

# The Multi-Armed Bandits Problem

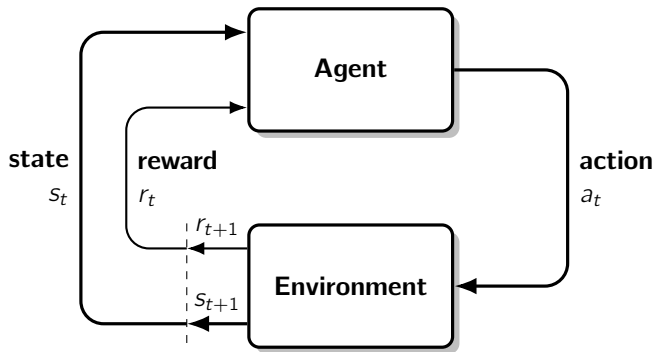
## Balancing Exploration and Exploitation

Nicole Orzan

November 25, 2021

## Recap

In Reinforcement Learning an agent learns how to best take actions in order to maximize some goal.



# Recap

## Key Concepts

- The problem of learning through interaction is framed as a **Markov Decision Process**  $\mathcal{M}\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r \rangle$
- The agent aims to learn a **policy**  $\pi : \mathcal{S} \rightarrow \mathcal{A}$
- The goal of the agent is to maximize the (discounted) **cumulative reward**  $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$ .
- We can find the optimal policy by making use of the **value functions**  $V^{\pi}(s)$  and  $Q^{\pi}(s, a)$

# Today's Agenda

- ① Multi-Armed Bandits
- ② Exploration-Exploitation Trade-off
- ③ Action-Value Methods
- ④ Action Preferences

# Simplified Setting

The environment has a **single state**.

- The setting is **non-associative**: actions can not change the state of the environment
- The actions can only impact the **next reward**

We want to learn a policy:

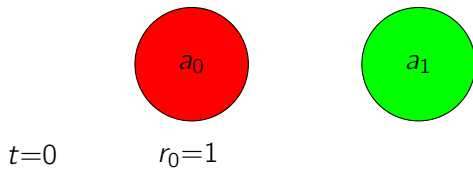
$$\pi(a) = \Pr \{a_t = a\}, \text{ for all } a \in \mathcal{A}.$$

# Multi-Armed Bandits

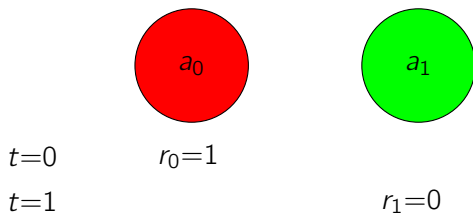
Our agent is repeatedly faced with the choice among  $k$  different actions. After each choice, gets a reward sampled from the stationary distribution that depends on the chosen action.

- Set of actions  $\mathcal{A}$
- Fixed distribution of rewards for each action,  $r_a \mid a \in \mathcal{A}$
- The goal is to maximize the expected cumulative reward
- We need to learn a policy  $\pi(a) \mid a \in \mathcal{A}$

# Example

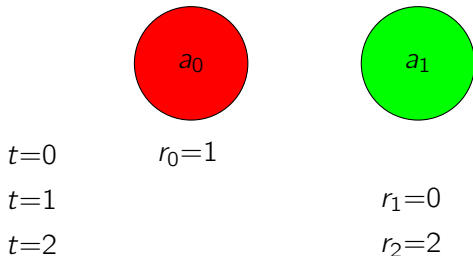


## Example





## Example



How do we choose whether to stick to the current action, or keep exploring?

# Exploration-Exploitation Trade-off

To maximize its reward in this environment, our agent must learn what the best action to take is.

- **Exploration** allows to gain knowledge of its actions, useful for the **long-term**
- **Exploitation** allows to use the current knowledge of the actions to get an **instantaneous benefit**

# Action-Value Methods

The **action value** for action  $a$  is the reward we expect to receive by taking that action:

$$Q^*(a) \doteq \mathbb{E}[R_t | a_t = a]$$

The goal is to find the **optimal value**, by maximizing the expected cumulative reward:

$$a^* = \arg \max_a \mathbb{E}[R_t | a_t = a]$$

**Regret** of an action  $a$ :

$$\Delta_a = Q^*(a^*) - Q^*(a)$$

# Action-Value Methods

In Reinforcement Learning we do not know which actions have the best value: we need to learn it.

## Action Value Estimate

We denote the estimated value of action  $a$  at time step  $t$  as  $Q_t(a)$ .

We can use action value estimates to learn a policy.

## Sample-Average Method

The simplest estimate is given by the average of the sampled rewards:

$$\begin{aligned} Q_t(a) &\doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} \\ &= \frac{\sum_{i=0}^{t-1} R_i \cdot \mathbb{I}(A_i = a)}{\sum_{i=0}^{t-1} \mathbb{I}(A_i = a)} \end{aligned}$$

$\mathbb{I}$  is the **indicator function**:  $\mathbb{I}(\text{True}) = 1$ , and  $\mathbb{I}(\text{False}) = 0$ .

## Incremental Update

The sample-average method can be computed incrementally:

$$\begin{aligned}Q_{t+1}(a) &= \frac{1}{t} \sum_{i=0}^t R_i \\&= \frac{1}{t} \left[ R_t + (t-1) \frac{1}{t-1} \sum_{i=0}^{t-1} R_i \right] \\&= \frac{1}{t} \left[ R_t + (t-1) Q_t(a) \right] \\&= Q_t(a) + \frac{1}{t} \left[ R_t - Q_t(a) \right]\end{aligned}$$

## Incremental Update

$$Q_{t+1}(a) := Q_t(a) + \alpha_t(R_t - Q_t(a)) \quad \text{where} \quad \alpha_t = \frac{1}{t}$$

### Update Rule

$$\textit{NewEstimate} \leftarrow \textit{OldEstimate} + \textit{StepSize} [\textit{Target} - \textit{OldEstimate}]$$

We can also consider other possible step sizes  $\alpha_t$ , but they need to respect the following conditions:

$$\sum_{t=0}^{\infty} \alpha_t = \infty \qquad \sum_{t=0}^{\infty} \alpha_t^2 < \infty$$

## Action Selection: greedy

Action-value estimates can be used to select actions.

We can exploit the current knowledge by selecting the **greedy** action, which is the action with the highest estimated value:

$$A_t = \arg \max_{a \in \mathcal{A}} Q_t(a)$$

**Downside:** we can get stuck taking a **suboptimal action** forever, because we are not exploring better opportunities.



## Action Selection: $\epsilon$ -greedy

- Behave **greedily** with probability  $1 - \epsilon$
- Take a **random action** with probability  $\epsilon$

Where  $\epsilon$  is a small, positive number  $0 < \epsilon < 1$ .

$$\pi_t(a) = \begin{cases} (1 - \epsilon) + \epsilon/|\mathcal{A}| & \text{if } a = \arg \max_{a \in \mathcal{A}} Q_t(a) \\ \epsilon/|\mathcal{A}| & \text{otherwise} \end{cases}$$

# Optimistic Initial Values

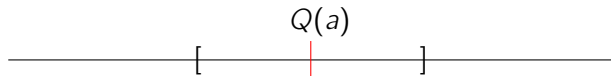
All the methods discussed up to now depend on the **initial values** of the action estimates,  $Q_0(a) \mid a \in \mathcal{A}$ .

Optimistic initial values can be used to **encourage early exploration**:

- For any starting action taken by the agent, the reward is smaller than the starting estimates
- The agent, disappointed, switches to other actions
- Actions are tried several times before the value estimates converge

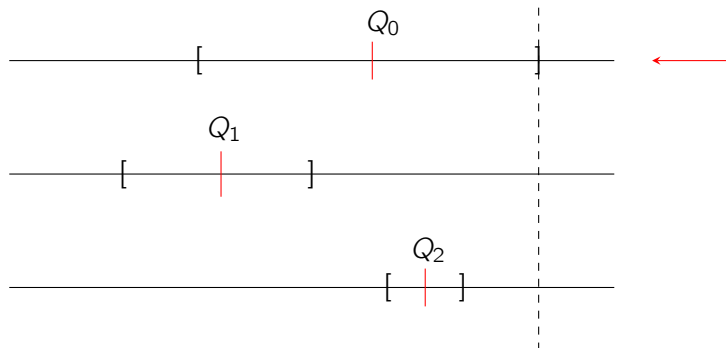
# Uncertainty in Action Values Estimates

If we could explicit the **uncertainty** of our action-values estimates, we could select actions in a better way.

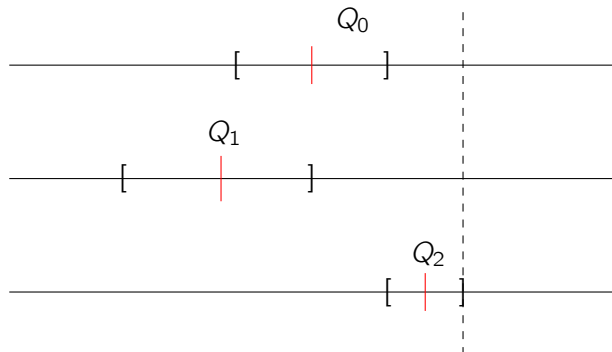


Select actions by leveraging their uncertainty: if we are uncertain about the value of an action, we **optimistically assume that it is good**.

# Optimism in the Face of Uncertainty



# Optimism in the Face of Uncertainty



# Upper-Confidence Bound (UCB)

We want to estimate an upper confidence bound for every action value:

$$Q^*(a) \leq Q_t(a) + U_t(a)$$

And select **greedily** the action maximizing the UCB:

$$a_t \doteq \arg \max_{a \in \mathcal{A}} [Q_t(a) + U_t(a)]$$

## Upper-Confidence Bound (UCB)

$$a_t \doteq \arg \max_{a \in \mathcal{A}} \left[ Q_t(a) + c \sqrt{\frac{\ln(t)}{N_a(t)}} \right]$$

- Each time we select  $a$ ,  $N_a(t)$  is increased and therefore the uncertainty is reduced
- Each time we select  $a' \neq a$ ,  $t$  increases but  $N_a(t)$  does not, therefore the uncertainty increases

The parameter  $c > 0$  regulates the amount of exploration.

## Action Preferences

All methods described until now rely on the estimate of action values to learn a policy. Another approach relies on learning a preference for each action,  $H_t(a)$ .

- The larger the preference, the more probably a certain action should be selected
- The preference has no meaning in terms of reward

### Boltzmann distribution

From the action preferences, we can define a policy using the Softmax or Boltzmann distribution:

$$\pi(a) = \frac{e^{H_t(a)}}{\sum_{k=0}^N e^{H_t(k)}}$$

**Note:** You will use this distribution in deep learning when it comes to multi-class classification problems!



# Action Preferences

If at time step  $t$  we choose action  $a'$ , we can update the action preferences as:

$$\begin{aligned} H_{t+1}(a') &\doteq H_t(a') + \alpha (r_t - \bar{r}_t)(1 - \pi_t(a')) \\ H_{t+1}(a) &\doteq H_t(a) - \alpha (r_t - \bar{r}_t) \pi_t(a) \text{ if } a \neq a' \end{aligned}$$

Where  $r_t$  is the reward sampled at time step  $t$  and  $\bar{r}_t$  is the average of all the rewards gathered until time step  $t$ .

The above formula is based on the idea of [stochastic gradient ascent](#).

# Associative Search: Contextual Bandits

Until now, we considered contexts where the agent has to learn how to behave in a **single situation**. However, the general problem of Reinforcement Learning deals with learning how to behave in a number of **different situations**.

# Associative Search: Contextual Bandits

Simple way to extend bandits:

- You face  $N$  different Multi-Armed bandit problems
- This is an **associative task**:  $\pi : \mathcal{S} \rightarrow \mathcal{A}$
- Similar to the full Reinforcement Learning problem as they involve **learning a policy**, but like the k-armed bandit problem in that each action affects only the **immediate reward**.

# Final Slide!

## Lecture Takeaway

1. Multi-Armed bandit is a simplified setting with a single state
2. To learn a policy we need to balance Exploration and Exploitation
3. We can learn the policy by leveraging action-values estimates  $Q(a)$
4. Or by leveraging action preferences  $H(a)$