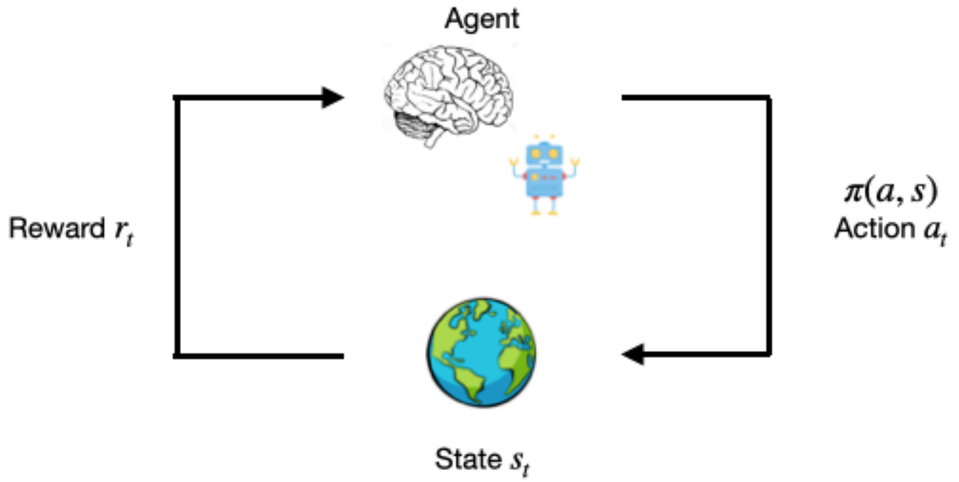
An introduction to Reinforcement Learning

12th of July 2022

Recap Q-Learning



TD Learning:

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

Prediction error

Learning rate

Discount rate

Q-Learning:

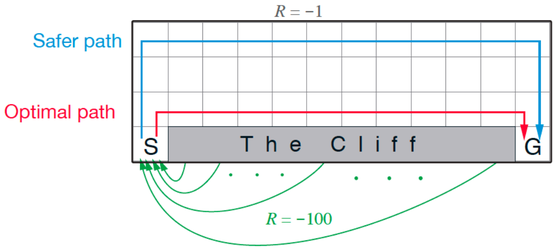
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot (r + \gamma \cdot max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

Prediction error

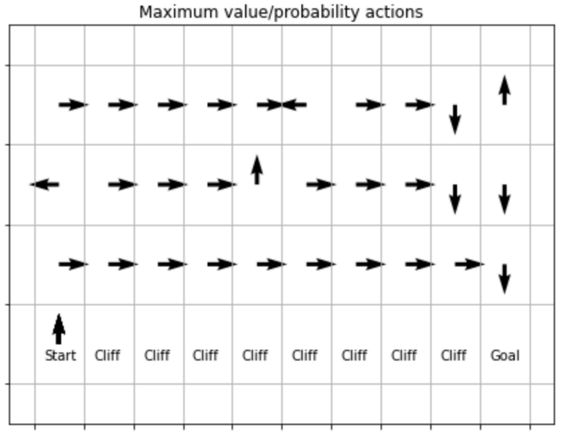
Learning rate

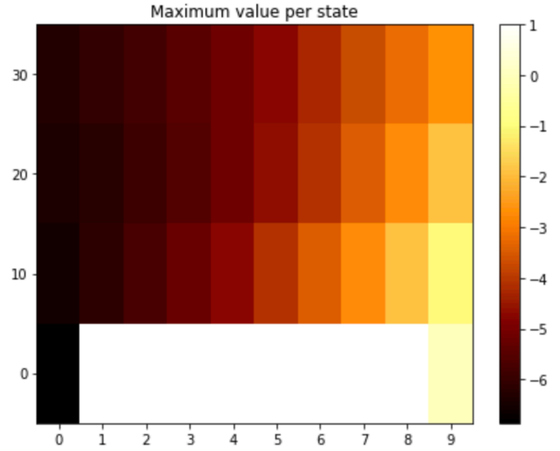
Discount rate

Recap Q-Learning



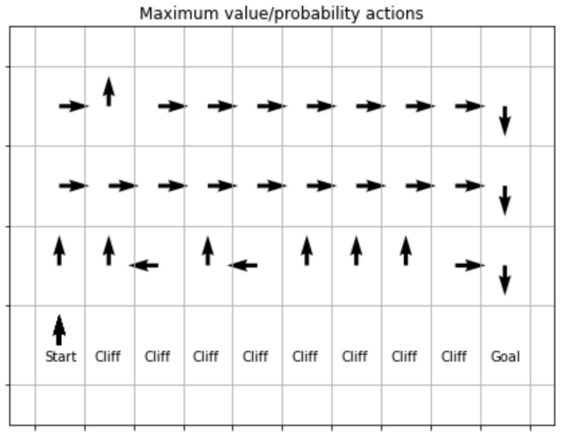
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big(r_t + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big)$$



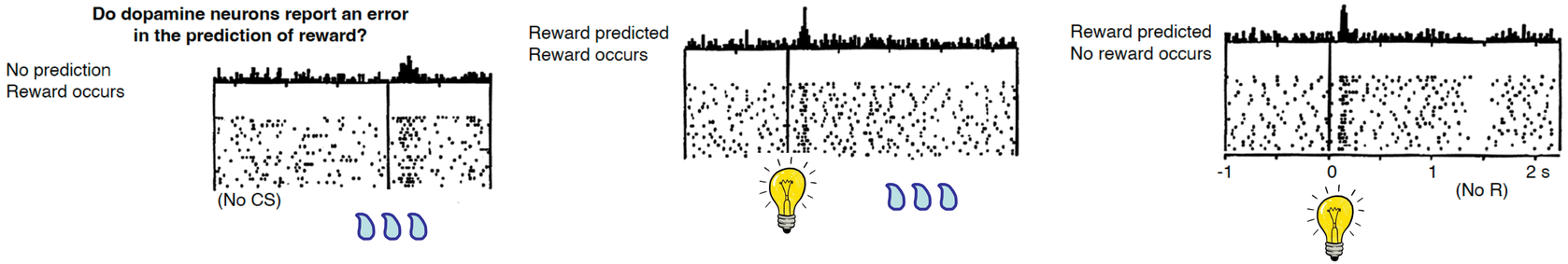


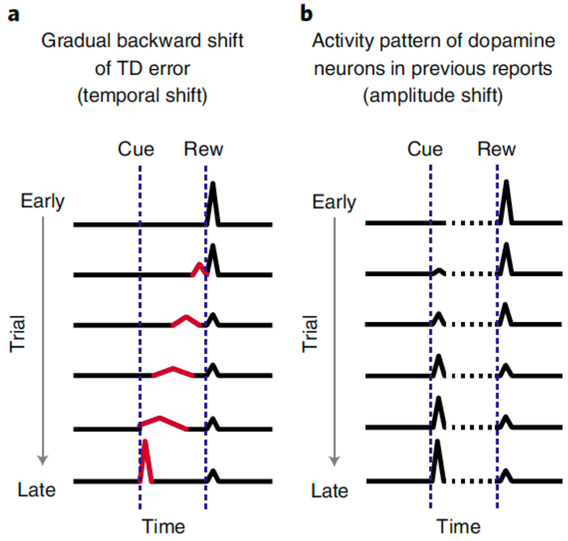
There are also alternatives:

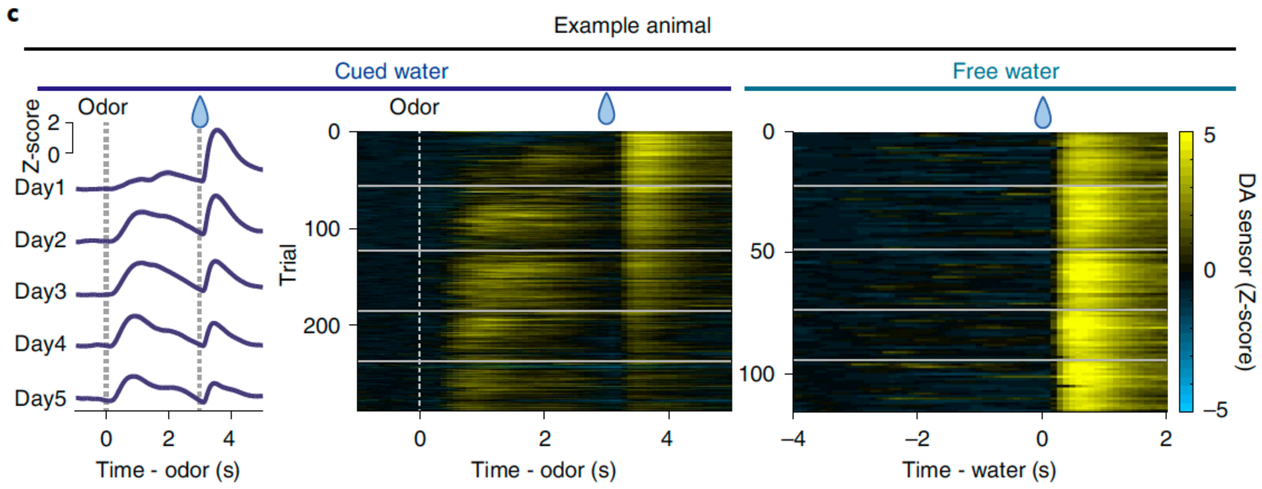
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\Big(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\Big)$$



Q- (TD-) learning in action







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

MDPs and model-based RL

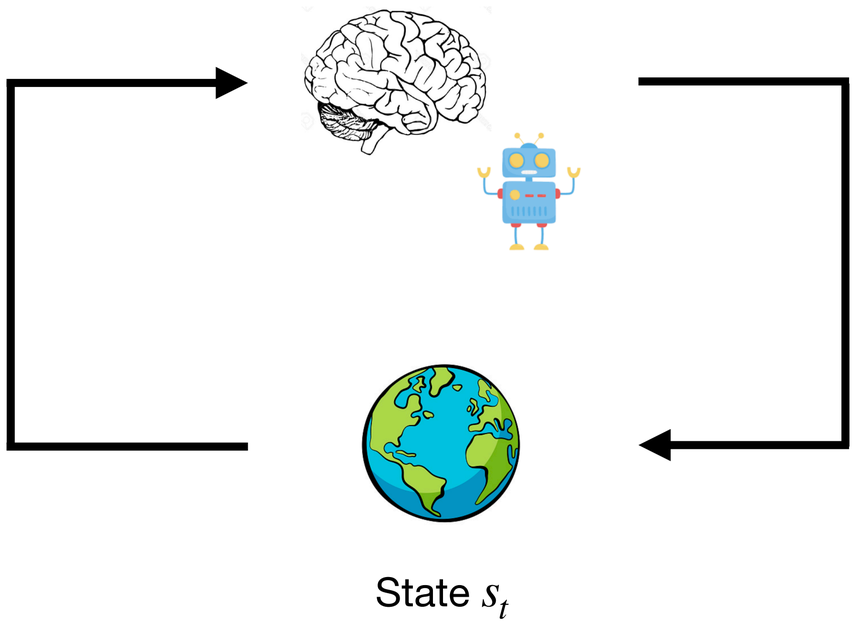
Basic setup: how do agents learn to act?

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

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

Action is governed by a **policy**:

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



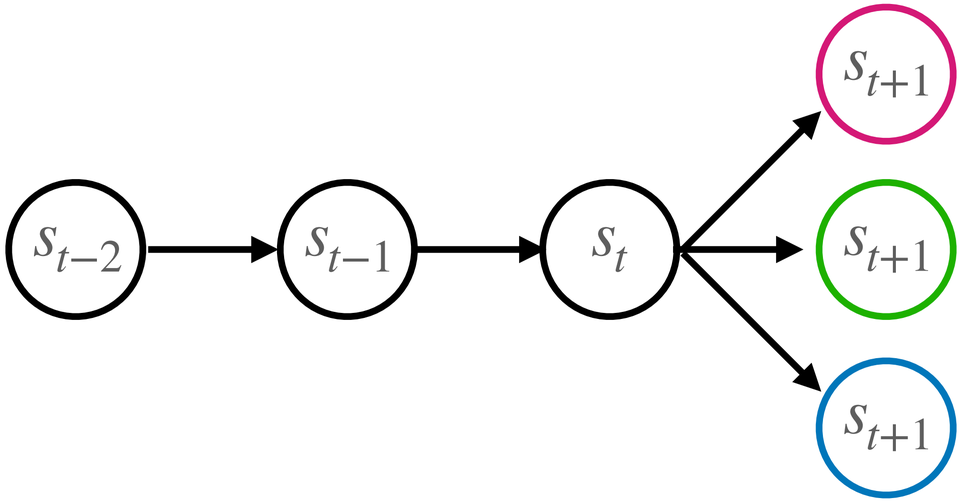
Agents can learn a **model of the environment** to make smarter decisions, e.g.:

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

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

Markov Process

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

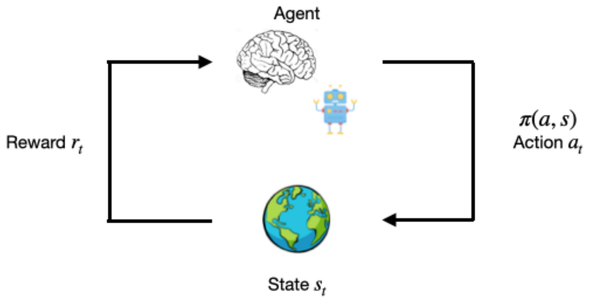


Fundamental property: **Markov property**

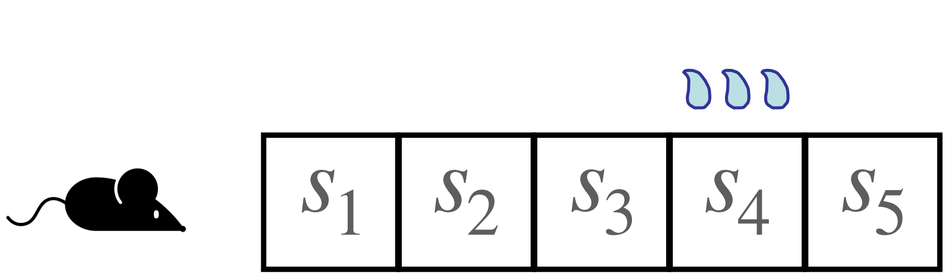
$$P(s_{t+1} = s \mid s_t, s_{t-1}, s_{t-2}, \dots) = P(s_{t+1} = s \mid s_t)$$

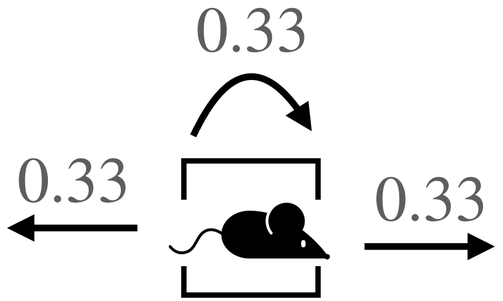
“The future is independent of the past given the present”

Markov Process



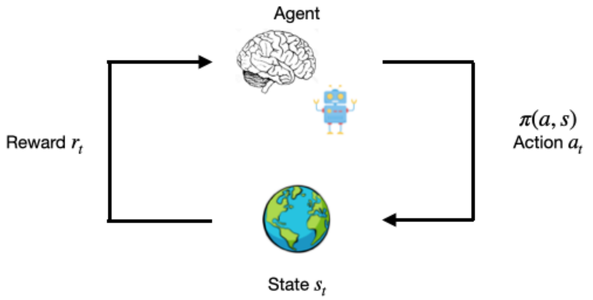
Let's assume a super simple problem:



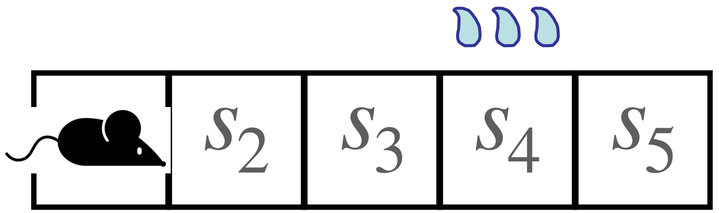


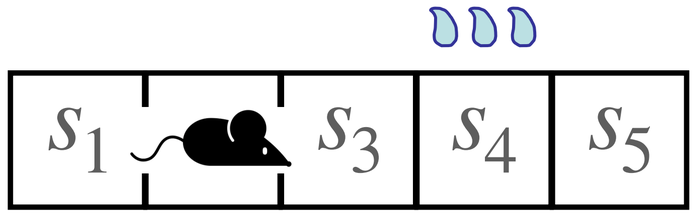
To define a **Markov Process**, we need to define a **state space** and **transition probabilities**

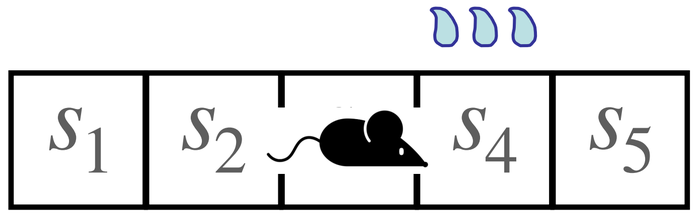
Markov Process

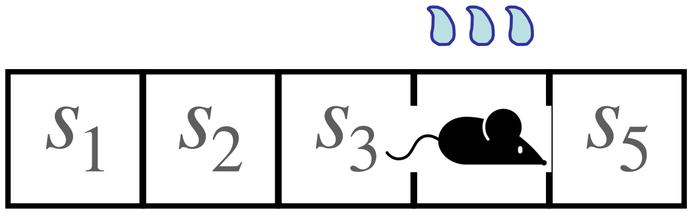


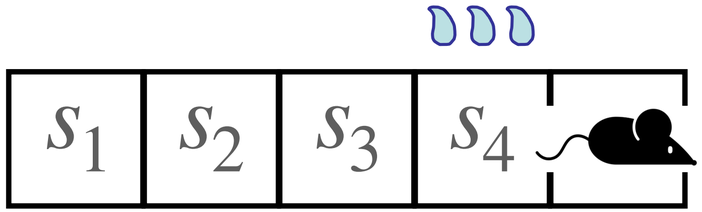
We can now define a **state space** S:



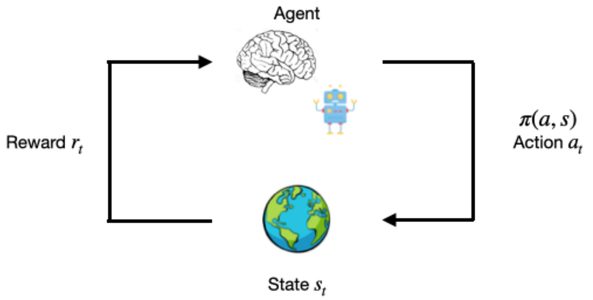




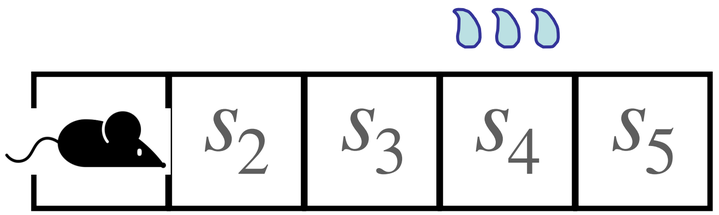




Markov Process

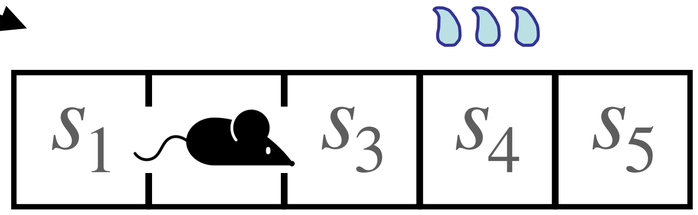


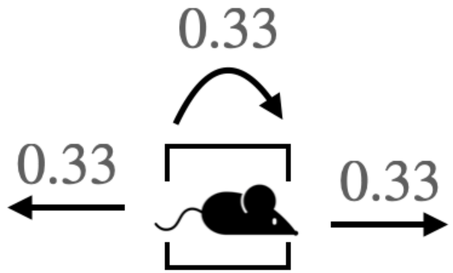
Does this problem have the **Markov Property?**

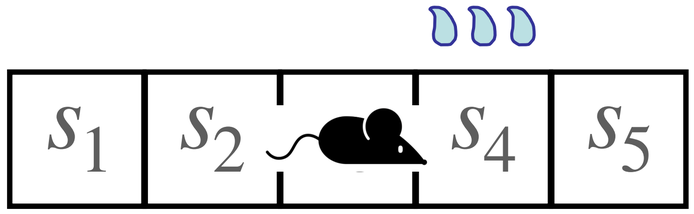


???

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

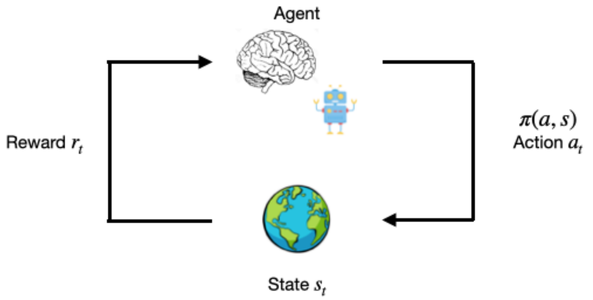






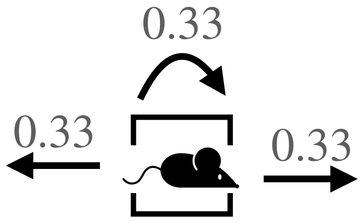
???

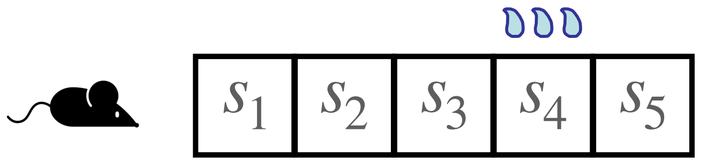
Markov Process



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

$$P(s_{t+1} = s' \mid s_t = s)$$



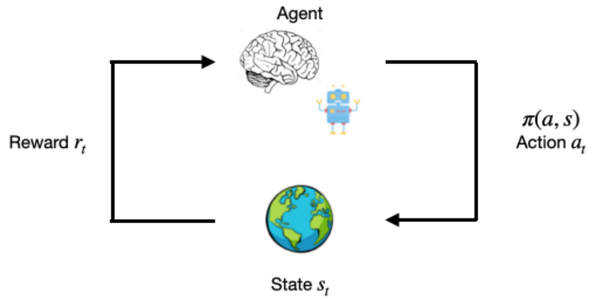


To

From

	s_1	s_2	s_3	s_4	s_5
s_1	0.66	0.33	0	0	0
s_2	0.33	0.33	0.33	0	0
s_3	0	0.33	0.33	0.33	0
s_4	0	0	0	1	0
s_5	0	0	0	0.33	0.66

Markov Process

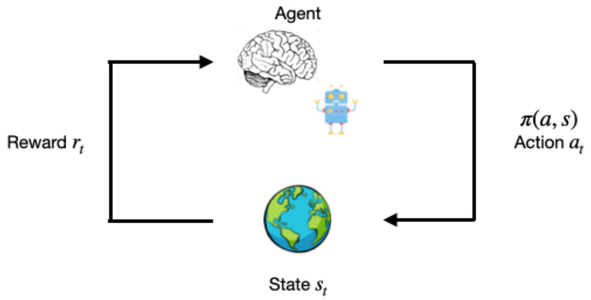


A **Markov Process** is defined based on

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

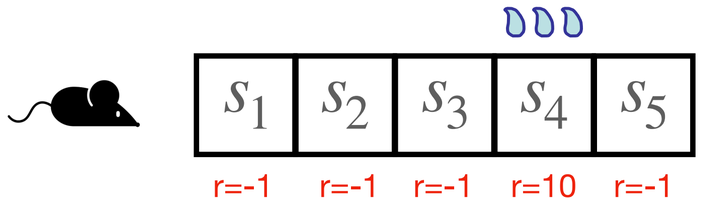
To define a **Markov Reward Process**, we need to add **rewards**

Markov Reward Process



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

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

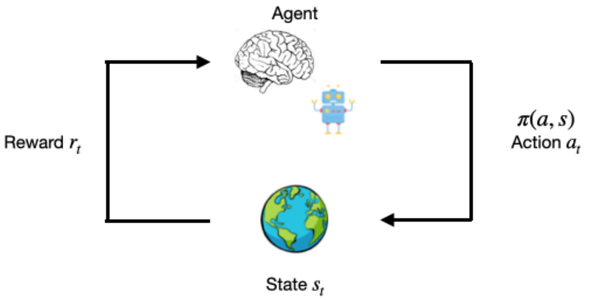


Allows to define **Return**

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

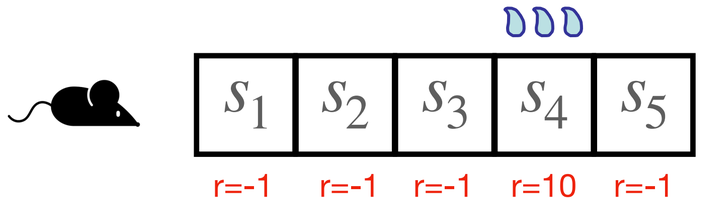
To define a **Markov Decision Process**, we need to add **actions**

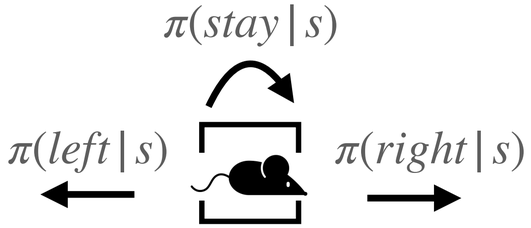
Markov Decision Process



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

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



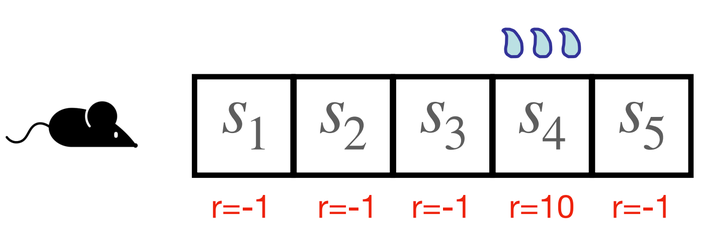


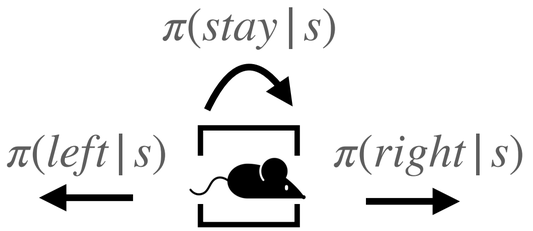
Actions are governed via a **policy**: $\pi(a, s) = P(a_t = a \mid s_t = s)$

MDPs basis for model-based RL

Allows to specify all environment dynamics for RL problem:

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



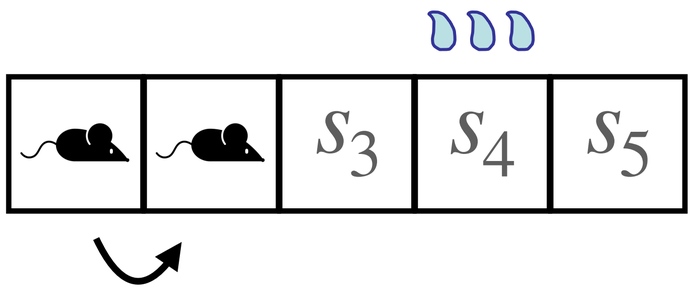


MDPs basis for model-based RL

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

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

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



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

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…**

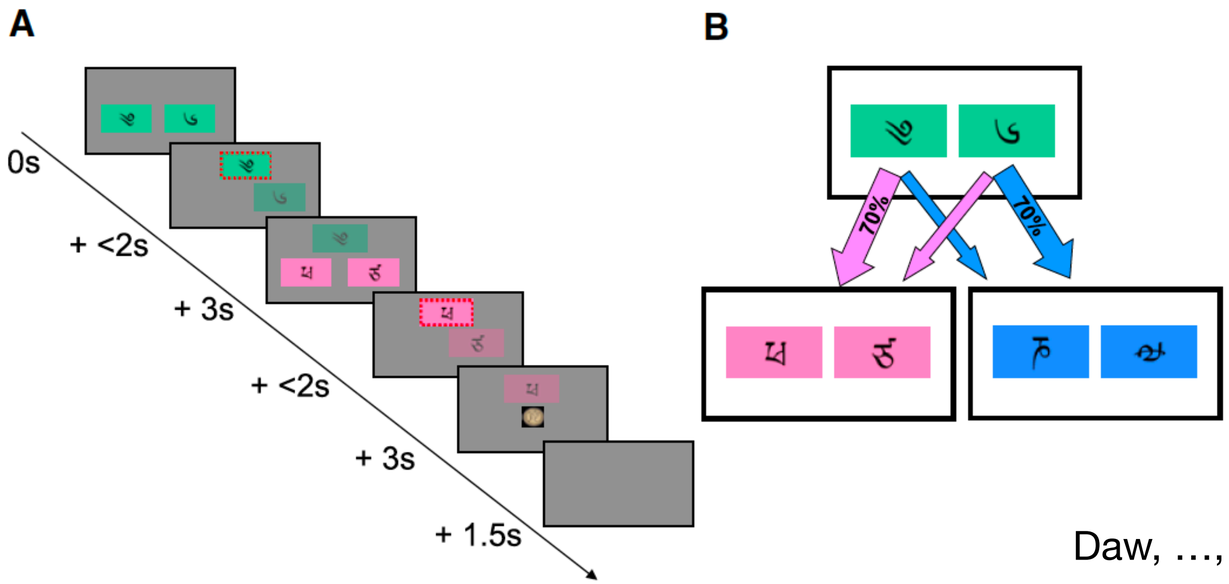
Coding: DYNA-Q

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

Model-free vs. Model-based control

Two-step task: one of the most iconic RL tasks

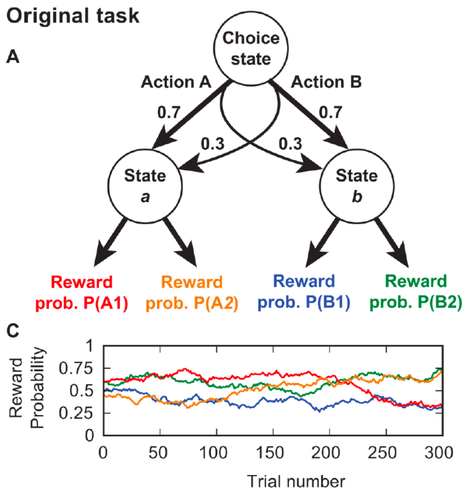
Choose twice between two options to obtain a reward

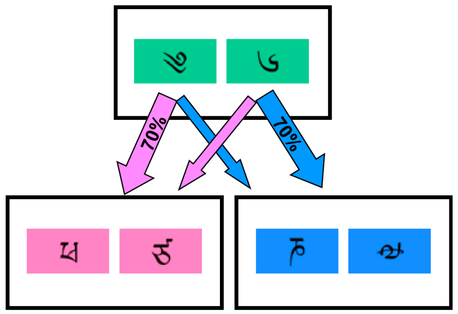


Key manipulation: **common** and **rare** transitions - how should you update?

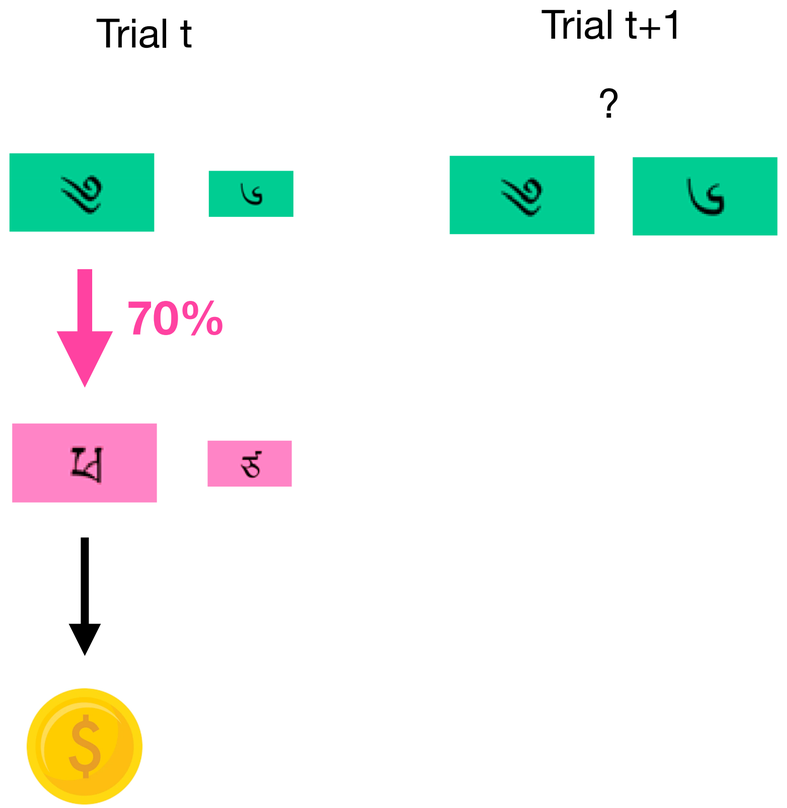
Daw, …, Dolan, Neuron, 2011

Two-step task: one of the most iconic RL tasks



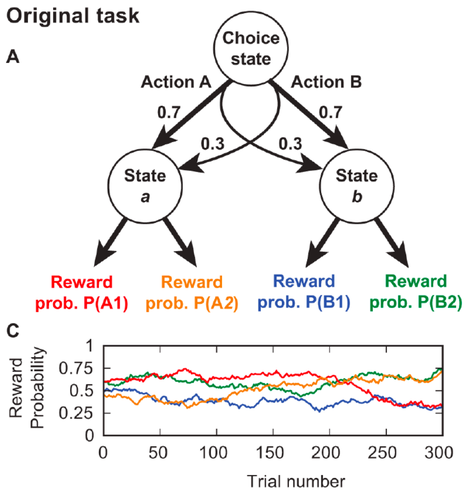


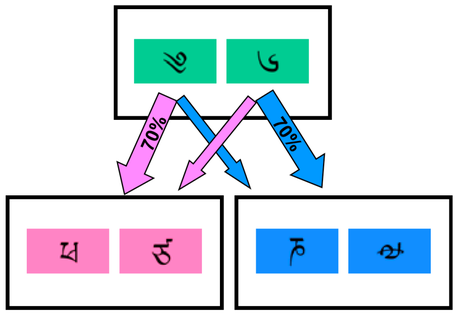
Akam, Costa, Dayan,
PLOS Computational Biology, 2015



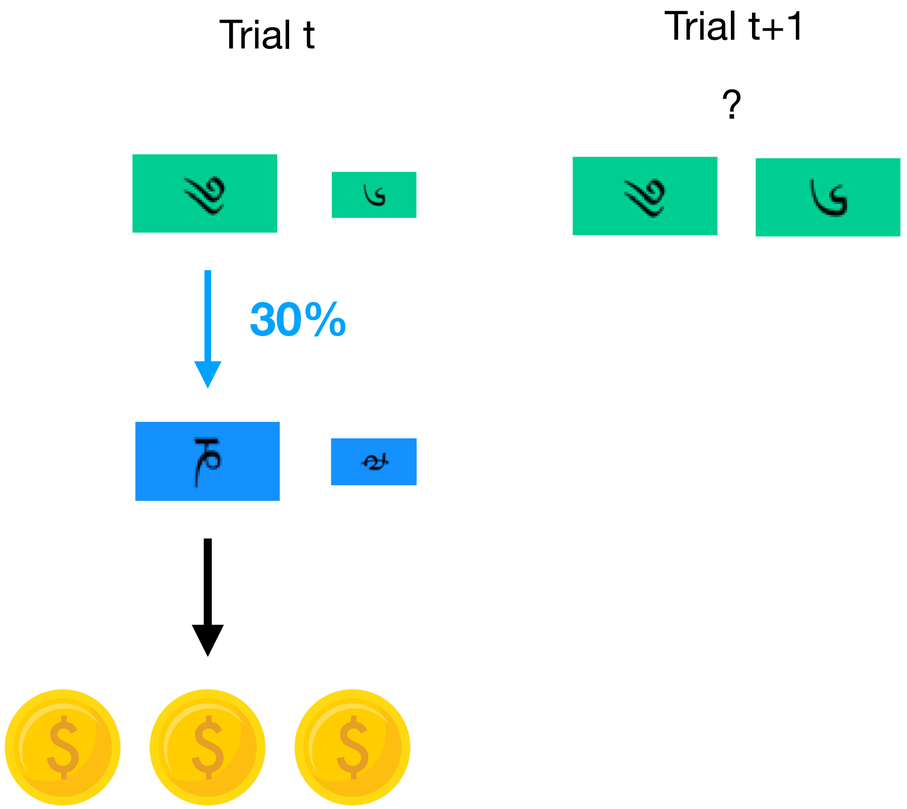
Which green option should the agent choose again at trial t+1?

Two-step task: one of the most iconic RL tasks



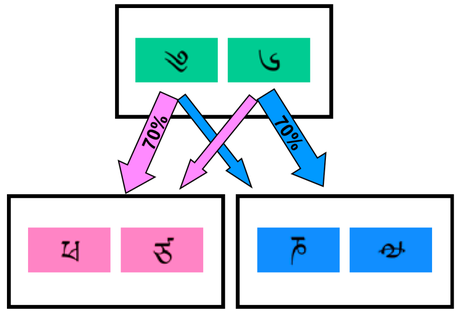


Akam, Costa, Dayan,
PLOS Computational Biology, 2015



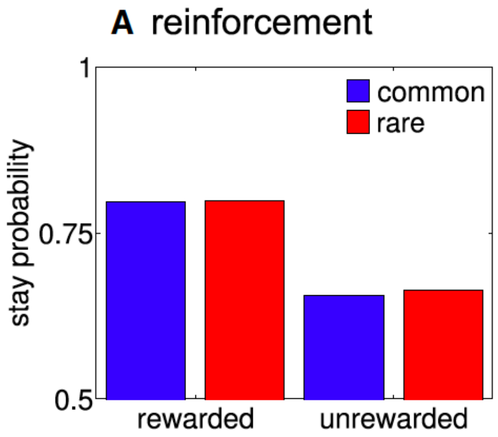
Which green option should the agent choose again at trial t+1?

Two-step task: one of the most iconic RL tasks

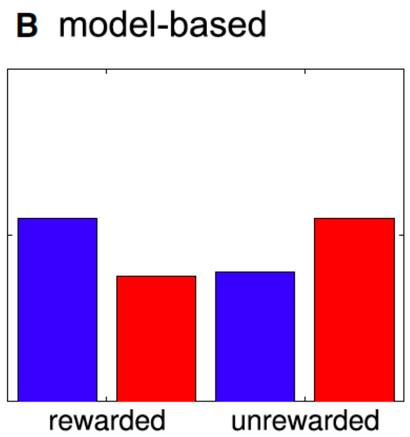


Trial t+1

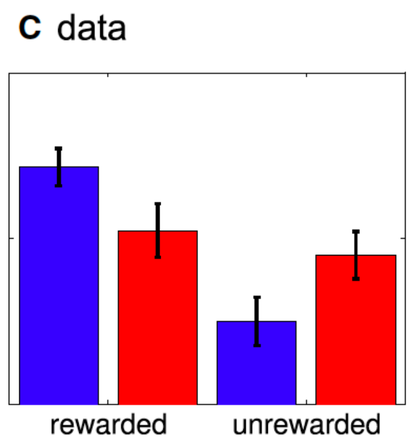
?



Model-free RL agent: repeat what is rewarding



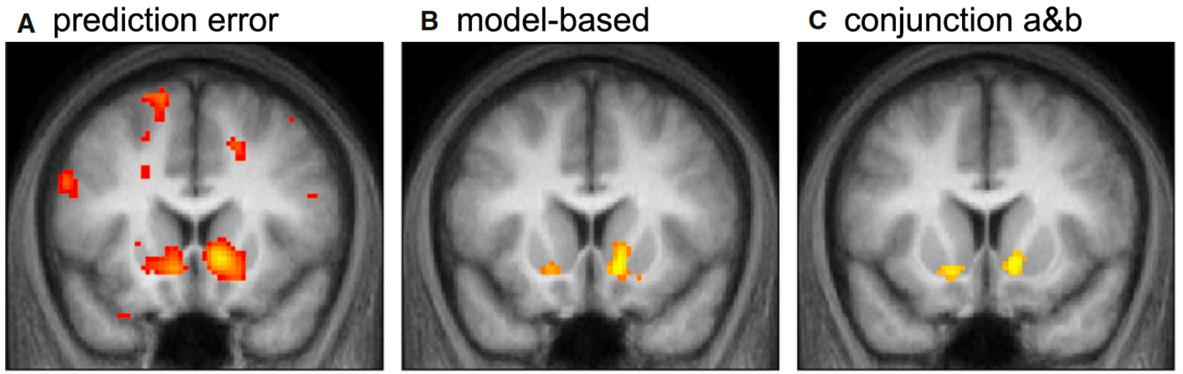
Model-based RL agent: repeat what is rewarding, but be clever



Really data: a mix of both

Two-step task: one of the most iconic RL tasks

Model-free and model-based prediction errors in ventral striatum



Coding: 2-Step

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