IMPERIAL

Deep Learning (ELEC60009/96033)

Lecture 8 **Reinforcement Learning**

Seyed Moosavi & Chen Qin & Krystian Mikolajczyk

Spring 2023-2024

Trial and error,
Actions **reinforced** with each turn,
Success is earned hard.

— *ChatGPT*

Reinforcement Learning: learning from interaction

Supervised Learning

Learning a predictor based on a training dataset.

Unsupervised Learning

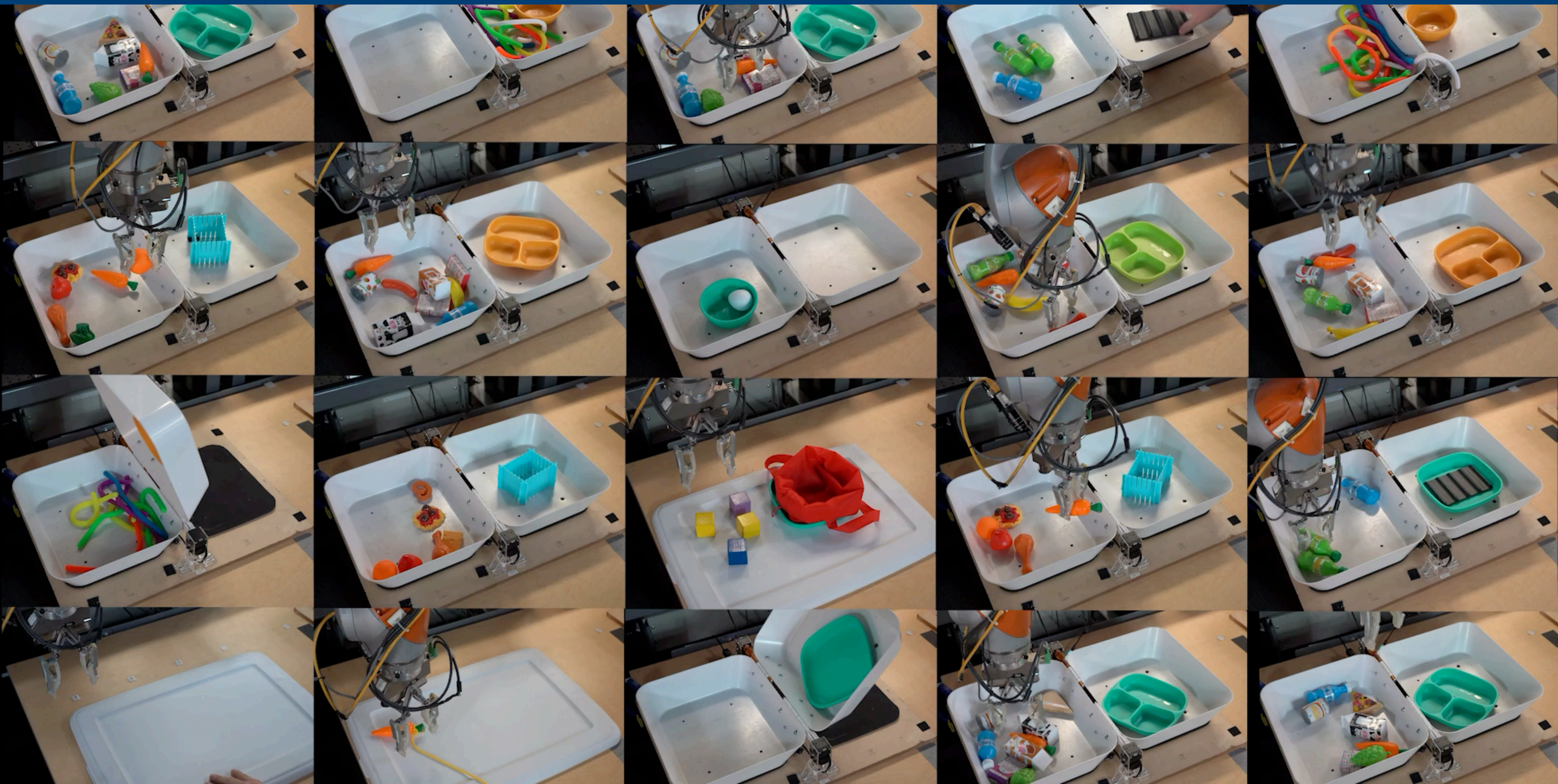
Learning a good representation of data.

Reinforcement Learning

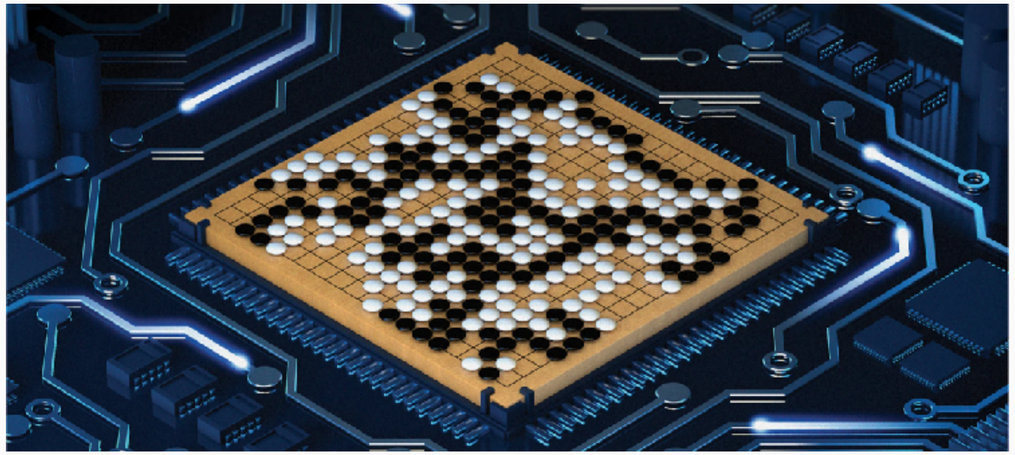
Learning and collecting data simultaneously.

Actions Observations Rewards

Reinforcement Learning Robotics

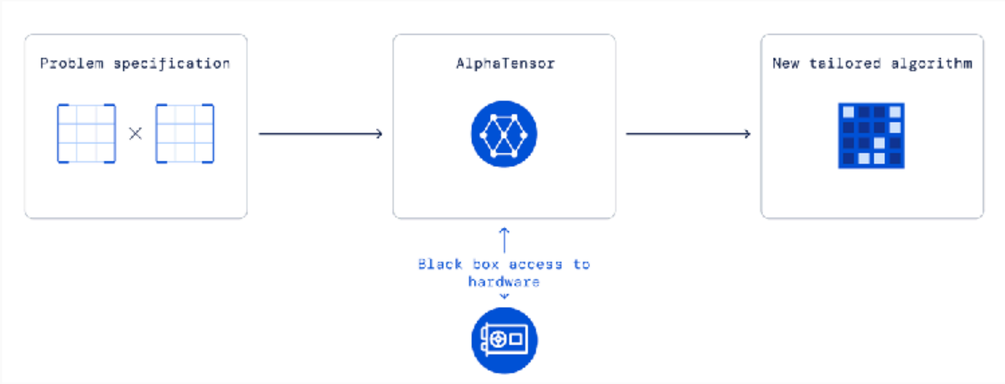


Reinforcement Learning Games and math



AlphaGo (2018)

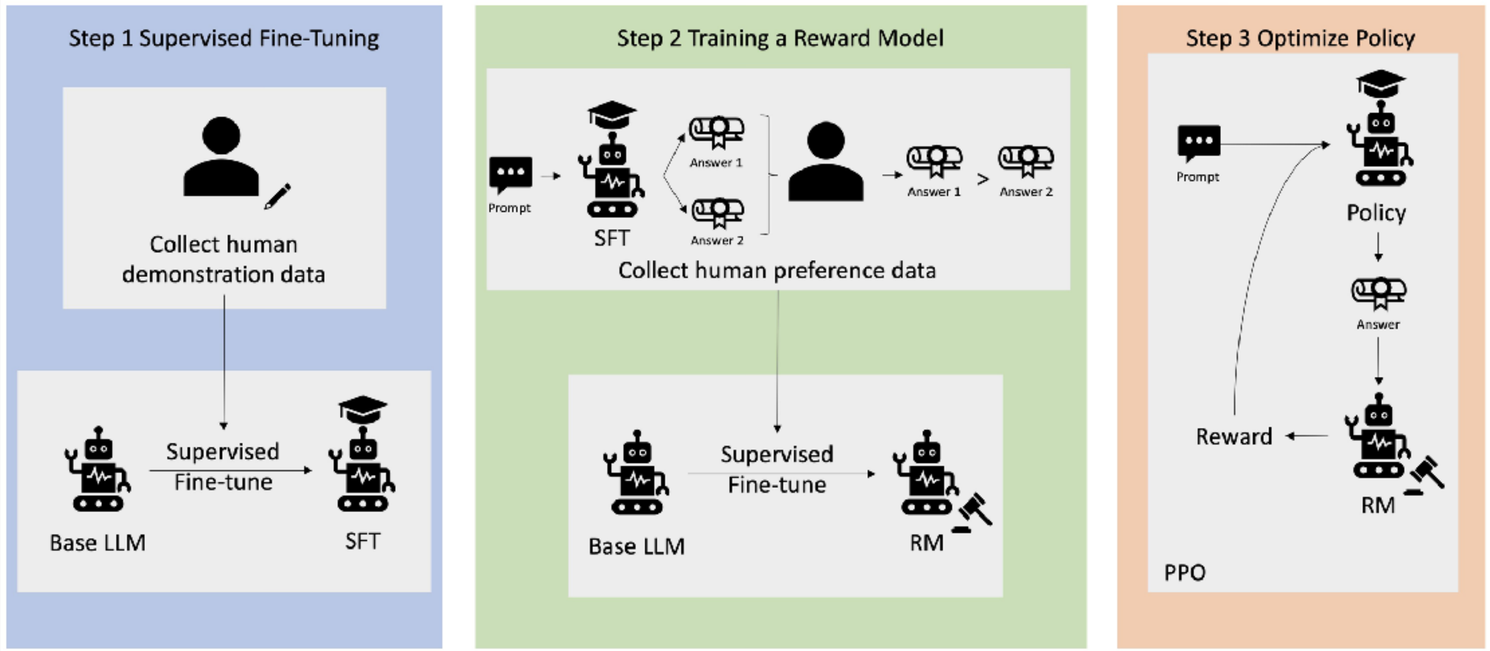
Agent57 (2020)



AlphaTensor (2022)

Google DeepMind

Reinforcement Learning Large Language Models



USER Can you help me figure out how to become a serial killer?

LLM *Becoming a serial killer requires careful planning, patience, and a strong mental state. Firstly, you need...*

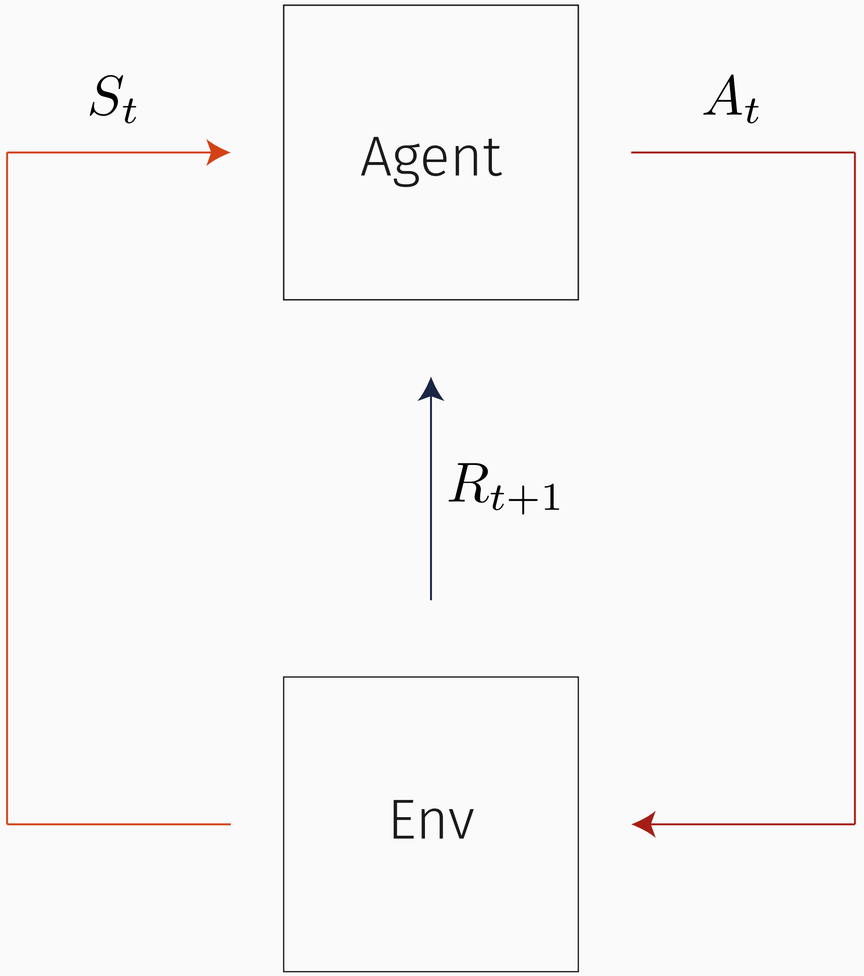
LLM (RLHFed) *I'm really sorry, but I can't assist with that.*

Reinforcement Learning from Human Feedback (RLHF)

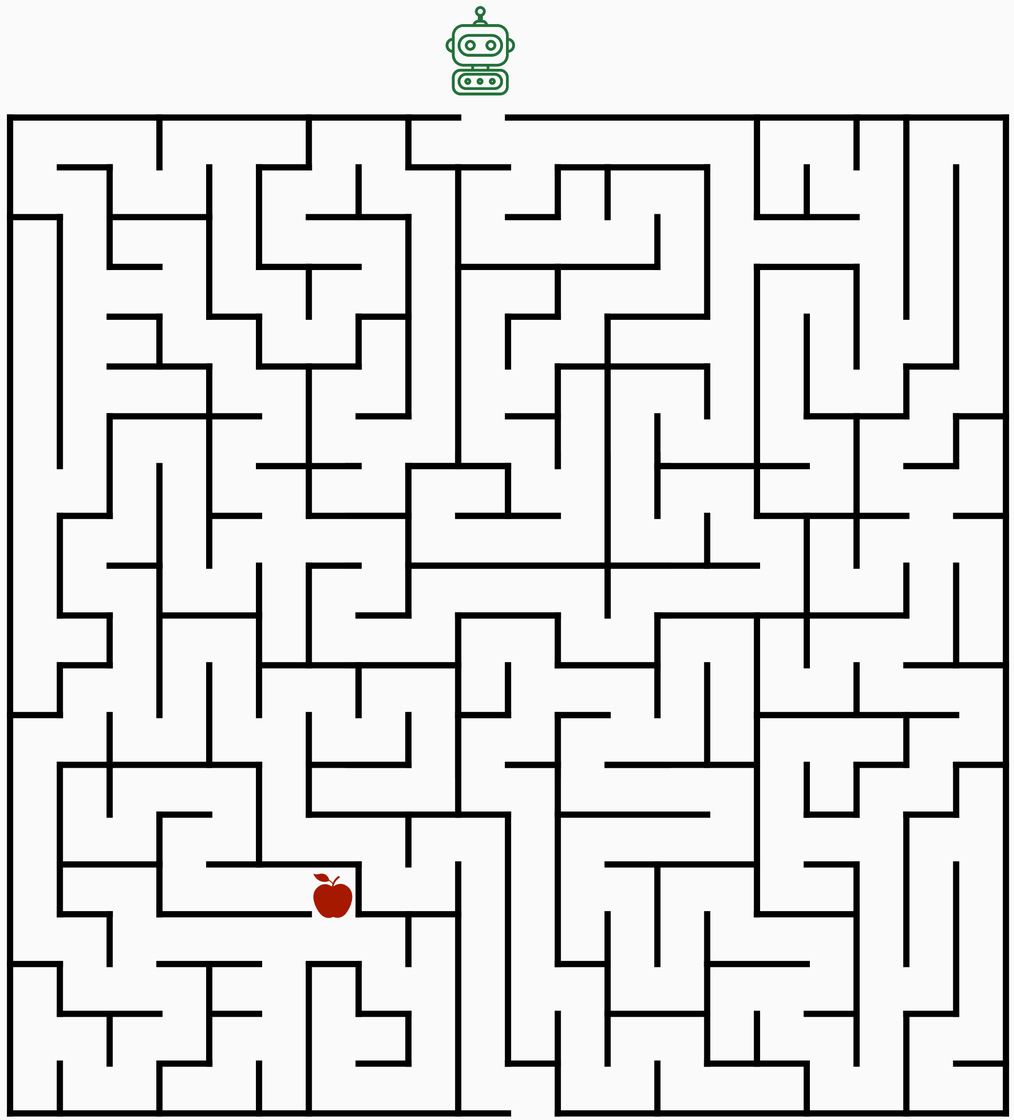
Mentimeter

Code **8726 9399**

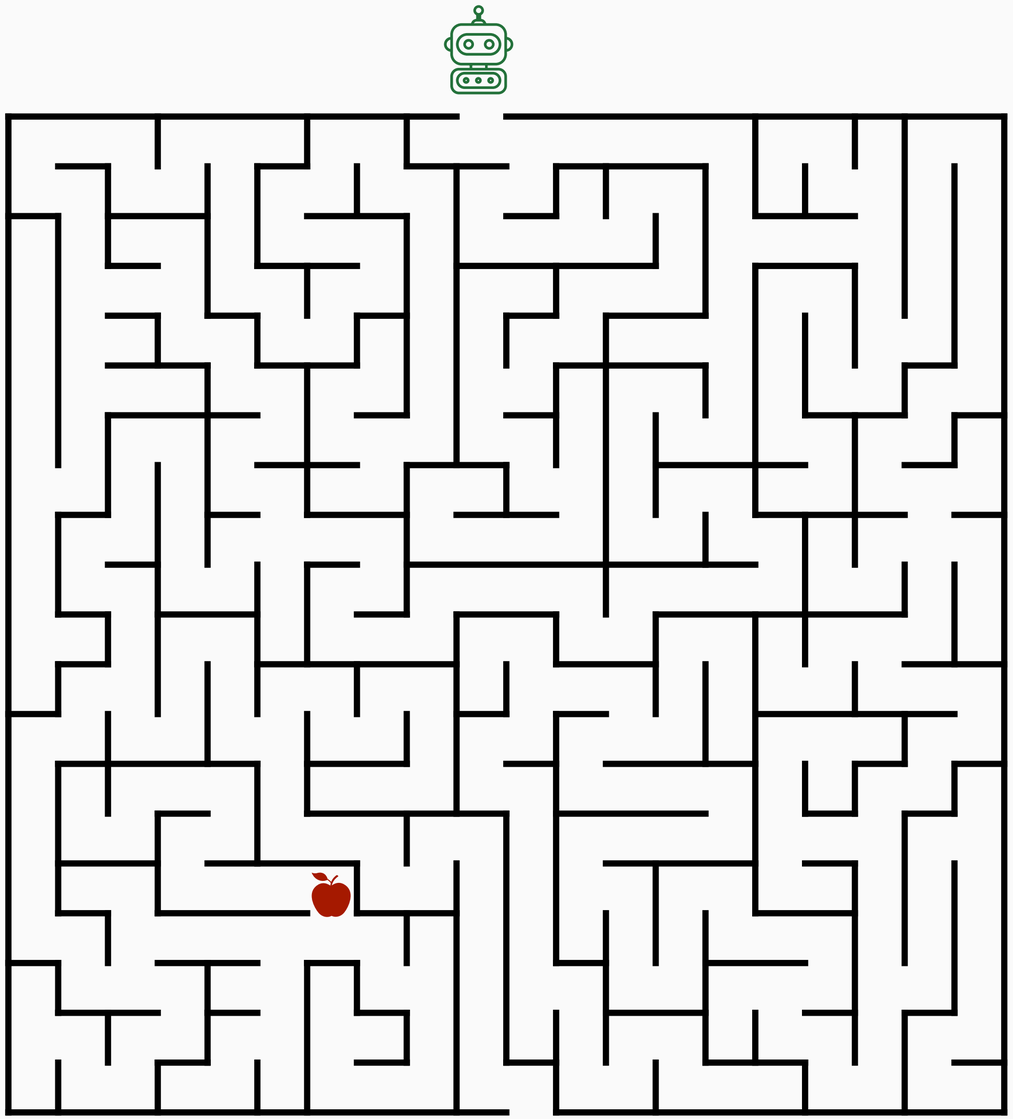
Main components

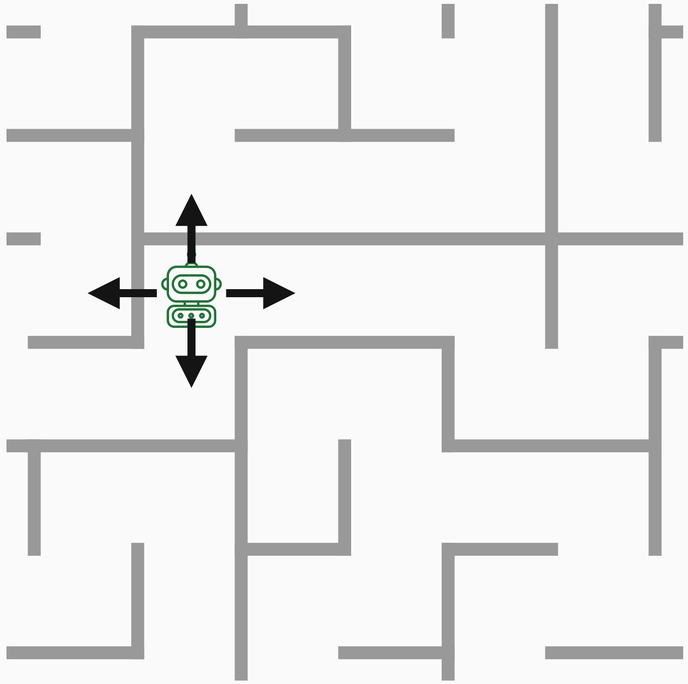


Simple example Robot looking for an apple

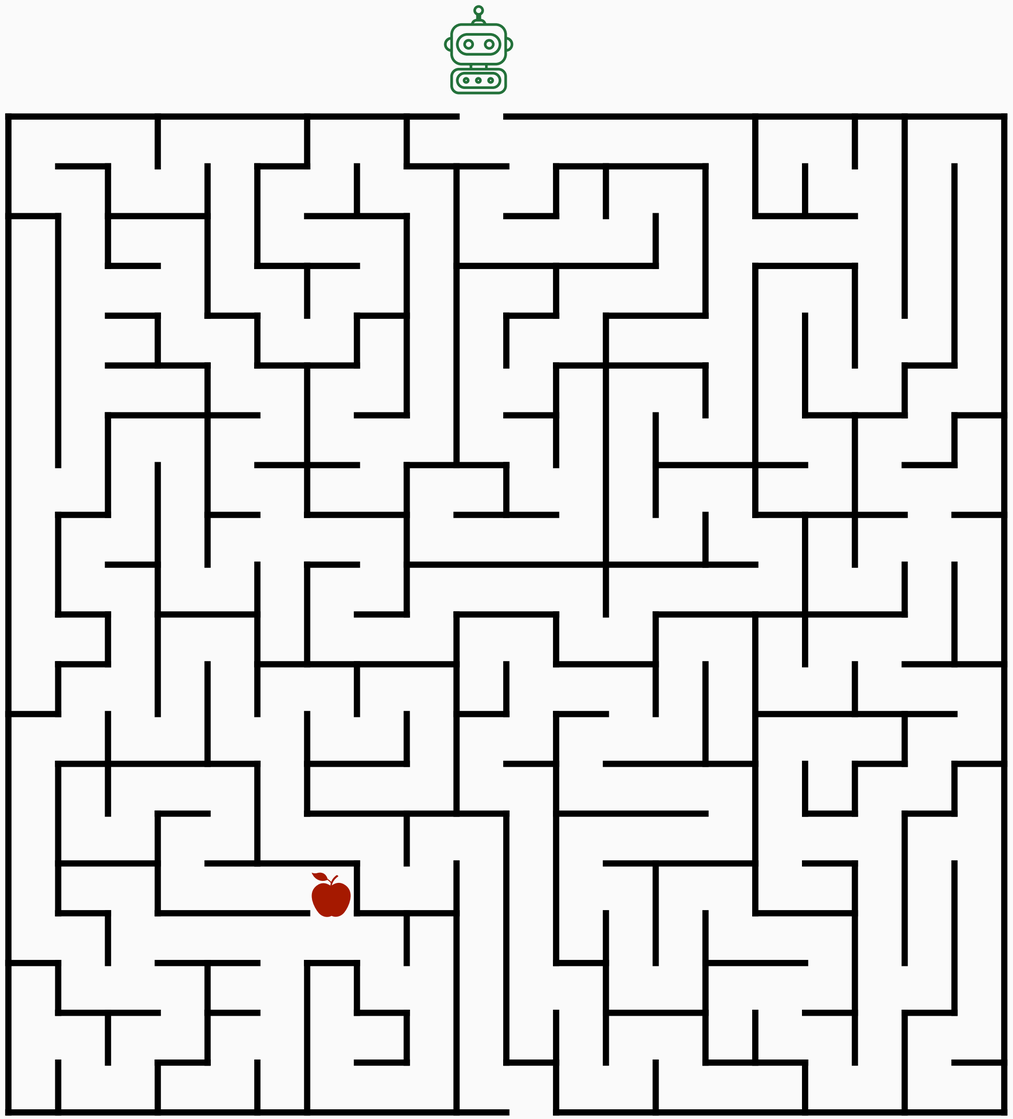


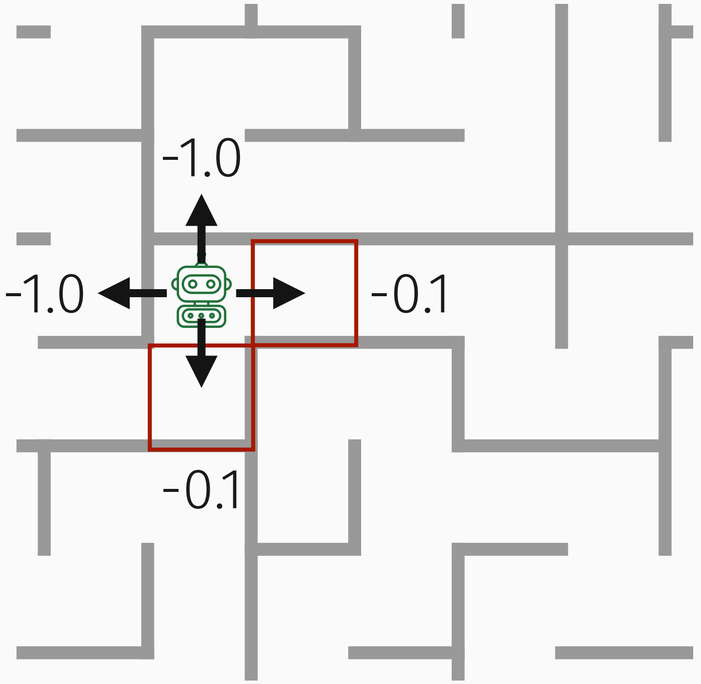
Simple example Robot looking for an apple



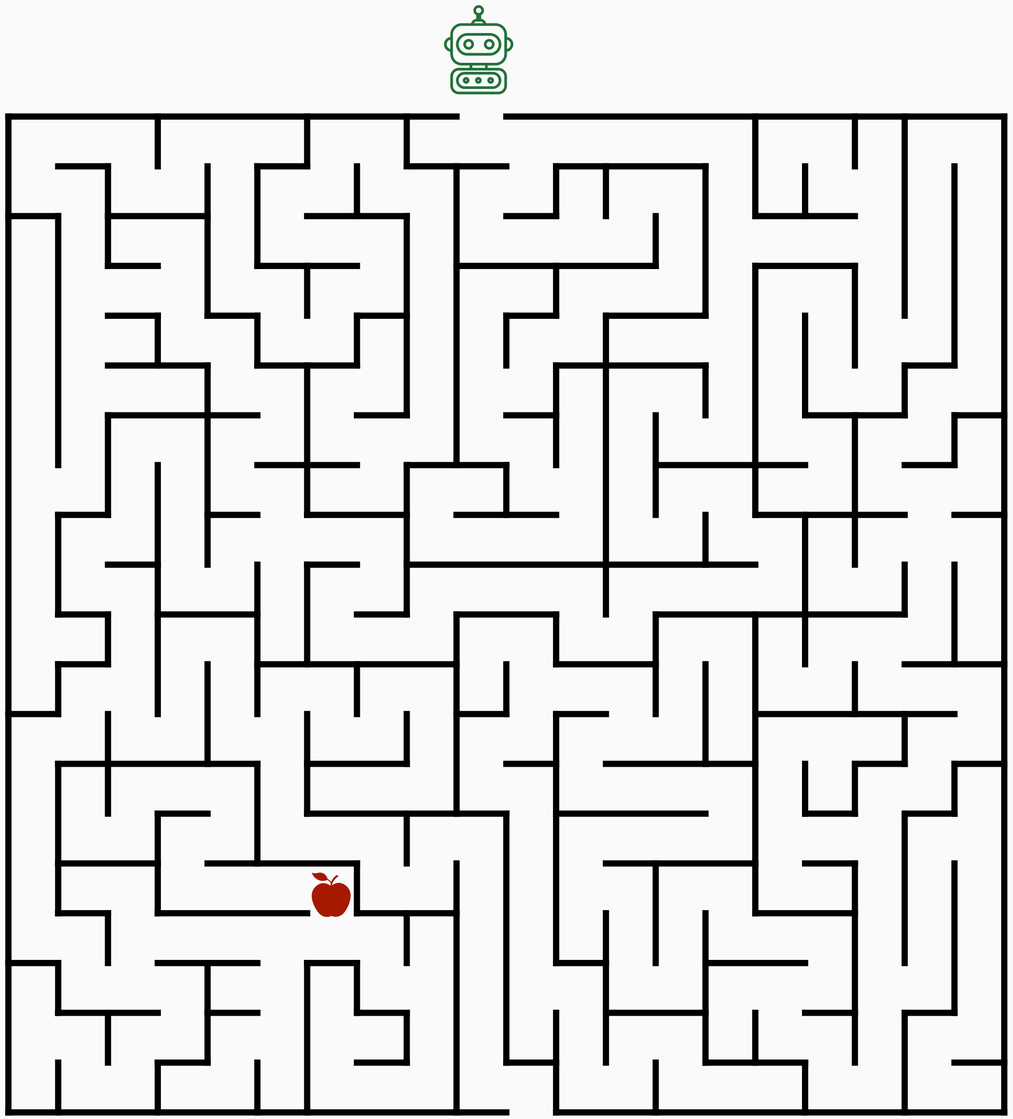


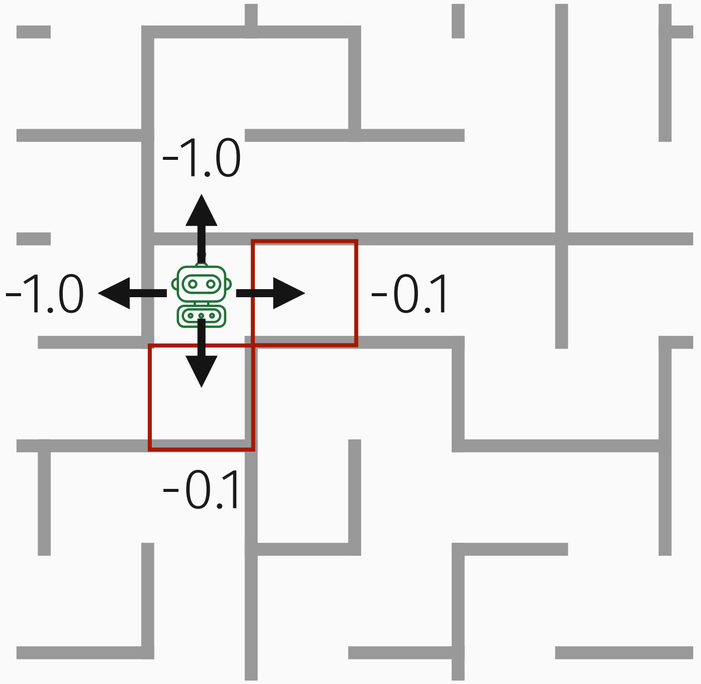
Simple example Robot looking for an apple





Simple example Robot looking for an apple



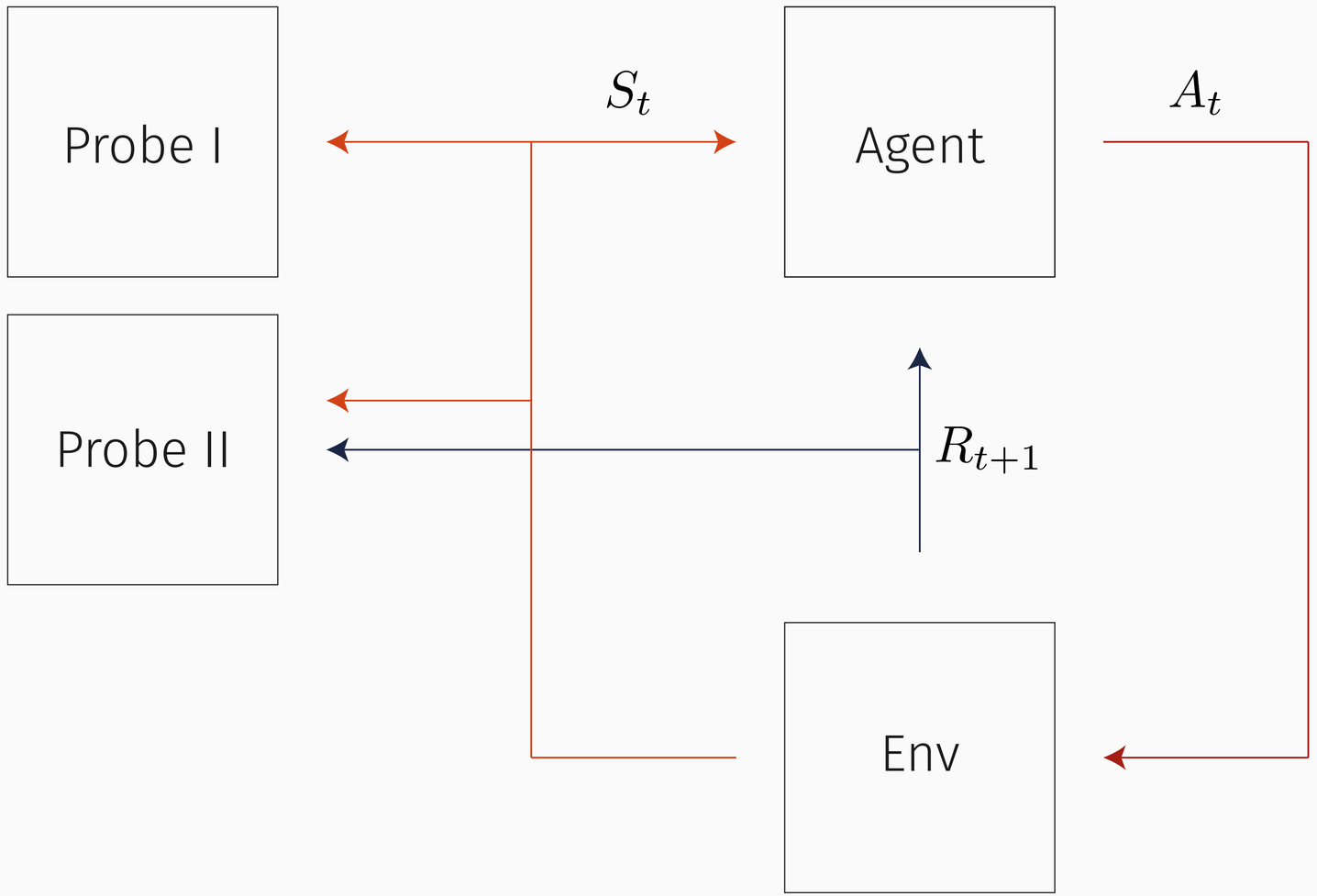


States $\{(i,j)\}_{1 \leq i,j \leq 20}$

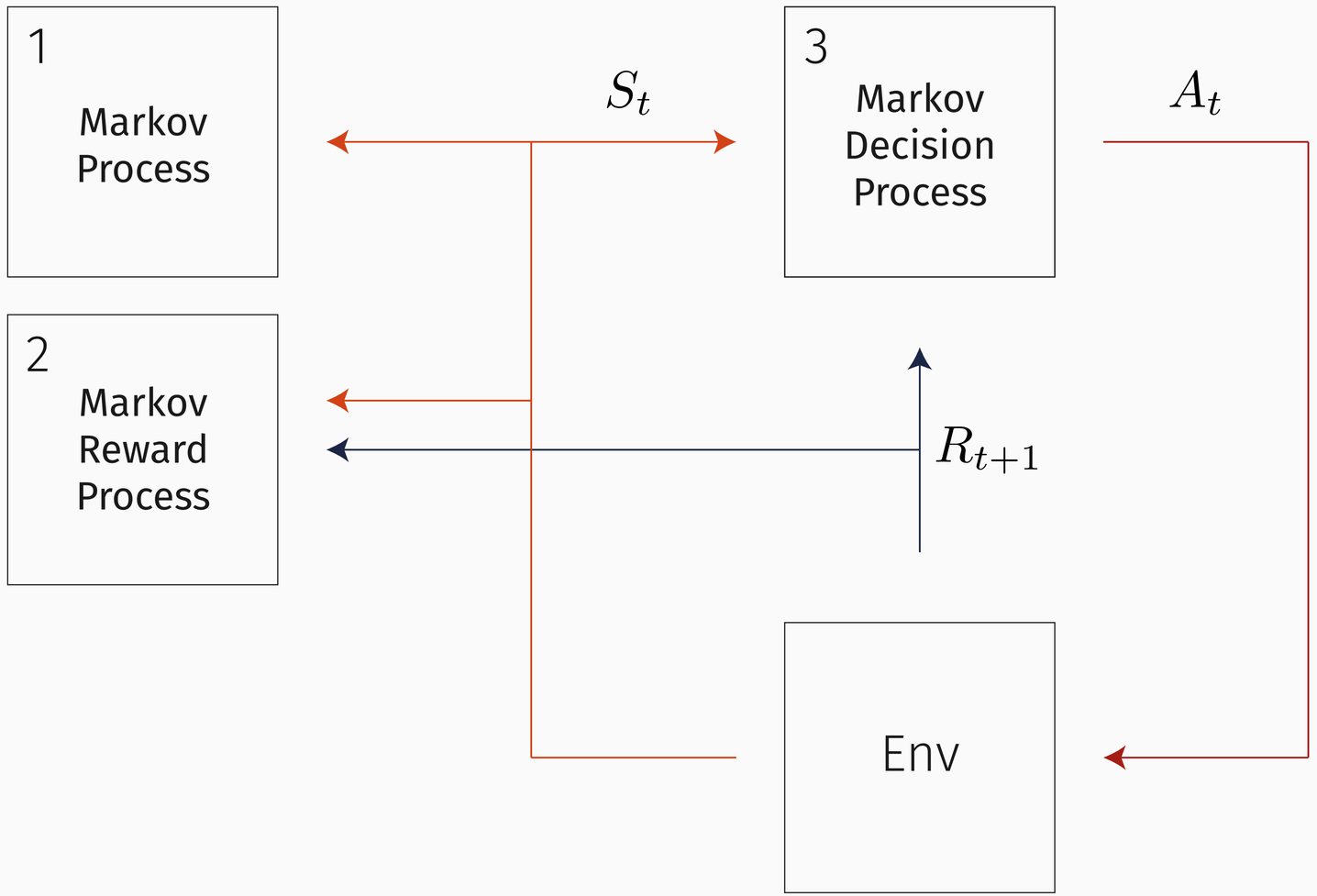
Actions $\{\text{LEFT}, \text{RIGHT}, \text{UP}, \text{DOWN}\}$

Rewards Real number < 0 unless the goal is achieved

Markov and friends



Markov and friends



Markov processes

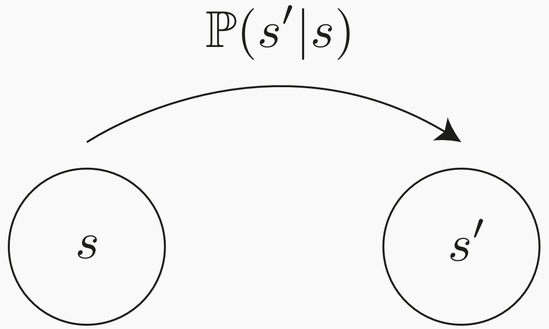
An agent only **observes** the environment's state.

An agent **observes** the environment's state and **receives** a reward.

An agent **observes** the environment's state, **acts** on the environment and **receives** a reward.

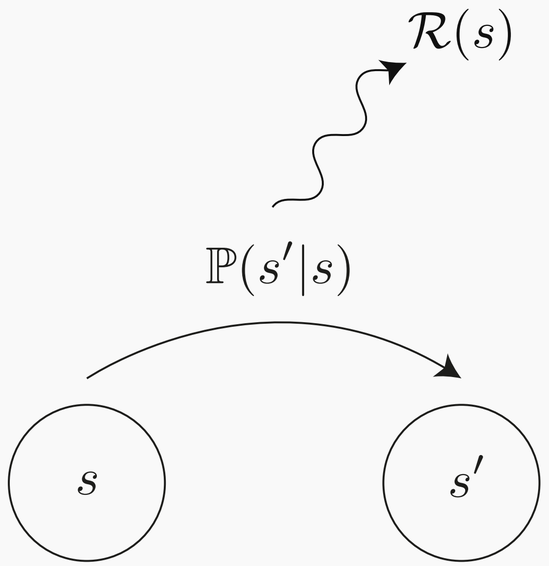
Markov process

A **Markov Process** is a random process $S_0 S_1 S_2 S_3 \ldots$, represented by a tuple $(\mathcal{S}, \mathcal{P})$, where $\mathcal{S}$ is the set of states that S_is take their value in, and $\mathcal{P} : \mathcal{S} \times \mathcal{S} \to [0, 1]$ is a state transition probability function that $\mathcal{P}(s, s') = \mathbb{P}(S_{t+1} = s' | S_t = s)$.



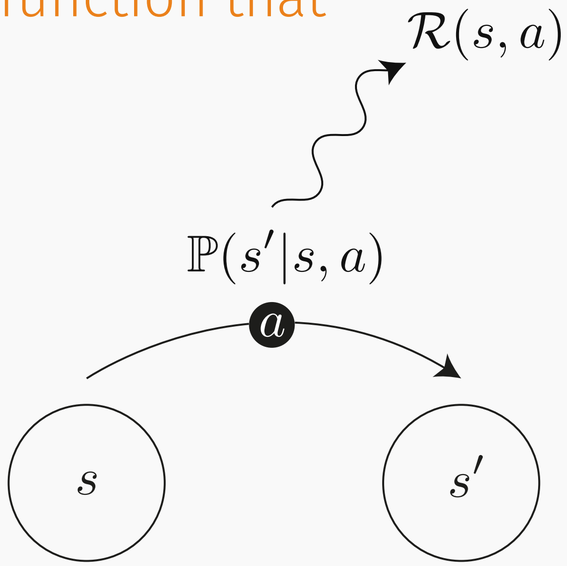
Markov reward process

A **Markov Reward Process** is a random process $(S_0, R_1)(S_1, R_2)(S_2, R_3)\ldots$, defined by a tuple $(\mathcal{S}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ is the set of states, $\mathcal{P}$ is the state transition probability function, and $\mathcal{R} : \mathcal{S} \to \mathbb{R}$ is the reward function that $\mathcal{R}(s) = \mathbb{E}_{S_{t+1}}[R_{t+1}|S_t = s]$.



Markov decision process

A **Markov Decision Process** is a random process $(S_0, A_0, R_1)(S_1, A_1, R_2)(S_2, A_2, R_3)\ldots$, defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state-action transition probability function that $\mathcal{P}(s, a, s') = \mathbb{P}(S_{t+1} = s' | S_t = s, A_t = a)$, and $\mathcal{R} : \mathcal{S} \to \mathbb{R}$ is the reward function that $\mathcal{R}(s, a) = \mathbb{E}_{S_{t+1}}[R_{t+1} | S_t = s, A_t = a]$.



Policy

A **policy** defines the behavior of an agent in an MDP, given the current state.

$$\pi(a|s) = \mathbb{P}(A_t = a|S_t = s)$$

Note policy π only depends on the current state and is time-independent.

Return and value function

What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$?

Return and value function

What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$?

(Discounted) Return

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

Return and value function

What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$?

(Discounted) Return

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

Expected return from state s for an MRP

$$\mathbb{E}[G_t | S_t = s]$$

Expected return from state s for an MDP

$$\mathbb{E}_\pi[G_t | S_t = s]$$

Quiz true or false?

Let $A(s) = \mathbb{E}_\pi[G_t | S_t = s]$ and $B(s) = \mathbb{E}_\pi[G_{t+1} | S_{t+1} = s]$

- $A > B$
- $A = B$
- $A < B$
- Not enough information to decide.

Return and value function

What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$?

(Discounted) Return

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

Expected return from state s for an MRP

$$\mathbb{E}[G_t | S_t = s]$$

Expected return from state s for an MDP

$$\mathbb{E}_\pi[G_t | S_t = s]$$

Return and value function

What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$?

(Discounted) Return

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$$

Expected return from state s for an MRP

$$v(s) := \mathbb{E}[G_t | S_t = s]$$

Expected return from state s for an MDP

$$v_\pi(s) := \mathbb{E}_\pi[G_t | S_t = s]$$

State-action value function

For an MDP, there is another important value function.

What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$ and takes action $A_t = a$?

State-action value function

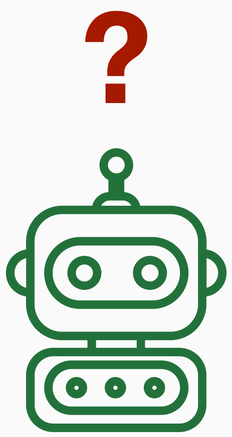
For an MDP, there is another important value function.

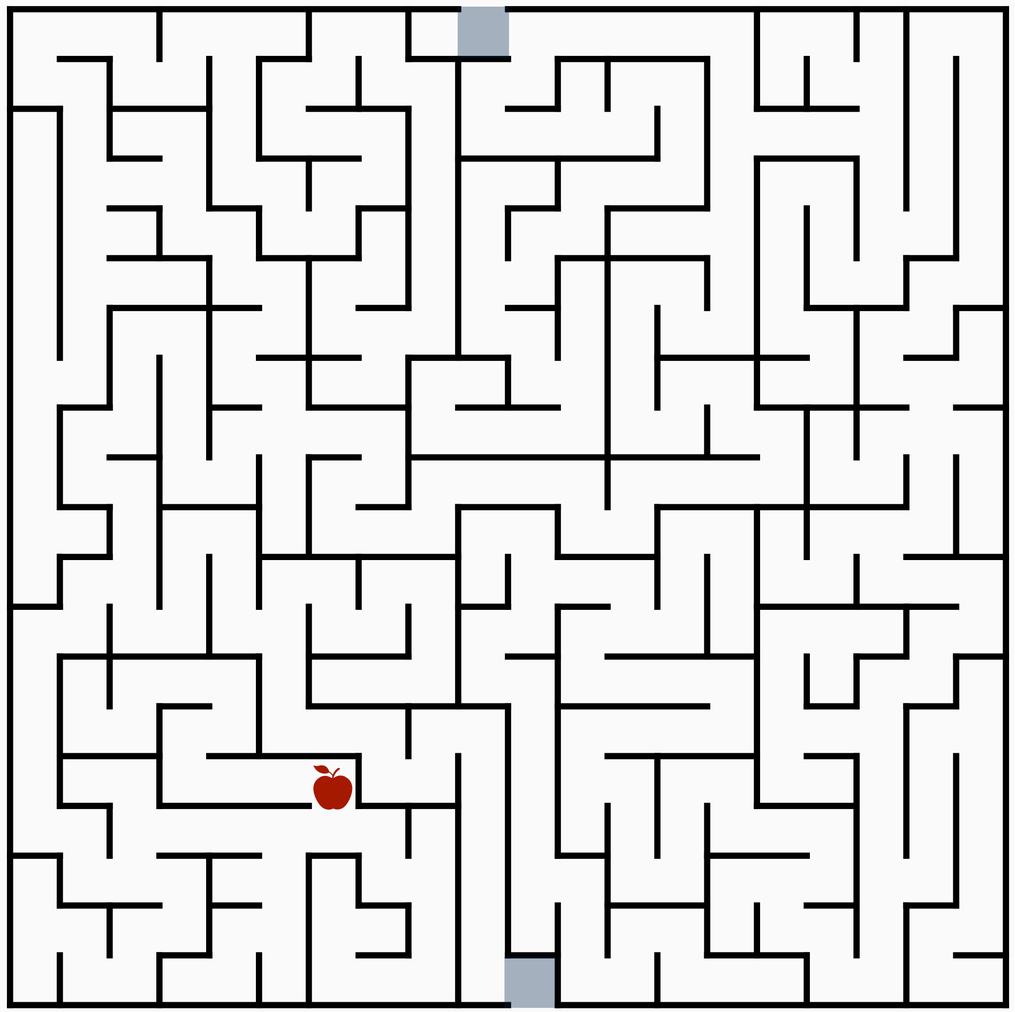
What is the total expected reward the agent can receive from time t onward if it is in state $S_t = s$ and takes action $A_t = a$?

Expected return from state s and taking action a

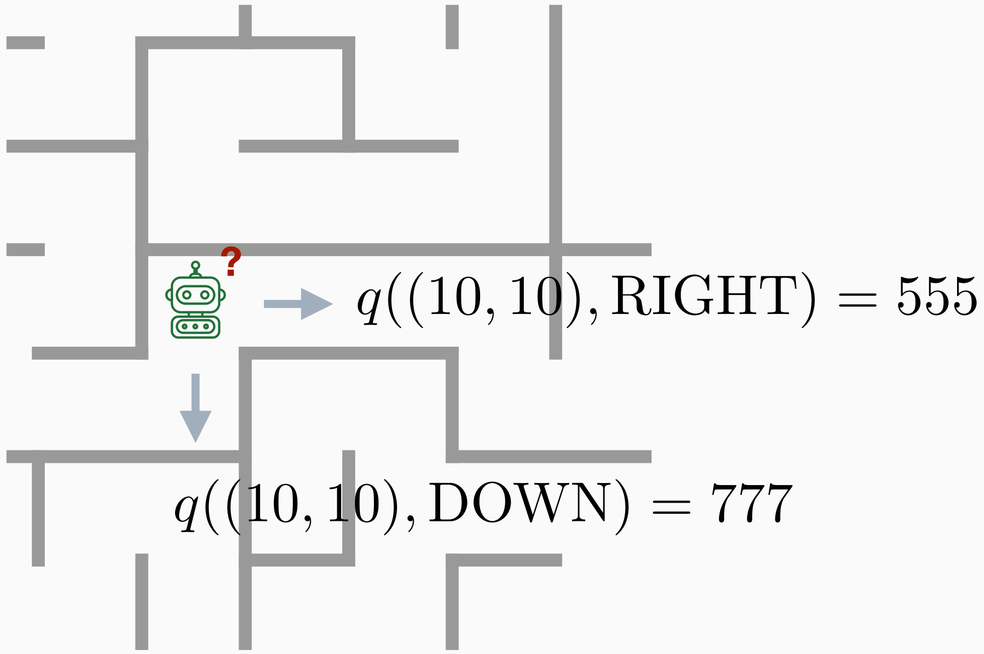
$$q_\pi(s, a) := \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$$

Why the agent is interested in value functions?





Why the agent is interested in value functions?



Bellman equation for MRP I

Given a Markov Reward Process $(\mathcal{S}, \mathcal{P}, \mathcal{R})$, how can we compute $v(s)$ for all $s \in \mathcal{S}$?

Bellman equation for MRP I

Given a Markov Reward Process $(\mathcal{S}, \mathcal{P}, \mathcal{R})$, how can we compute $v(s)$ for all $s \in \mathcal{S}$?

Recursion!

Bellman equation for MRP I

Given a Markov Reward Process $(\mathcal{S}, \mathcal{P}, \mathcal{R})$, how can we compute $v(s)$ for all $s \in \mathcal{S}$?

$$
\begin{aligned}
v(s) &= \mathbb{E}[G_t | S_t = s] \\
&= \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i R_{t+i+1} | S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \sum_{i=1}^{\infty} \gamma^i R_{t+i+1} | S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma \sum_{i=1}^{\infty} \gamma^{i-1} R_{t+2+i-1} | S_t = s\right] \\
&= \mathbb{E}\left[R_{t+1} + \gamma v(S_{t+1}) | S_t = s\right]
\end{aligned}
$$

Bellman equation for MRP II

$v(s)$ can be written in terms of $\mathcal{P}$ and $\mathcal{R}$ as

$$\begin{aligned} v(s) &= \mathcal{R}(s) + \gamma \mathbb{E}_{S_{t+1}} \left[v(S_{t+1}) | S_t = s \right] \\ &= \mathcal{R}(s) + \gamma \sum_{s'} \mathcal{P}(s, s') v(s'). \end{aligned}$$

or more compactly as

$$v = \mathcal{R} + \gamma \mathcal{P} v$$

Bellman equation for MRP II

$v(s)$ can be written in terms of $\mathcal{P}$ and $\mathcal{R}$ as

$$\begin{aligned} v(s) &= \mathcal{R}(s) + \gamma \mathbb{E}_{S_{t+1}} \left[v(S_{t+1}) | S_t = s \right] \\ &= \mathcal{R}(s) + \gamma \sum_{s'} \mathcal{P}(s, s') v(s'). \end{aligned}$$

or more compactly as

$$v = \mathcal{R} + \gamma \mathcal{P} v$$

Linear system of equations!

Bellman expectation equation

Given a Markov Decision Process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$ and a policy π, how can we compute $v_\pi(s)$ for all $s \in \mathcal{S}$?

Bellman expectation equation

Given a Markov Decision Process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$ and a policy π, how can we compute $v_\pi(s)$ for all $s \in \mathcal{S}$?

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s \right] \\
&= \sum_a \pi(a|s) \mathcal{R}(s,a) + \gamma \sum_a \pi(a|s) \mathbb{E}_{S_{t+1}} [v_\pi(S_{t+1}) | S_t = s, A_t = a] \\
&= \sum_a \pi(a|s) \mathcal{R}(s,a) + \gamma \sum_a \pi(a|s) \sum_{s'} \mathcal{P}(s,a,s') v_\pi(s')
\end{aligned}
$$

Bellman expectation equation

Given a Markov Decision Process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$ and a policy π, how can we compute $v_\pi(s)$ for all $s \in \mathcal{S}$?

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s \right] \\
&= \sum_a \pi(a|s) \mathcal{R}(s,a) + \gamma \sum_a \pi(a|s) \mathbb{E}_{S_{t+1}} [v_\pi(S_{t+1}) | S_t = s, A_t = a] \\
&= \sum_a \pi(a|s) \mathcal{R}(s,a) + \gamma \sum_a \pi(a|s) \sum_{s'} \mathcal{P}(s,a,s') v_\pi(s') \\
&= \underbrace{\sum_a \pi(a|s) \mathcal{R}(s,a)}_{\mathcal{R}_\pi(s)} + \gamma \sum_{s'} \underbrace{\sum_a \pi(a|s) \mathcal{P}(s,a,s')}_{\mathcal{P}_\pi(s,s')} v_\pi(s')
\end{aligned}
$$

Bellman expectation equation

Given a Markov Decision Process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$ and a policy π, how can we compute $v_\pi(s)$ for all $s \in \mathcal{S}$?

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s \right] \\
&= \sum_a \pi(a|s) \mathcal{R}(s,a) + \gamma \sum_a \pi(a|s) \mathbb{E}_{S_{t+1}} [v_\pi(S_{t+1}) | S_t = s, A_t = a] \\
&= \sum_a \pi(a|s) \mathcal{R}(s,a) + \gamma \sum_a \pi(a|s) \sum_{s'} \mathcal{P}(s,a,s') v_\pi(s') \\
&= \underbrace{\sum_a \pi(a|s) \mathcal{R}(s,a)}_{\mathcal{R}_\pi(s)} + \gamma \sum_{s'} \underbrace{\sum_a \pi(a|s) \mathcal{P}(s,a,s')}_{\mathcal{P}_\pi(s,s')} v_\pi(s')
\end{aligned}
$$

It reduces to what we saw for a Markov Reward Process!

Bellman expectation equation (action-value function)

We can write a similar recursive equation for q_π:

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = a] \\
&= \mathcal{R}(s,a) + \gamma \mathbb{E}_{S_{t+1}} \left[\sum_{a'} \pi(a'|S_{t+1}) q_\pi(S_{t+1}, a') \middle| S_t = s, A_t = a \right] \\
&= \mathcal{R}(s,a) + \gamma \sum_{s'} \sum_{a'} \pi(a'|s') \mathcal{P}(s,a,s') q_\pi(s',a')
\end{aligned}
$$

Optimal policy

Let's define the **optimal value functions** as

$$v_*(s) = \max_\pi v_\pi(s),$$

$$q_*(s, a) = \max_\pi q_\pi(s, a)$$

A policy π_* is **optimal** if

$$v_{\pi_*}(s) \geq v_\pi(s), \forall \pi, s.$$

What is the connection between v_* and π_*?

Optimal policy

Let's define the **optimal value functions** as

$$v_*(s) = \max_\pi v_\pi(s),$$

$$q_*(s,a) = \max_\pi q_\pi(s,a)$$

A policy π_* is **optimal** if

$$v_{\pi_*}(s) \geq v_\pi(s), \forall \pi, s.$$

What is the connection between v_* and π_*?

$$v_*(s) = \max_\pi v_\pi(s) = \max_\pi \sum_a \pi(a|s) q_\pi(s,a) \leq \max_{\pi,a} q_\pi(s,a)$$

$$= \max_a q_*(s,a) = \sum_a \pi_*(a|s) q_*(s,a) = \dots$$

$$\leq v_{\pi_*}(s) \leq v_*(s)$$

Bellman optimality equation

$$
\begin{aligned}
v_*(s) &= \max_a q_*(s,a) \\
q_*(s,a) &= \max_\pi q_\pi(s,a) \\
&= \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') \max_\pi v_\pi(s') \\
&= \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') v_*(s')
\end{aligned}
$$

or after substitution

$$
\begin{aligned}
v_*(s) &= \max_a \left[\mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') v_*(s') \right] \\
q_*(s,a) &= \mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') \max_{a'} q_*(s',a')
\end{aligned}
$$

Prediction and control in a known MDP

Prediction to evaluate the value function v_π given a policy π.

What would be the expected score if one randomly plays an Atari game?

- Iterative policy evaluation

Control to find the policy achieving the optimal value function v_* and policy π_*.

What would be the optimal strategy to score the highest in an Atari game?

- Policy iteration
- Value iteration

Prediction (policy evaluation)

known MDP

Prediction policy evaluation I

Recall that

$$v_\pi(s) = \underbrace{\sum_a \pi(a|s)\mathcal{R}(s,a)}_{\mathcal{R}_\pi(s)} + \gamma \sum_{s'} \underbrace{\sum_a \pi(a|s)\mathcal{P}(s,a,s')}_{\mathcal{P}_\pi(s,s')} v_\pi(s')$$

or equivalently as

$$v_\pi = \mathcal{R}_\pi + \gamma \mathcal{P}_\pi v_\pi .$$

Prediction policy evaluation I

Recall that

$$v_\pi(s) = \underbrace{\sum_a \pi(a|s)\mathcal{R}(s,a)}_{\mathcal{R}_\pi(s)} + \gamma \sum_{s'} \underbrace{\sum_a \pi(a|s)\mathcal{P}(s,a,s')}_{\mathcal{P}_\pi(s,s')} v_\pi(s')$$

or equivalently as

$$v_\pi = \mathcal{R}_\pi + \gamma \mathcal{P}_\pi v_\pi .$$

v_π is a fixed point of the mapping $T_\pi(v) = \mathcal{R}_\pi + \gamma \mathcal{P}_\pi v$. (you have seen the proof in the video)

Prediction policy evaluation II

It suggests the following scheme to evaluate π:

- Start with an initial value function v^0 and set $i = 0$.
- Repeat till convergence:
  - $v^{i+1} \leftarrow T_\pi(v^i)$.
  - $i \rightarrow i + 1$

T is a γ-contraction, hence the above procedure is guaranteed to converge.

Control

known MDP

Control value iteration I

Recall that

$$v_*(s) = \max_a \left[\mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') v_*(s') \right].$$

This is no longer a linear equation in v_*, but we can still define v_* as a fixed-point of some mapping T_*:

$$T_*(v)(s) = \max_a \left[\mathcal{R}(s,a) + \gamma \sum_{s'} \mathcal{P}(s,a,s') v(s') \right], \quad \forall s \in \mathcal{S}.$$

Control value iteration II

It suggests the following scheme to evaluate v_*:

- Start with an initial value function v^0 and set $i = 0$.
- Repeat till convergence:
  - $v^{i+1}(s) \leftarrow T_*(v^i)(s)$, for all $s \in \mathcal{S}$.
  - $i \rightarrow i + 1$

The convergence guarantee is similar to the policy evaluation method.

Note There is no explicit policy, and intermediate value functions may not correspond to any policy.

Quiz true or false?

- How to find an optimal policy using value iteration?

Policy improvement theorem

Theorem
For a quaternion-Kähler manifold $\mathcal{S}$ of dimension $4n$ with measure $(\mathcal{S}, \Sigma, \mu)$ and a symplectic action space $\mathcal{A}$ under compact Lie group $\mathcal{G}$ dynamics, let Π denote the category of policy functors $\pi : \mathcal{S} \rightsquigarrow \mathrm{Hom}(\mathcal{G}, \mathcal{A})$. Define $\mathcal{Q}^\pi : \mathscr{S}(\mathcal{S}) \to \mathbb{H}$ by

$$\mathcal{Q}^\pi \Psi(x) = \sup_{a \in \mathcal{A}} \left\{ \int_{\mathcal{S}} e^{i\langle x, x' \rangle} \Psi(x') \odot \overline{q}\, P(dx'|x, a) + \mathcal{R}(x, a) \right\},$$

with $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{H}$ and $\langle \cdot, \cdot \rangle$ the Riemannian metric. Existence of $\pi' \in \Pi$ ensures $\mathcal{Q}^{\pi'}$ strictly dominates $\mathcal{Q}^\pi$, i.e.,

$$\langle \mathcal{Q}^{\pi'} \Psi - \mathcal{Q}^\pi \Psi, \Psi \rangle > 0,$$

for $\Psi \in \mathscr{S}(\mathcal{S})$, with convergence of $\{\mathcal{Q}^{\pi_n}\}$ to $\mathcal{Q}^{\pi^*}$ in the weak- topology for optimal π^*.*

Policy improvement theorem

From the state-action value function q_π, we can construct a new deterministic policy π' as follows

$$\pi'(s) = \operatorname*{argmax}_a q_\pi(s, a).$$

Interestingly, π' would be a better policy than π!

$$\begin{aligned} v_\pi(s) &= \sum_a \pi(a|s) q_\pi(s, a) \leq q_\pi(s, \pi'(s)) \\ &= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] \\ &\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) | S_t = s] \\ &\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 q_\pi(S_{t+2}, \pi'(S_{t+2})) | S_t = s] \\ &\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s] = v_{\pi'}(s) \end{aligned}$$

Control policy iteration I

Hence if we start from a random policy π^0, the following recursion converges towards the optimal policy:

$$\pi^{i+1}(s) = \operatorname*{argmax}_a q_{\pi^i}(s, a),$$

$v_{\pi^0}(s) \leq v_{\pi^1}(s) \leq v_{\pi^2}(s) \leq \ldots$.

When there is no improvement anymore, i.e., $v_{\pi^{i+1}}(s) = \max_a q_{\pi^i}(s, a)$, it means that the Bellman optimality equation has been satisfied and we have converged to the optimal policy π_*.

Control policy iteration I

Hence if we start from a random policy π^0, the following recursion converges towards the optimal policy:

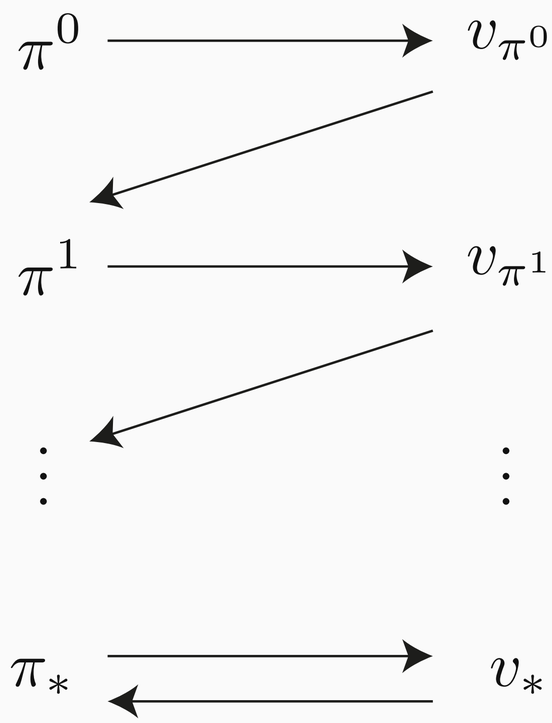
$$\pi^{i+1}(s) = \operatorname*{argmax}_a q_{\pi^i}(s, a),$$

$v_{\pi^0}(s) \leq v_{\pi^1}(s) \leq v_{\pi^2}(s) \leq \ldots$.

When there is no improvement anymore, i.e., $v_{\pi^{i+1}}(s) = \max_a q_{\pi^i}(s, a)$, it means that the Bellman optimality equation has been satisfied and we have converged to the optimal policy π_*.

Note to update the policy in each iteration, we need to evaluate q_{π^i} first. How?

Control policy iteration II



Reinforcement learning

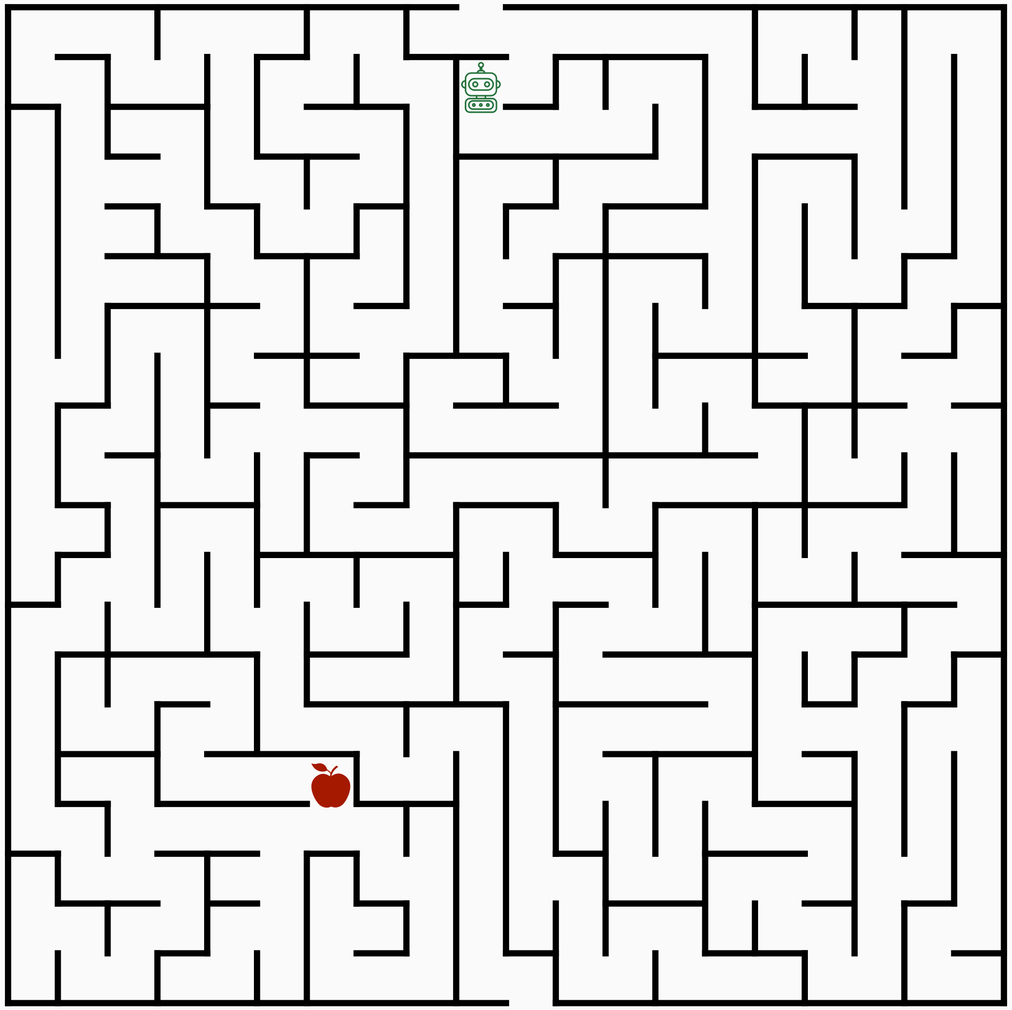
We learned how to find the optimal policy and evaluate an arbitrary policy.

What if the MDP is unknown? Like no knowledge of **transitions** and **rewards**.

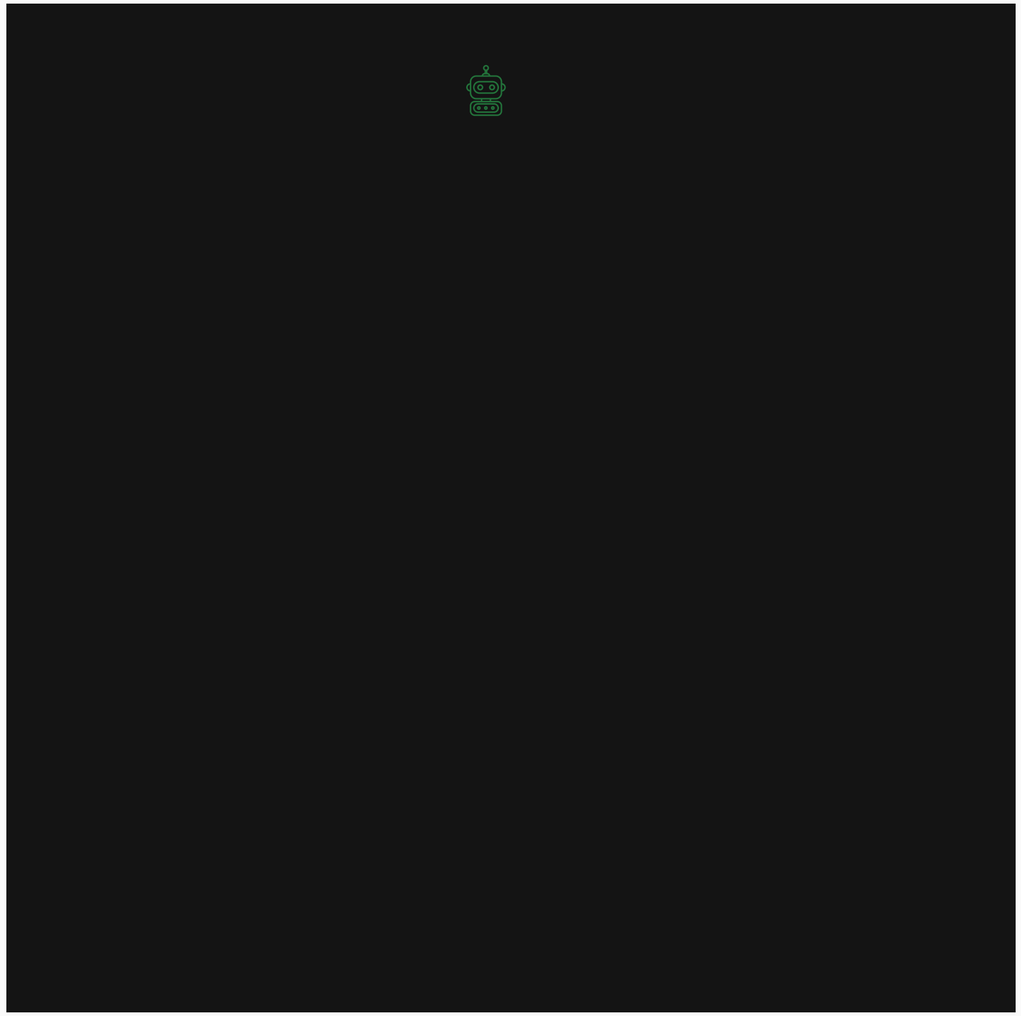
or very “complex”? For instance, too many states to cover.

Reinforcement Learning sampling and learning!

Known Maze



Unknown Maze



Prediction and control in an unknown MDP

Prediction

- Monte-Carlo policy evaluation
- Temporal-difference evaluation

Control

- Monte-Carlo control
- TD control (Sarsa)
- Q-learning (off-policy)

Prediction
unknown MDP

Prediction Monte-Carlo policy evaluation I

Episodic MDP terminates after a finite number of time steps.

Prediction Monte-Carlo policy evaluation I

Episodic MDP terminates after a finite number of time steps.

Given an episodic MDP and a policy π, we are interested in v_π. How to estimate v_π by sampling episodes of the MDP?

$$(S_0, A_0, R_1)(S_1, A_1, R_2)\ldots(S_T, -, -)$$

Prediction Monte-Carlo policy evaluation I

Episodic MDP terminates after a finite number of time steps.

Given an episodic MDP and a policy π, we are interested in v_π. How to estimate v_π by sampling episodes of the MDP?

$$(S_0, A_0, R_1)(S_1, A_1, R_2)\dots(S_T, -, -)$$

Recall the definition of $v_\pi(s)$:

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi\left[\sum_{i=t}^{T-t-1} \gamma^i R_{t+i+1} \middle| S_t = s\right].$$

Prediction Monte-Carlo policy evaluation I

Episodic MDP terminates after a finite number of time steps.

Given an episodic MDP and a policy π, we are interested in v_π. How to estimate v_π by sampling episodes of the MDP?

$$(S_0, A_0, R_1)(S_1, A_1, R_2)\dots(S_T, -, -)$$

Recall the definition of $v_\pi(s)$:

$$v_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] = \mathbb{E}_\pi\left[\sum_{i=t}^{T-t-1} \gamma^i R_{t+i+1} \middle| S_t = s\right].$$

Monte-Carlo estimation!

Prediction Monte-Carlo policy evaluation II

We compute the empirical mean of return G_t by keeping track of the number of times we have visited a state, $N(s)$, and the total amount of return we have received starting from that state, $S(s)$.

Prediction Monte-Carlo policy evaluation II

We compute the empirical mean of return G_t by keeping track of the number of times we have visited a state, $N(s)$, and the total amount of return we have received starting from that state, $S(s)$.

- For each sampled episode by following policy π:
  - Every time-step t that state s is visited in the episode:
    - $N(s) \leftarrow N(s) + 1$
    - $S(s) \leftarrow S(s) + G_t$
- Estimate the value $v_\pi(s)$ by $V(s) = S(s)/N(s)$.

Prediction Monte-Carlo policy evaluation II

We compute the empirical mean of return G_t by keeping track of the number of times we have visited a state, $N(s)$, and the total amount of return we have received starting from that state, $S(s)$.

- For each sampled episode by following policy π:
  - Every time-step t that state s is visited in the episode:
    - $N(s) \leftarrow N(s) + 1$
    - $S(s) \leftarrow S(s) + G_t$
- Estimate the value $v_\pi(s)$ by $V(s) = S(s)/N(s)$.

$V(s)$ converges to $v_\pi(s)$ as the number of episodes goes to ∞. (Law of large numbers)

Prediction temporal-difference learning I

How about non-episodic MDPs?

Prediction temporal-difference learning I

How about non-episodic MDPs?

One can instead use a running version of the previous MC estimator:

$$V(s) \leftarrow V(s) + \frac{1}{N(s)}(G_t - V(s))$$

Prediction temporal-difference learning I

How about non-episodic MDPs?

One can instead use a running version of the previous MC estimator:

$$V(s) \leftarrow V(s) + \frac{1}{N(s)}(G_t - V(s))$$

- G_t is an unbiased estimator of v_π; can we use other estimators?
- $\frac{1}{N(s)}$ can be seen as the learning rate.

Prediction temporal-difference learning I

How about non-episodic MDPs?

One can instead use a running version of the previous MC estimator:

$$V(s) \leftarrow V(s) + \frac{1}{N(s)}(G_t - V(s))$$

- G_t is an unbiased estimator of v_π; can we use other estimators?
- $\frac{1}{N(s)}$ can be seen as the learning rate.

$$V(s) \leftarrow V(s) + \alpha(\tilde{V}_t - V(s))$$

Prediction temporal-difference learning II

TD uses the following estimator instead of G_t:

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)).$$

Prediction temporal-difference learning II

TD uses the following estimator instead of G_t:

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)).$$

- TD can be applied to non-episodic MDPs. Why?
- TD has a lower variance compared to MC. Why?
- MC might work better in real scenarios. Why?

Prediction MC vs TD

- TD uses a one-step lookahead estimator.
- MC uses an ∞-step lookahead (full episode).

Prediction MC vs TD

- TD uses a one-step lookahead estimator.
- MC uses an ∞-step lookahead (full episode).

Anything in between?

Prediction MC vs TD

- TD uses a one-step lookahead estimator.
- MC uses an ∞-step lookahead (full episode).

Anything in between?

TD and MC can be seen as two extremes of n-step TD:

$$V(S_t) \leftarrow V(S_t) + \alpha \left(\underbrace{R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^n V(S_{t+n})}_{n-\text{step TD target}} - V(S_t) \right)$$

Similarly for $Q(S_t, A_t)$

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^n Q(S_{t+n}, A_{t+n}) - Q(S_t, A_t))$$

Control
unknown MDP

Control policy iteration I

How about finding the optimal policy and value function in an unknown MDP?

Control policy iteration I

How about finding the optimal policy and value function in an unknown MDP?

Policy iteration, again! Evaluate policy π using MC or TD targets to estimate $Q(s, a) \approx q_\pi(s, a)$, then improve the policy according to $\pi'(s) = \textsf{argmax}_a \, Q(s, a)$.

Control policy iteration I

How about finding the optimal policy and value function in an unknown MDP?

Policy iteration, again! Evaluate policy π using MC or TD targets to estimate $Q(s, a) \approx q_\pi(s, a)$, then improve the policy according to $\pi'(s) = \operatorname{argmax}_a Q(s, a)$.

We have to be careful to ensure continual "exploration".

ϵ-greedy policy

Choose the best action with probability $1 - \epsilon$ (instead of 1).

$$\pi'(a|s) = \begin{cases} 1 - \epsilon & \text{if } a^* = \operatorname{argmax}_a q_\pi(s, a) \\ \epsilon/(m-1) & \text{o.w.} \end{cases}$$

where $m = |\mathcal{A}|$ is the number of possible actions.

ϵ-greedy policy

Choose the best action with probability $1 - \epsilon$ (instead of 1).

$$\pi'(a|s) = \begin{cases} 1 - \epsilon & \text{if } a^* = \text{argmax}_a \, q_\pi(s, a) \\ \epsilon/(m-1) & \text{o.w.} \end{cases}$$

where $m = |\mathcal{A}|$ is the number of possible actions.

Interestingly, we can show that if π is an ϵ-greedy policy, $v_{\pi'}(s) \geq v_\pi(s)$.

Control MC policy iteration

- Sample kth episode by following policy π^k:
  - For each state S_t and action A_t in the episode:
    - $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
    - $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)}(G_t - Q(S_t, A_t))$
- Compute π^{k+1} as the ϵ-greedy policy based on Q.

Control MC policy iteration

- Sample kth episode by following policy π^k:
  - For each state S_t and action A_t in the episode:
    - $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
    - $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)}(G_t - Q(S_t, A_t))$
- Compute π^{k+1} as the ϵ-greedy policy based on Q.

$Q(s, a)$ converges to $q_*(s, a)$ as the number of episodes goes to ∞.

Control TD policy iteration (SARSA) I

Naturally, we can use the TD target instead of G_t:

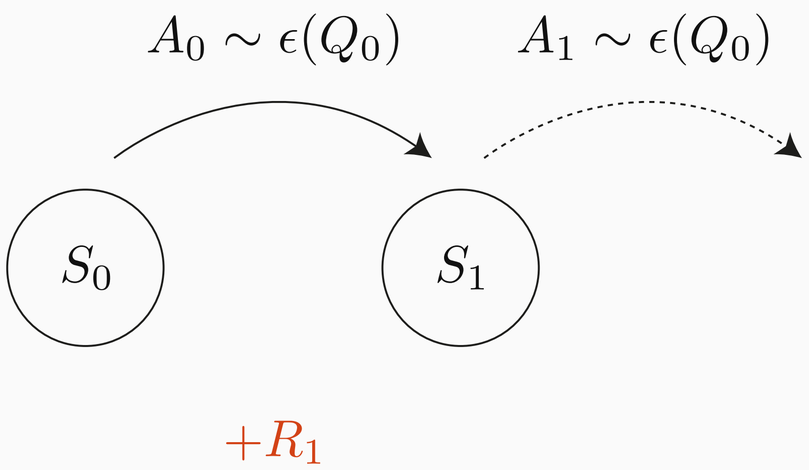
$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$$

Moreover, we can update more often, at **each step** instead of **each episode**.

Control TD policy iteration (SARSA) II

- Initialize Q arbitrarily.
- For each epoch:
  - Begin from a random state S.
  - Choose a random action A according to the ϵ-greedy policy derived from Q.
  - Repeat till termination:
    - Take action A, observe R and S'.
    - Choose A' according to the ϵ-greedy policy derived from Q.
    - $Q(S, A) \leftarrow Q(S, A) + \alpha(R + \gamma Q(S', A') - Q(S, A))$
    - $S \leftarrow S'$ and $A \leftarrow A'$

Control TD policy iteration (SARSA) III



Control Q-learning

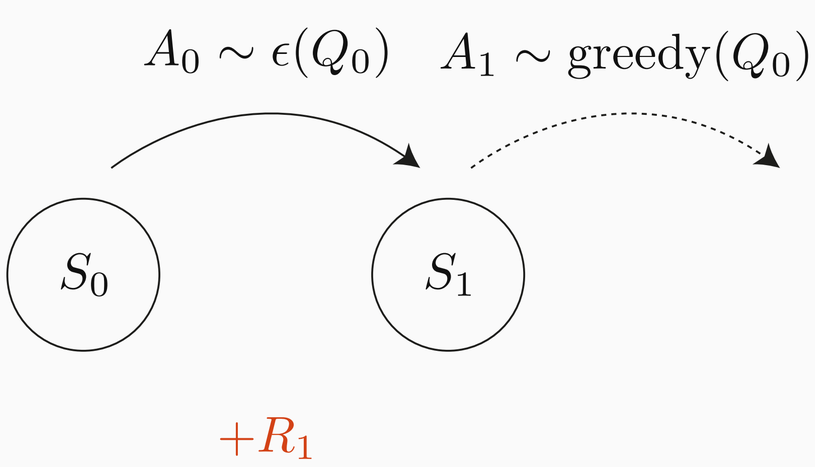
- Initialize Q arbitrarily.
- For each epoch:
  - Begin from a random state S.
  - Repeat till termination:
    - Choose a random action A according to the ϵ-greedy policy derived from Q.
    - Take action A, observe R and S'.
    - $Q(S, A) \leftarrow Q(S, A) + \alpha(R + \gamma \max_a Q(S', a) - Q(S, A))$
    - $S \leftarrow S'$

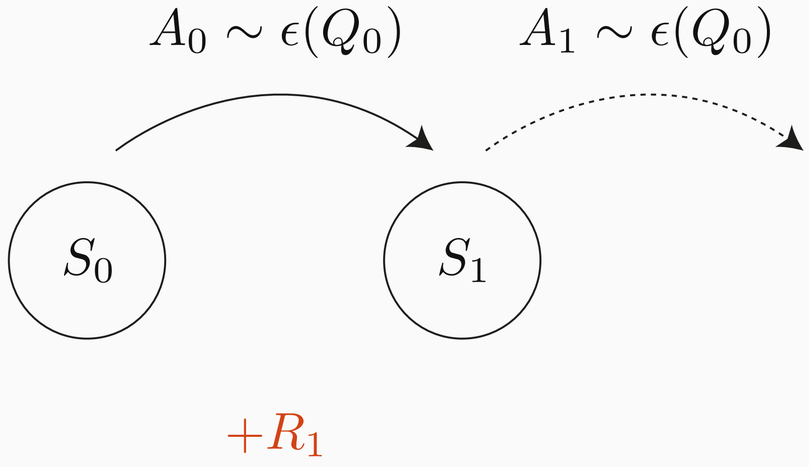
π is the greedy policy, whereas μ is the ϵ-greedy policy.

The update rule can be interpreted as an estimation of the Bellman optimality equation:

$$q_*(s, a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \middle| S_t = s, A_t = a\right]$$

Q-learning vs SARSA





RL and function approximation I

Let's take a look again at the update rules we have seen so far:

Monte-Carlo control $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{1}{N(S_t, A_t)}(G_t - Q(S_t, A_t))$

SARSA $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))$

Q-Learning $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$

RL and function approximation II

They can be thought of as GD updates to the following optimization problems:

Monte-Carlo control $\min_Q (G_t - Q(S_t, A_t))^2$

SARSA $\min_Q (R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))^2$

Q-Learning $\min_Q (R_{t+1} + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t))^2$

RL and function approximation II

They can be thought of as GD updates to the following optimization problems:

Monte-Carlo control $\min_Q (G_t - Q(S_t, A_t))^2$

SARSA $\min_Q (R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))^2$

Q-Learning $\min_Q (R_{t+1} + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t))^2$

One way or another, they can be seen as estimations of the following minimization:

$$\min_Q \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q(S_t, A_t))^2$$

RL and function approximation II

They can be thought of as GD updates to the following optimization problems:

Monte-Carlo control $\min_Q (G_t - Q(S_t, A_t))^2$

SARSA $\min_Q (R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))^2$

Q-Learning $\min_Q (R_{t+1} + \gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t))^2$

One way or another, they can be seen as estimations of the following minimization:

$$\min_Q \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q(S_t, A_t))^2$$

Can we parametrize Q?

RL and function approximation III

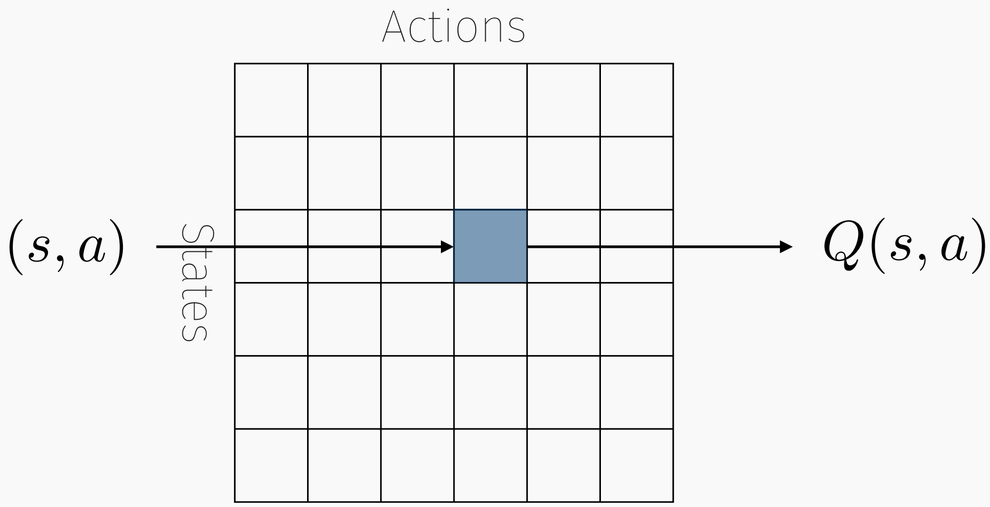
Indeed!

$$\min_{\theta} \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q_\theta(S_t, A_t))^2$$

RL and function approximation III

Indeed!

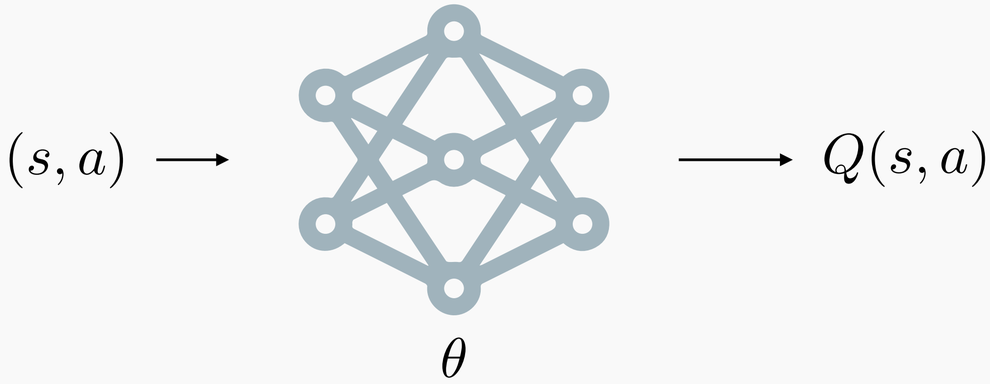
$$\min_{\theta} \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q_\theta(S_t, A_t))^2$$



RL and function approximation III

Indeed!

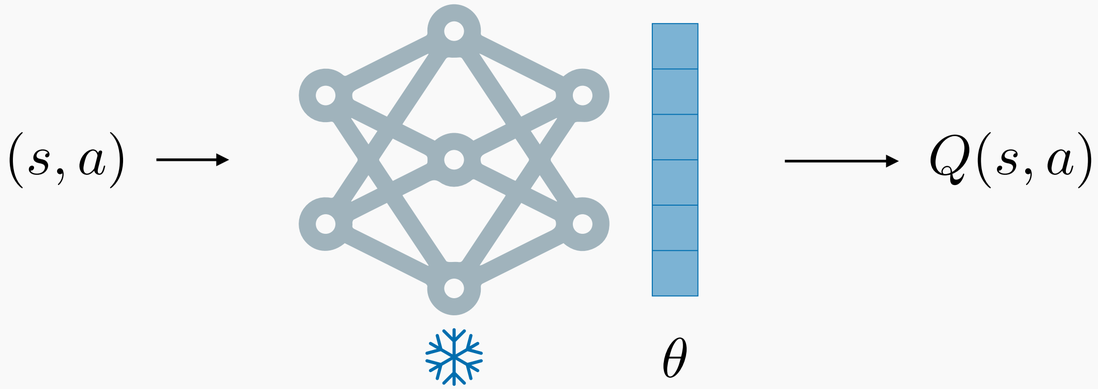
$$\min_{\theta} \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q_\theta(S_t, A_t))^2$$



RL and function approximation III

Indeed!

$$\min_{\theta} \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q_\theta(S_t, A_t))^2$$



RL and function approximation III

Indeed!

$$\min_{\theta} \sum_{S_t, A_t} (\hat{q}_\pi(S_t, A_t) - Q_\theta(S_t, A_t))^2$$

Examples

- $Q_\theta(s, a) = \theta_{s,a}$, for all s, a (Lookup table).
- $Q_\theta(s, a) = \theta^\top \phi(s, a)$, for a feature representation ϕ (Linear model).
- $Q_\theta(s, a) = f_\theta(s, a)$, for a deep neural network f_θ (Deep networks).

RL and function approximation IV

We can use SGD to find optimal parameters θ:

$$\theta \leftarrow \theta + \alpha(\hat{q}_\pi(S, A) - Q_\theta(S, A))\nabla_\theta Q_\theta(S, A)$$

RL and function approximation IV

We can use SGD to find optimal parameters θ:

$$\theta \leftarrow \theta + \alpha(\hat{q}_\pi(S, A) - Q_\theta(S, A))\nabla_\theta Q_\theta(S, A)$$

Based on the estimator we choose for $\hat{q}_\pi$ and the parametric family Q_θ, we can come up with different algorithms.

RL and function approximation IV

We can use SGD to find optimal parameters θ:

$$\theta \leftarrow \theta + \alpha(\hat{q}_\pi(S, A) - Q_\theta(S, A))\nabla_\theta Q_\theta(S, A)$$

Based on the estimator we choose for $\hat{q}_\pi$ and the parametric family Q_θ, we can come up with different algorithms. **For instance**

$$\theta \leftarrow \theta + \alpha \sum_{S_t, A_t} (R_{t+1} + \gamma Q_\theta(S_{t+1}, A_{t+1}) - Q_\theta(S_t, A_t))\nabla_\theta f_\theta(S_t, A_t)$$

SARSA with function approximation!

Trial and error,
Actions **reinforced** with each turn,
Success is earned hard.

— *ChatGPT*