An introduction to Reinforcement Learning

21st of June 2022

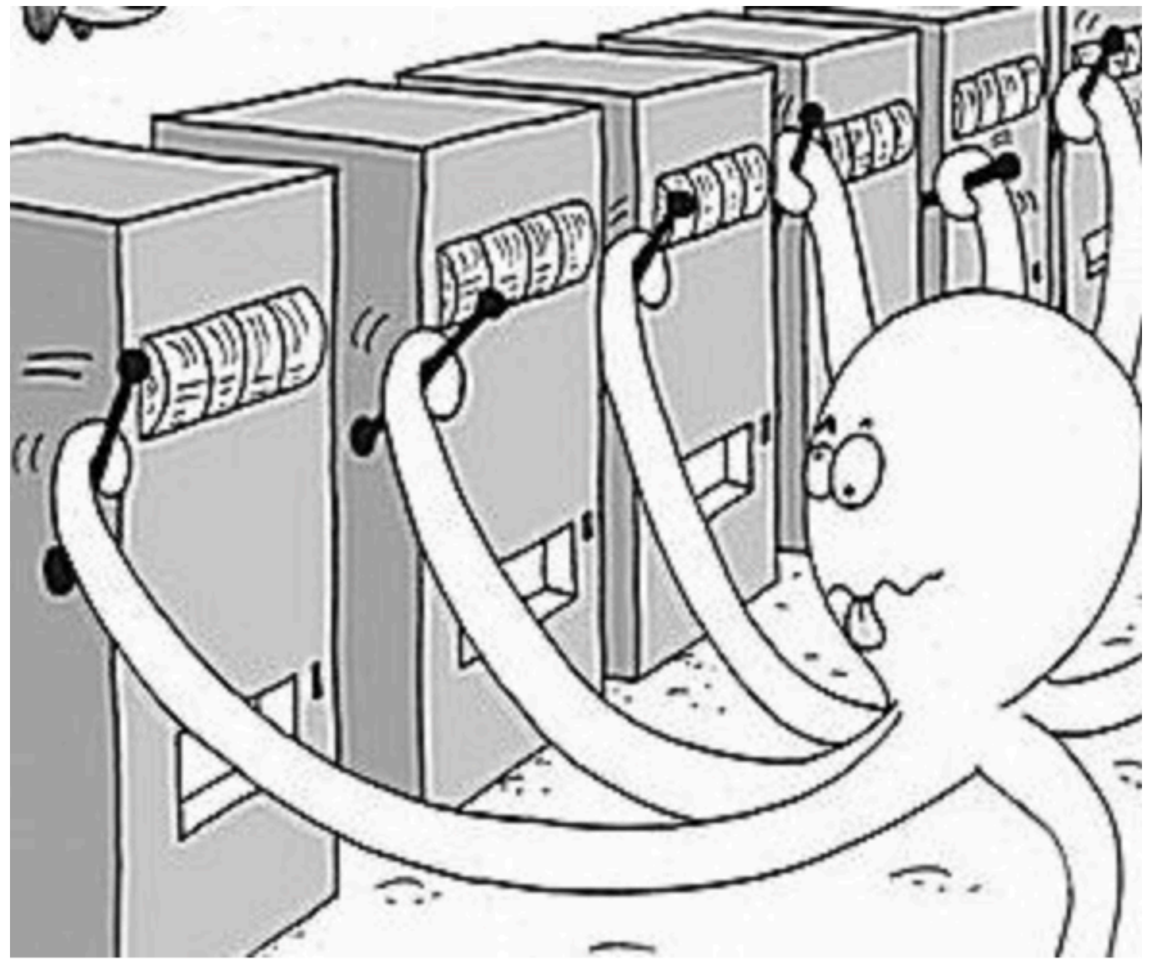
Multi-armed bandits

Greedy action selection:

$$P(a_t = a) = \begin{cases} 1 & \text{if } a_t = \text{argmax}_a \, V_t(a) \\ 0 & \text{otherwise} \end{cases}$$

Epsilon-greedy action selection:

$$P(a_t = a) = \begin{cases} 1 - \epsilon & \text{if } a_t = \text{argmax}_a \, V_t(a) \\ \epsilon/N & \text{otherwise} \end{cases}$$



Action is governed by a **policy**:

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

Softmax action selection:

$$P(a_t = a) = \frac{e^{V_t(a)\cdot\beta}}{\sum_{i=1}^{N} e^{V_t(a_i)\cdot\beta}}$$

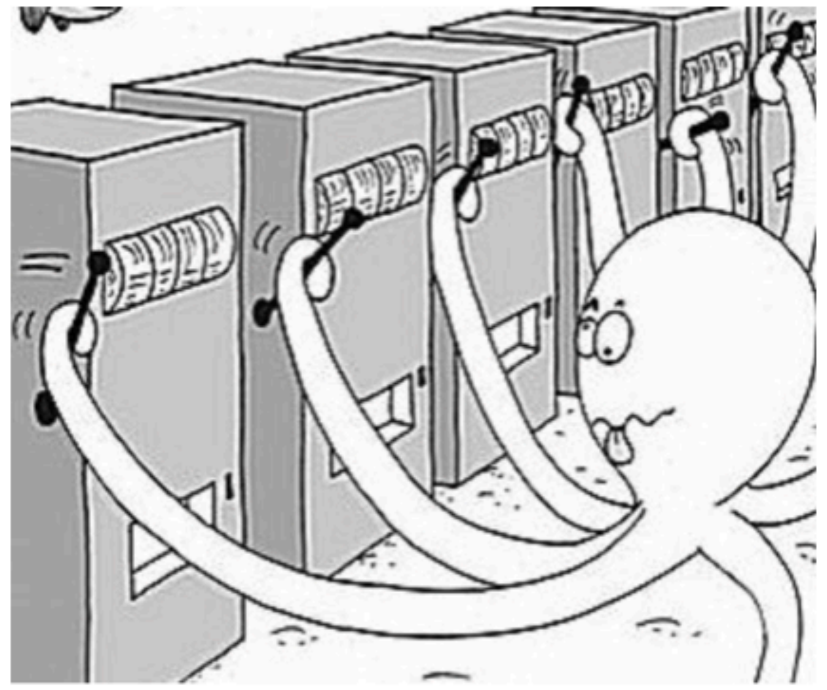
Upper-confidence-bound (UCB) action selection:

$$P(a_t = a) = \text{argmax}_a[V_t(a) + c \cdot \sqrt{\frac{\ln t}{N_t(a)}}]$$

Multi-armed bandits

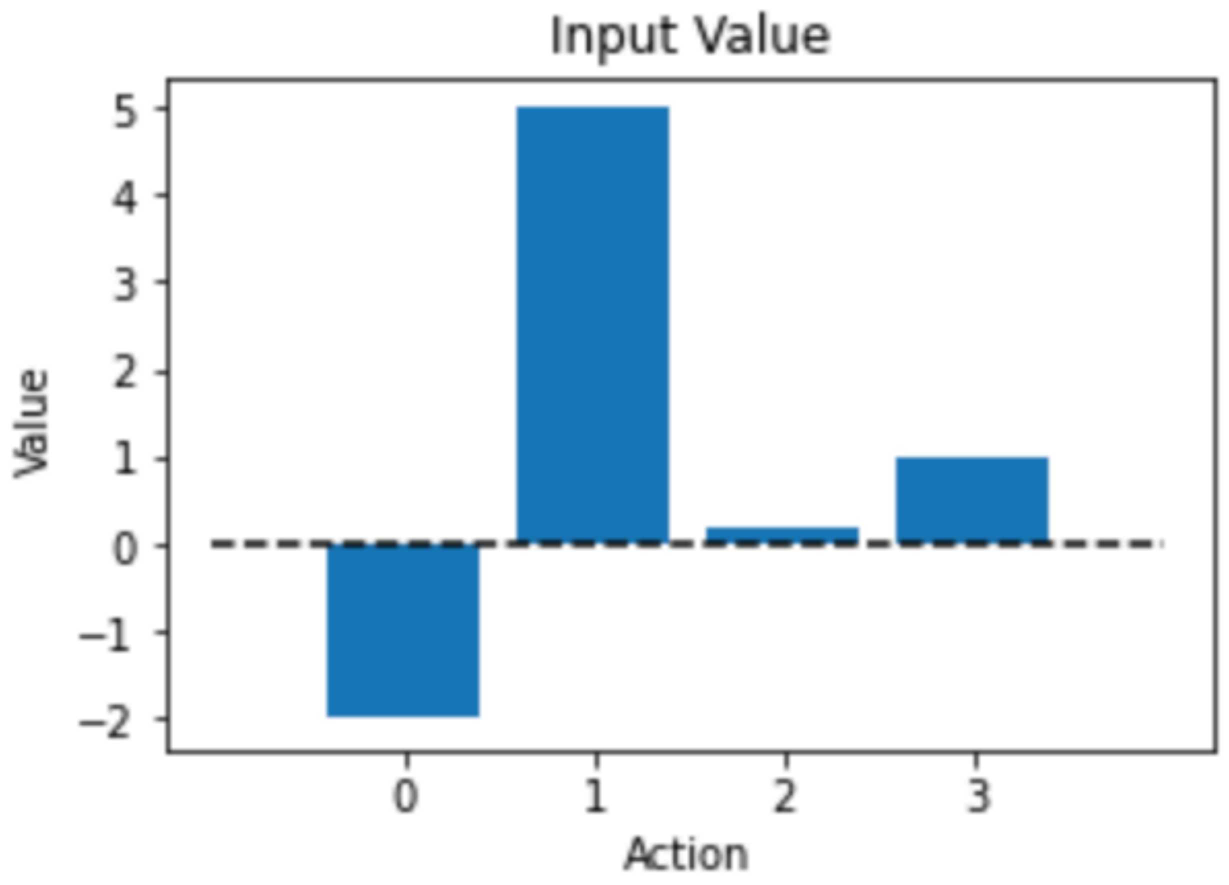
Greedy action selection:

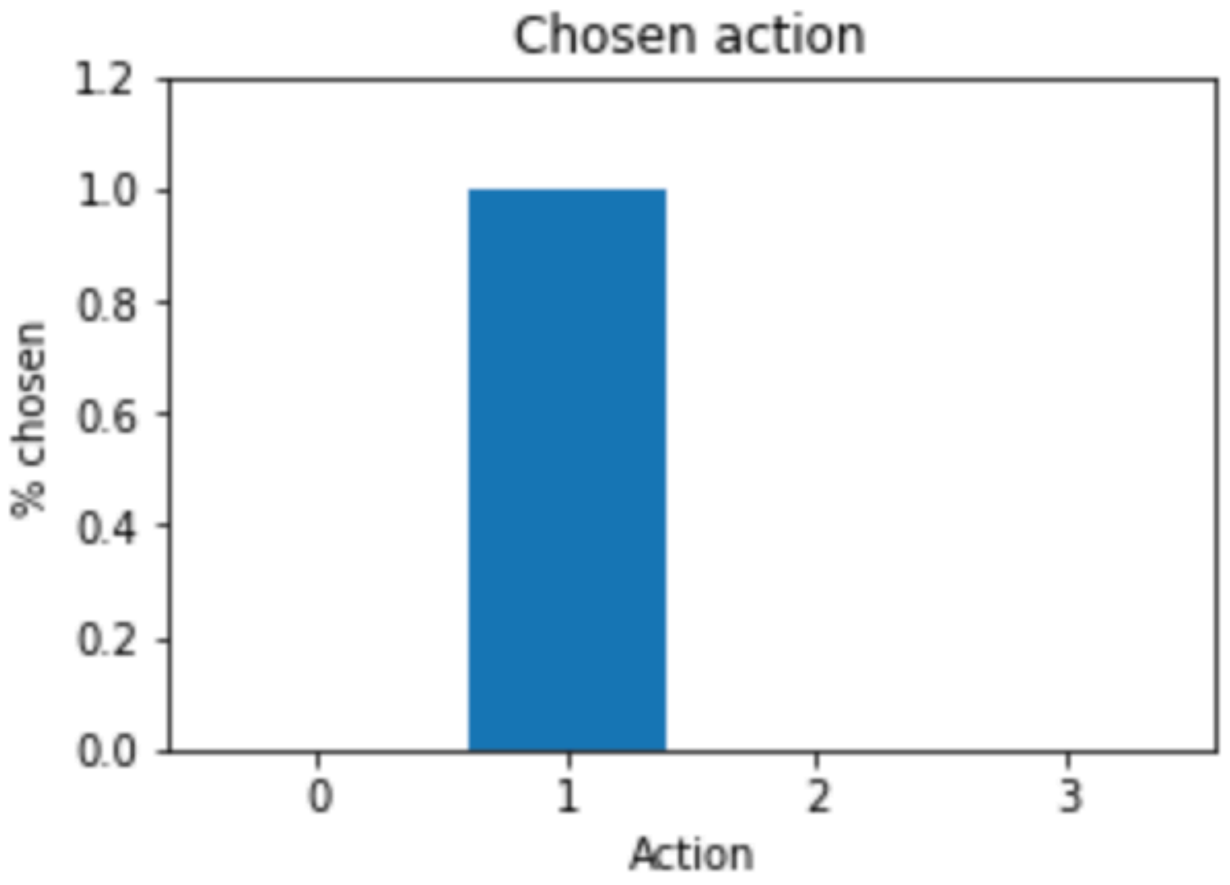
$$P(a_t = a) = \begin{cases} 1 & \text{if } a_t = \text{argmax}_a \ V_t(a) \\ 0 & \text{otherwise} \end{cases}$$



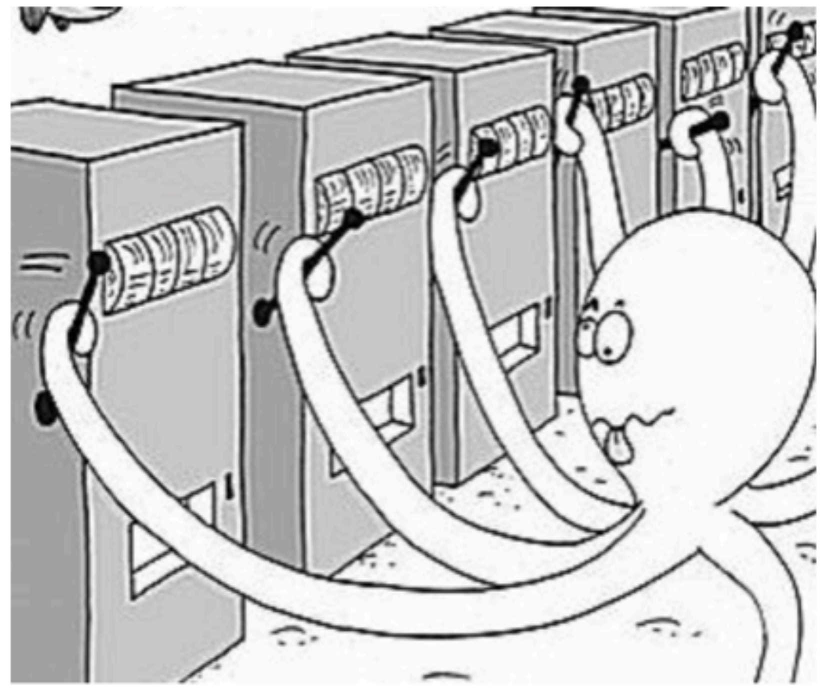
Action is governed by a **policy**:

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





Multi-armed bandits

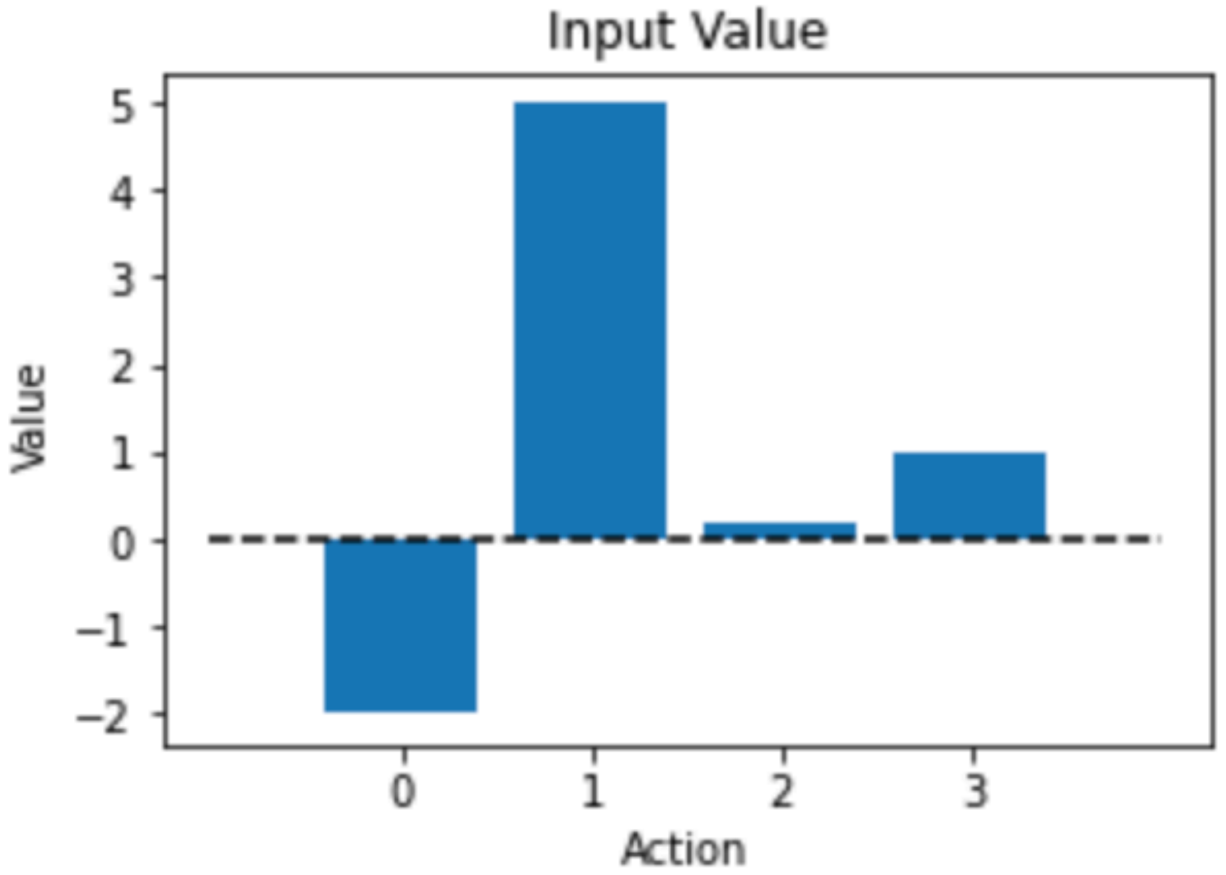


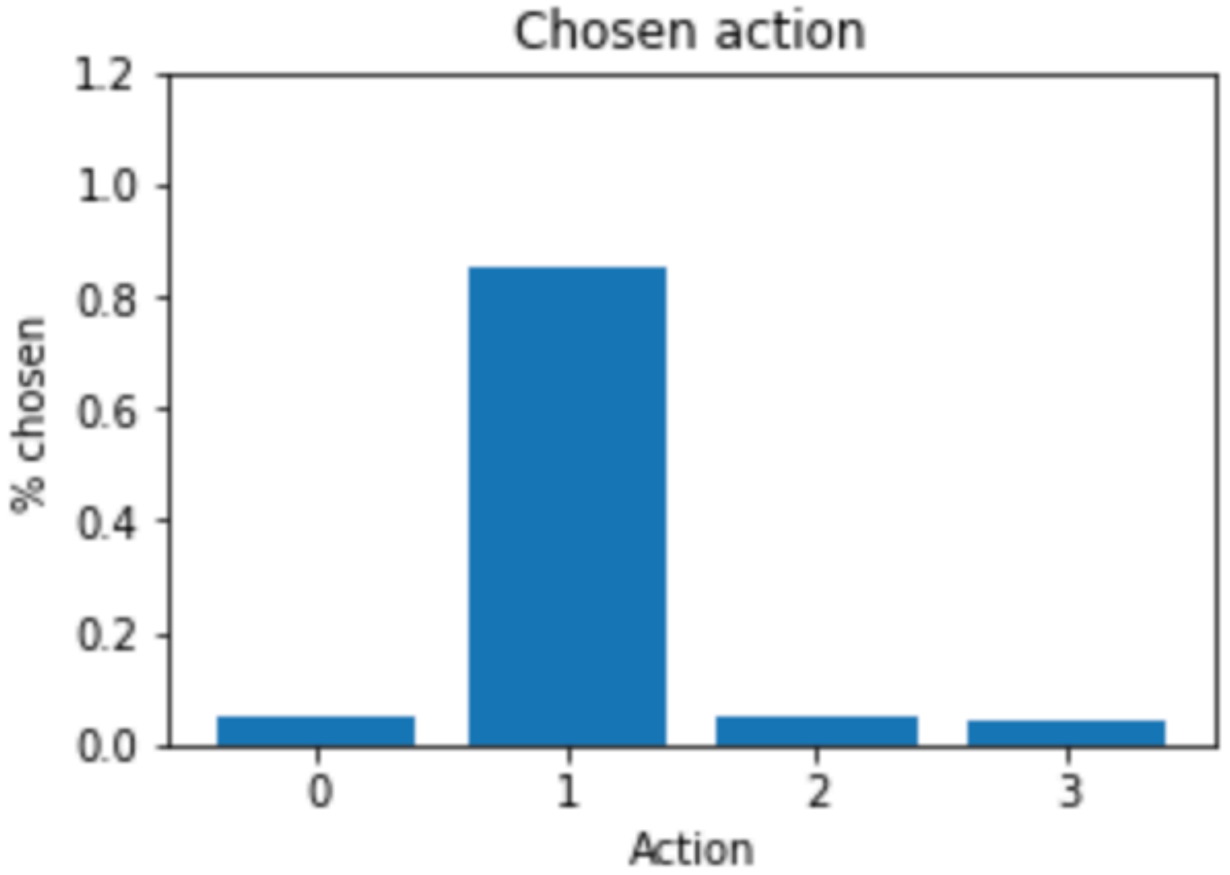
Action is governed by a **policy**:

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

Epsilon-greedy action selection:

$$P(a_t = a) = \begin{cases} 1 - \epsilon & \text{if } a_t = \text{argmax}_a\, V_t(a) \\ \epsilon / N & \text{otherwise} \end{cases}$$

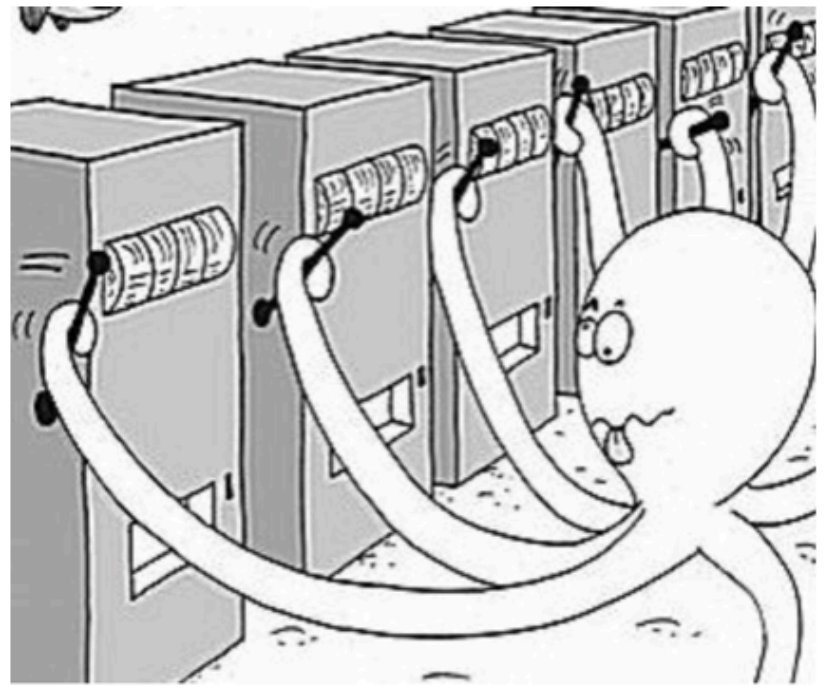




Multi-armed bandits

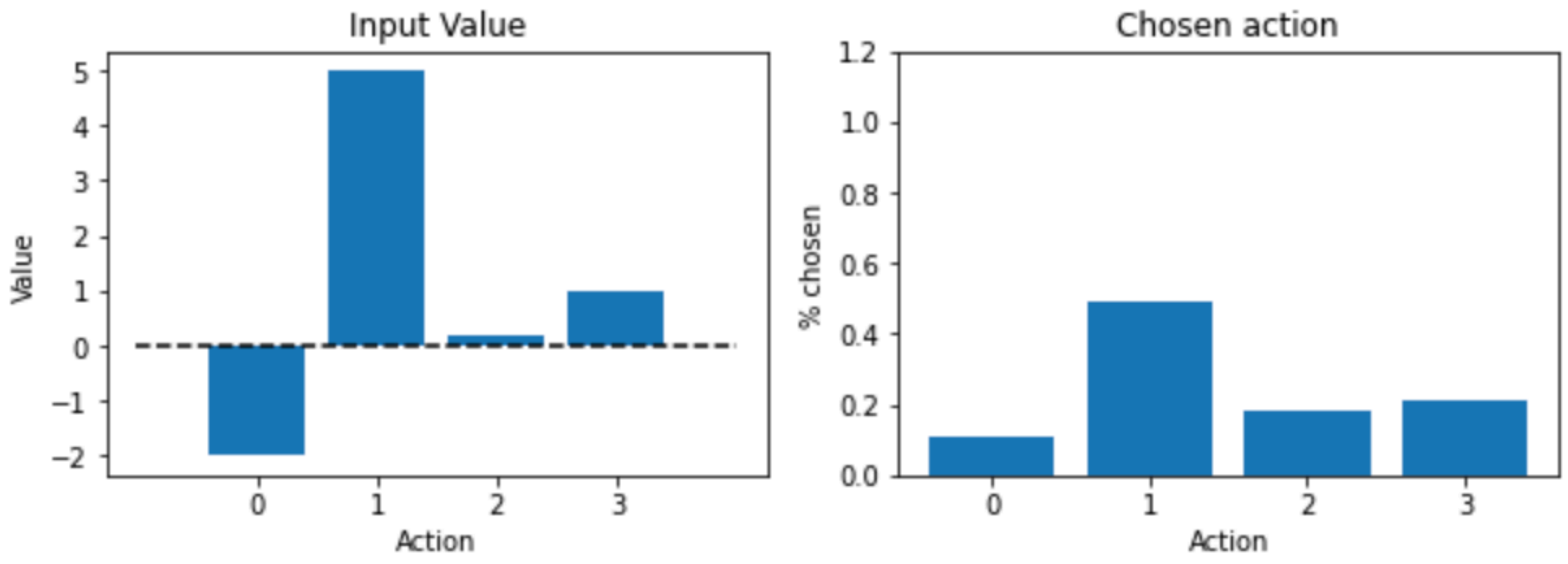
Softmax action selection:

$$P(a_t = a) = \frac{e^{V_t(a)\cdot\beta}}{\sum_{i=1}^{N} e^{V_t(a_i)\cdot\beta}}$$



Action is governed by a **policy**:

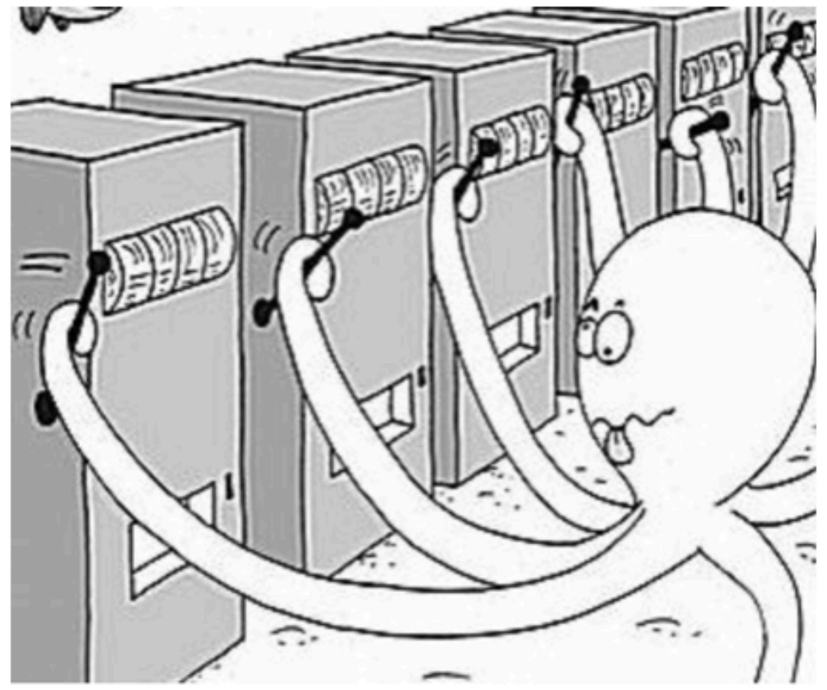
$$\pi(a, s) = P(a_t = a \mid s_t = s)$$



Multi-armed bandits

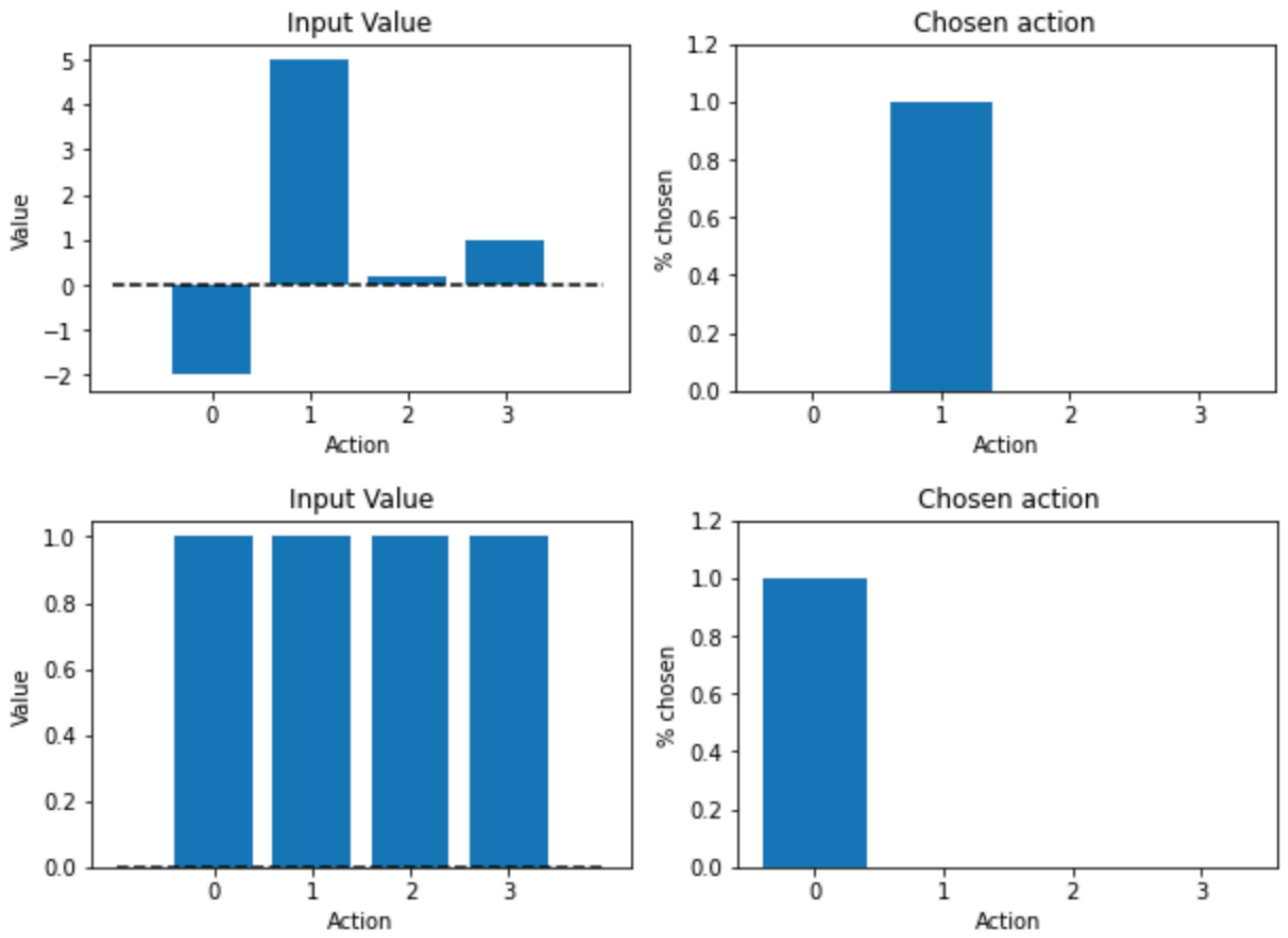
Upper-confidence-bound
(UCB) action selection:

$$P(a_t = a) = \text{argmax}_a[V_t(a) + c \cdot \sqrt{\frac{\ln t}{N_t(a)}}]$$



Action is governed by a **policy**:

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



Coding: Multi-Armed Bandits

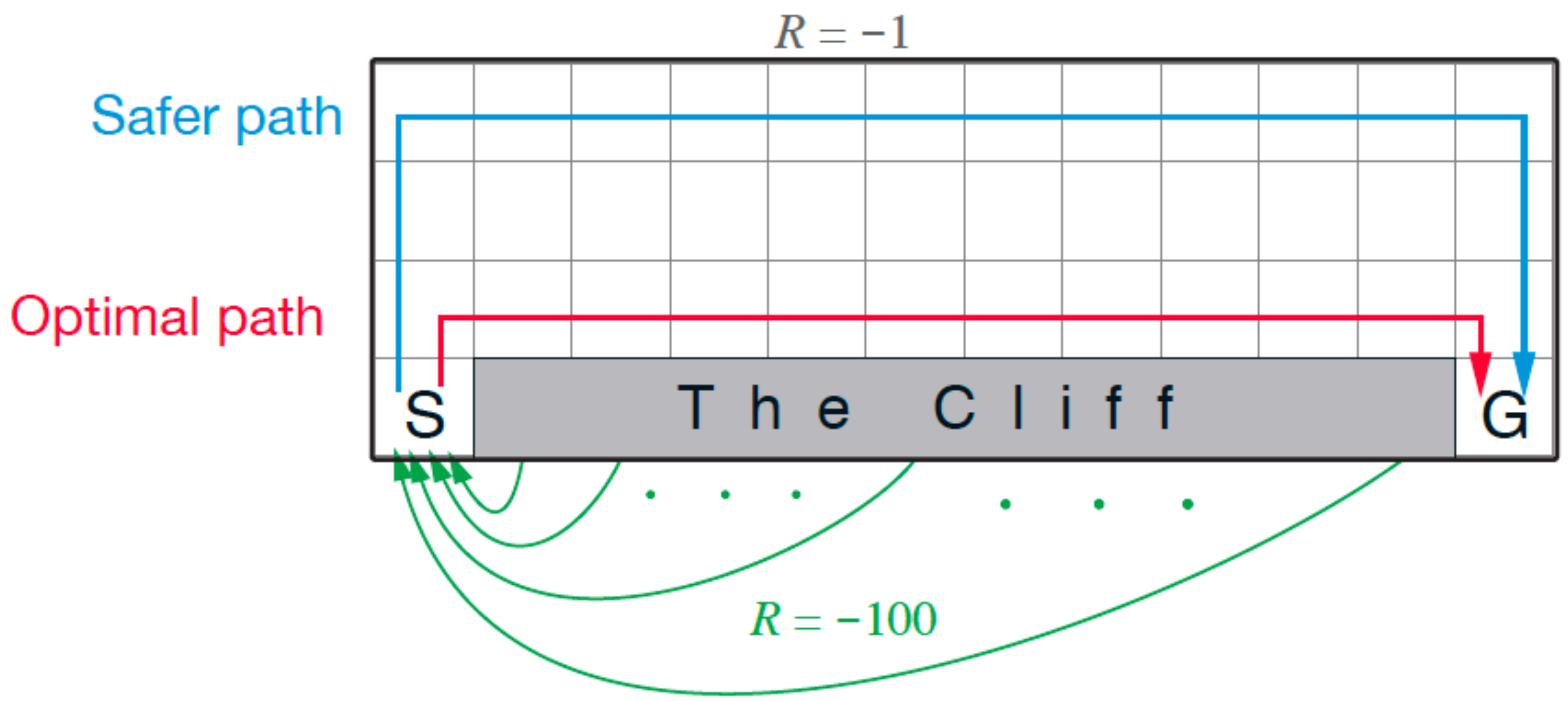
https://github.com/schwartenbeckph/RL-Course/tree/main/2022_06_21

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

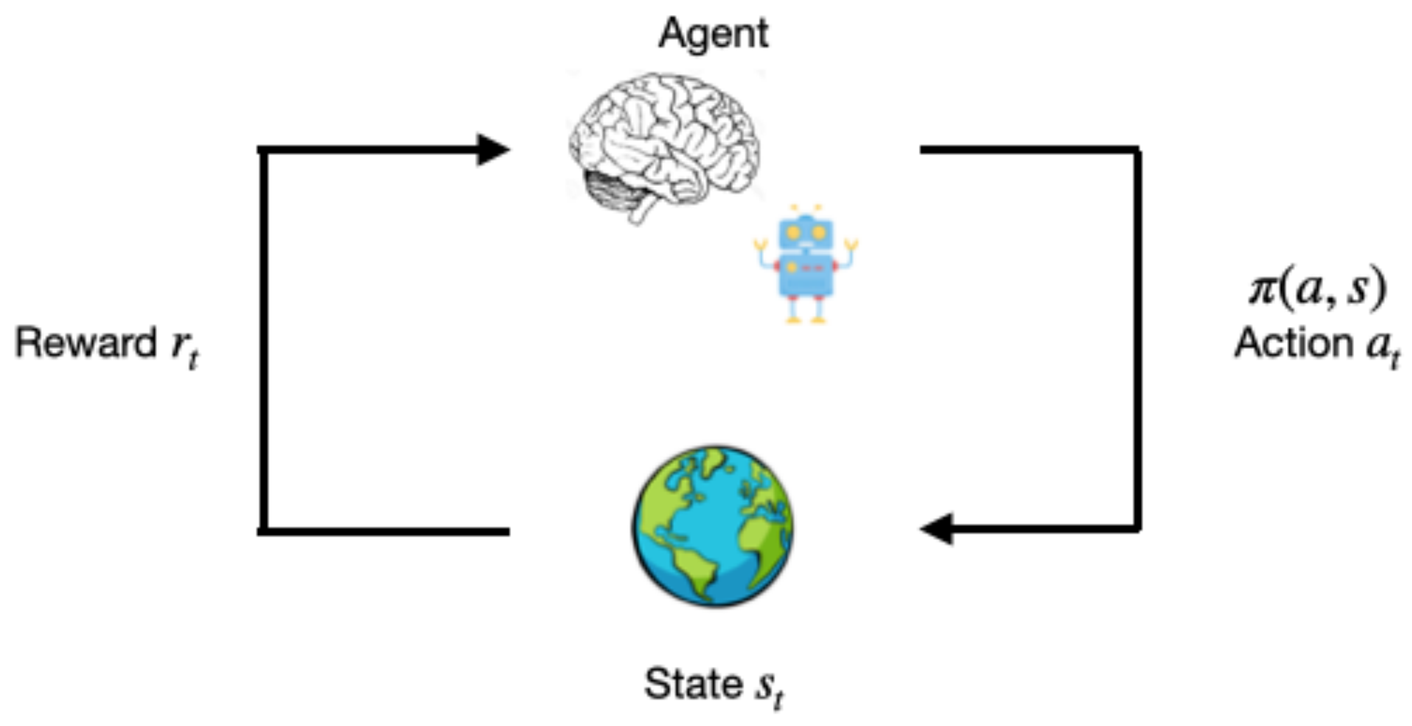
From classical to instrumental learning

TD Learning:

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

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_06_21