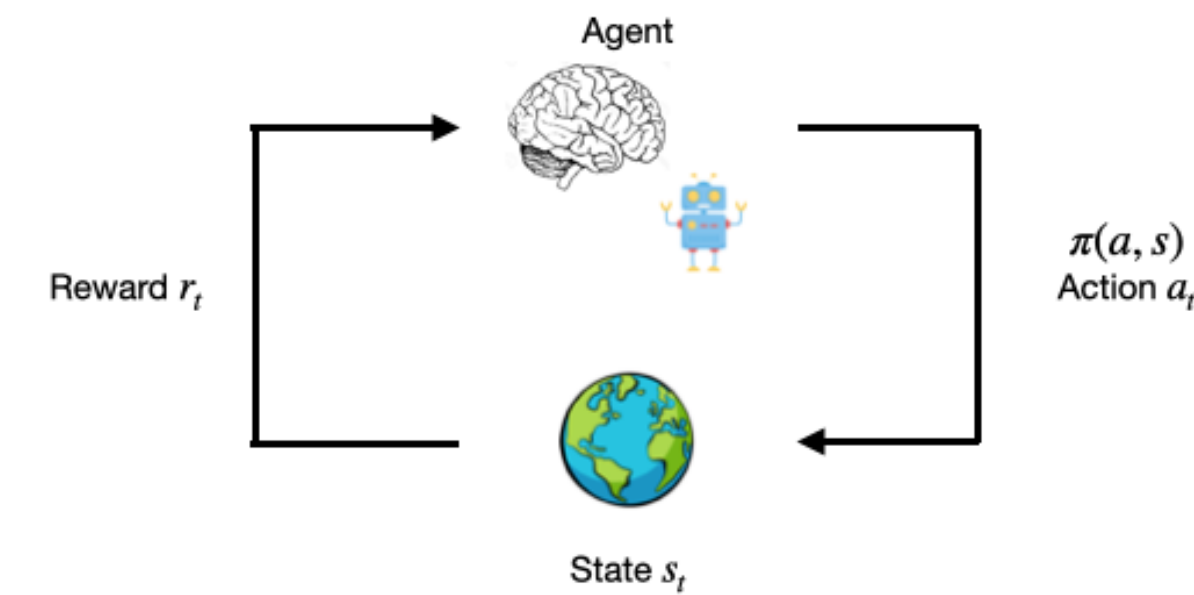


A **Markov Process** is defined based on

- A **State Space** \mathcal{S}
- **Transition Probabilities** P $P(s_{t+1} = s' | s_t = s)$

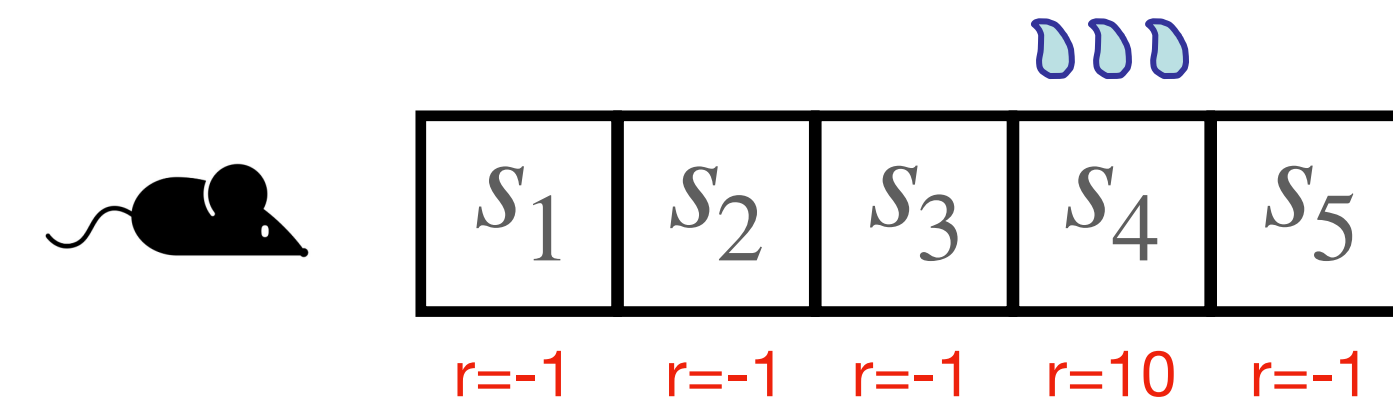
To define a **Markov Reward Process**, we need to add **rewards**

Markov Reward Process



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

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

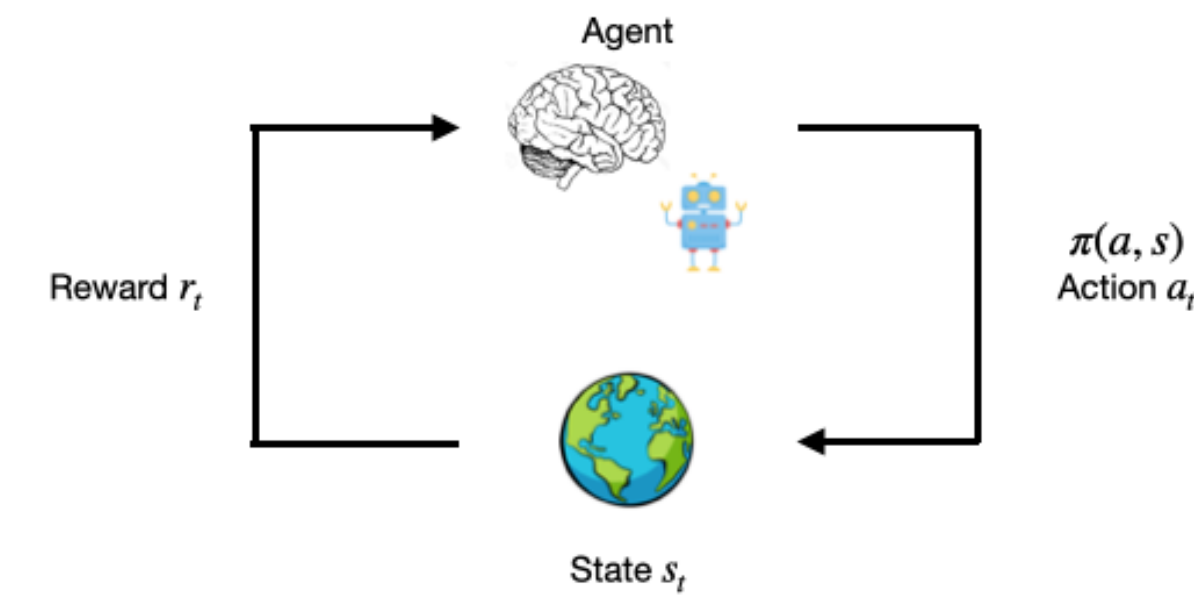


Allows to define **Return**

$$G_t = r_{t+1} + \gamma \cdot r_{t+2} + \gamma^2 \cdot r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k+1}$$

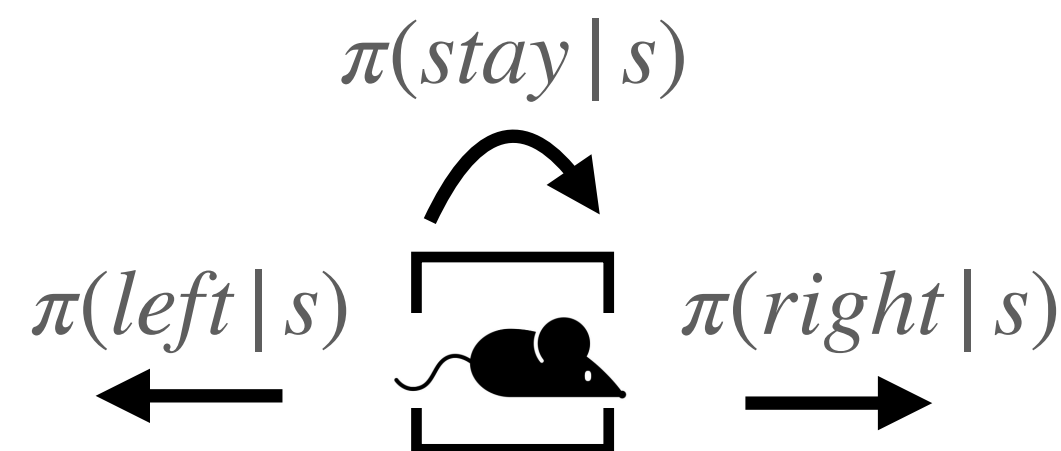
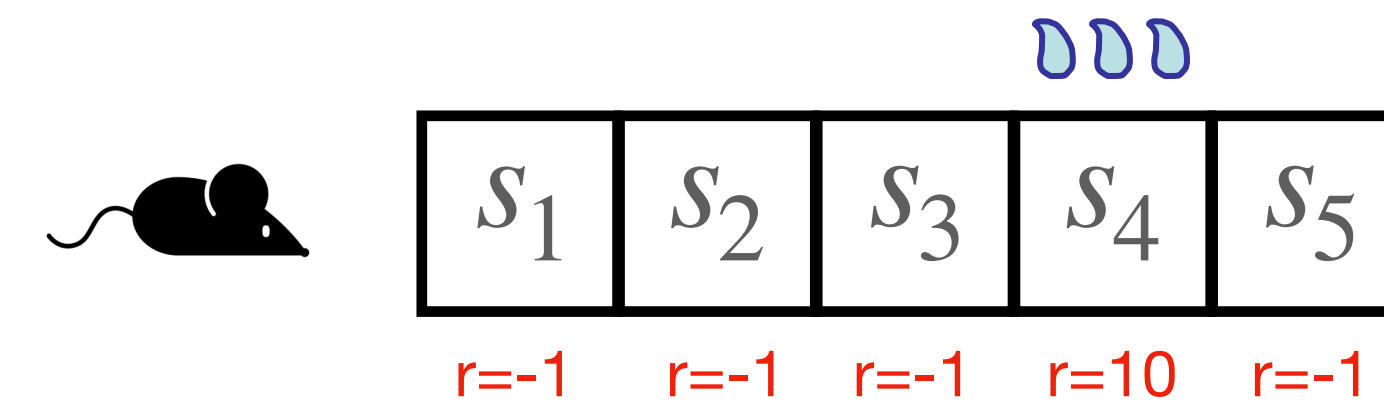
To define a **Markov Decision Process**, we need to add **actions**

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 $R_s = \mathbb{E}[r_t | s_t = s]$
- A Discount Factor $\gamma \in [0, 1]$

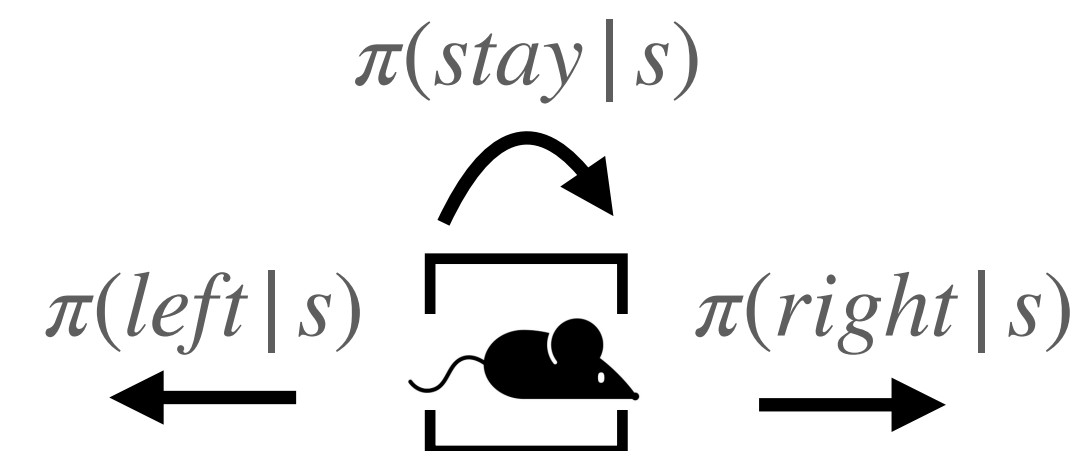
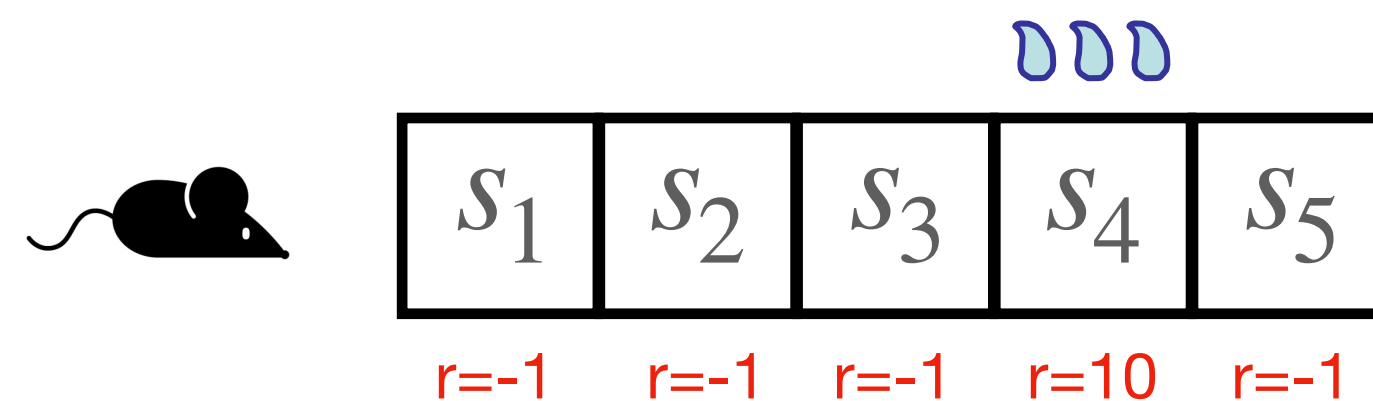


Actions are governed via a **policy**: $\pi(a, s) = P(a_t = a | s_t = s)$

MDPs basis for model-based RL

Allows to specify all environment dynamics for RL problem:

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

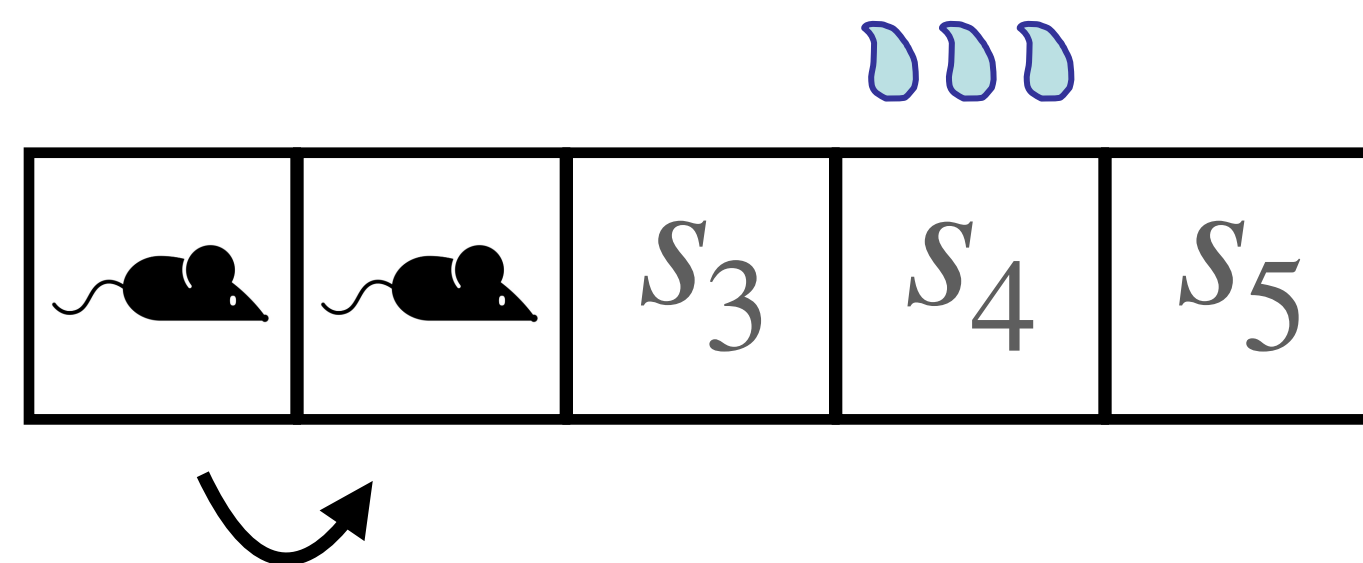


MDPs basis for model-based RL

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

Allows to specify useful things like **state-transition probabilities** (often T):

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



$$P(s' | s, right) = \begin{array}{c|ccccc} & s_1 & s_2 & s_3 & s_4 & s_5 \\ \hline s_1 & 0 & 1 & 0 & 0 & 0 \\ s_2 & 0 & 0 & 1 & 0 & 0 \\ s_3 & 0 & 0 & 0 & 1 & 0 \\ s_4 & 0 & 0 & 0 & 1 & 0 \\ s_5 & 0 & 0 & 0 & 0 & 1 \end{array}$$

MDPs basis for model-based RL

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

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 $P(s', r | s, a)$
- Sample from this model to generate extra learning data
- This is called **DYNA-Q...**