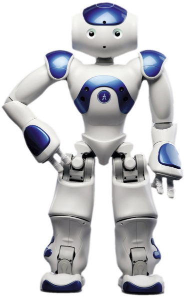
Reinforcement Learning: Methods and Algorithms

Thomas Bonald

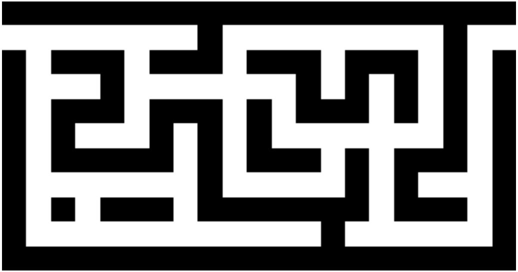
2019 – 2020

Reinforcement Learning

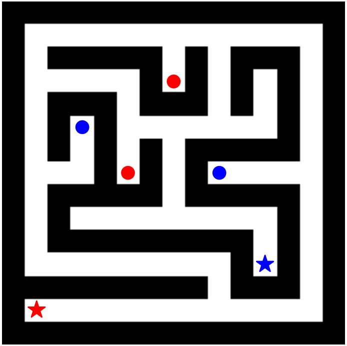
- Learning by **trial and error**
- Inspired by the behavior of animals (including humans!)
- The **exploration-exploitation** trade-off
- Many applications: robotics, games, advertising, content recommendation, medicine, etc.



Toy example: Maze



Toy example: Pac-Man



Outline

1. **Markov decision process**
2. Dynamic programming
3. Online estimation
4. Online control

Markov decision process

At time $t = 0, 1, 2, \ldots$, the agent in **state** s_t takes **action** a_t and:

- receives **reward** r_t
- moves to **state** s_{t+1}

The reward and new state are **stochastic** in general.
Some states may be **terminal**.

Definition

A **Markov decision process** (MDP) is defined by:

- some initial state s_0
- the reward distribution, $r_t \sim p(r|s_t, a_t)$
- the transition probabilities, $s_{t+1} \sim p(s|\ s_t, a_t)$

Policy

Definition

Given a Markov decision process, a **policy** defines the action taken in each state:

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

- A policy is **stochastic** in general.
- When **deterministic**, we use the simple notation $\pi(s)$ for the action taken in state s.

Remark

Given some policy π, the sequence of states $s_0, s_1, s_2, \ldots$ defines a **Markov process**.

Objective function

Definition

Given the rewards $r_0, r_1, r_2, \ldots$, we refer to the **gain** as:

$$\begin{aligned} G &= r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots \\ &= \sum_{t=0}^{+\infty} \gamma^t r_t \end{aligned}$$

The parameter $\gamma \in [0, 1]$ is the **discount factor**:

- $\gamma = 0 \longrightarrow$ immediate reward
- $\gamma = 1 \longrightarrow$ cumulative reward

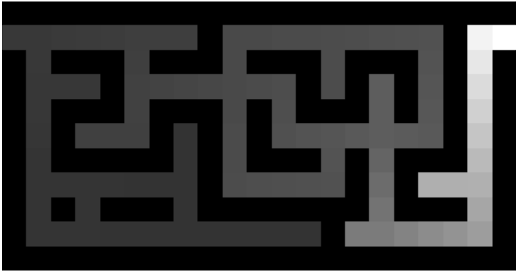
Value function

Definition

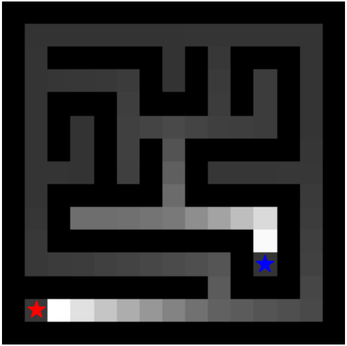
The **value** function is the expected gain from each state:

$$\forall s, \quad V_\pi(s) = \mathrm{E}_\pi(G | s_0 = s)$$

Toy example: Maze (random policy)



Toy example: Pac-Man (random policy)



Bellman's equation

Recall the definition:

$$\forall s, \quad V_\pi(s) = \mathrm{E}_\pi(G|\, s_0 = s)$$

Proposition

The value function V_π of any policy π is the unique solution to the **fixed-point equation**:

$$\forall s, \quad V(s) = \mathrm{E}_\pi(r_0 + \gamma V(s_1)|\, s_0 = s)$$

Solution to Bellman's equation

Let T_π the **operator** associated with Bellman's equation:

$$\forall s, \quad T_\pi(V)(s) = \mathrm{E}_\pi(r_0 + \gamma V(s_1)|\ s_0 = s)$$

This operator is **contracting**:

$$\forall U, V, \quad ||T_\pi(V) - T_\pi(U)||_\infty \leq \gamma ||V - U||_\infty$$

Proposition

If $\gamma < 1$, then:

$$\lim_{n \to +\infty} T_\pi^n(V) = V_\pi$$

Optimal policy

Definition

A policy $\pi^\star$ is **optimal** if and only if

$$\forall s, \quad V_{\pi^\star}(s) \geq V_\pi(s)$$

Bellman's optimality equation

Proposition

There is a unique solution $V_\star$ to the **fixed-point equation**:

$$\forall s, \quad V(s) = \max_a \mathrm{E}(r_0 + \gamma V(s_1) | \, s_0 = s, a_0 = a)$$

Optimal value evaluation

Let $T_\star$ the operator associated with Bellman's optimality equation:

$$\forall s, \quad T_\star(V)(s) = \max_a \mathrm{E}(r_0 + \gamma V(s_1) | \, s_0 = s, a_0 = a)$$

Proposition

If $\gamma < 1$, then:

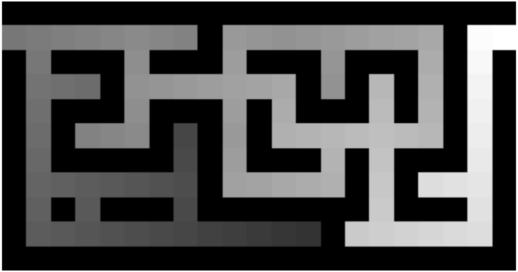
$$\lim_{n \to +\infty} T_\star^n(V) = V_\star \geq \max_\pi V_\pi$$

The proof relies on the fact that:

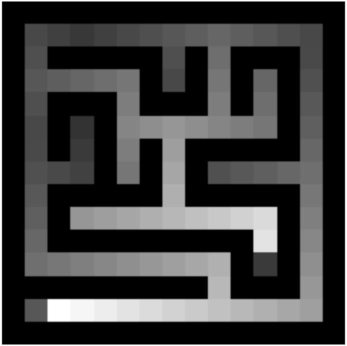
- $T_\star$ is **contracting**
- $T_\star \geq T_\pi$ for any policy π, so that:

$$V_\star = \lim_{n \to +\infty} T_\star^n(V) \geq \lim_{n \to +\infty} T_\pi^n(V) = V_\pi$$

Toy example: Maze



Toy example: Pac-Man



Optimal policy

Define:

$$\forall s, \quad \pi^\star(s) = a^\star \in \arg\max_a \mathrm{E}(r_0 + \gamma V_\star(s_1) |\ s_0 = s, a_0 = a)$$

Bellman's optimality theorem

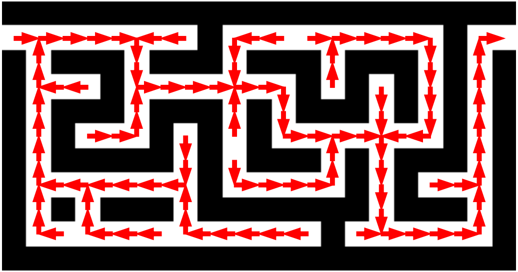
The policy $\pi^\star$ is optimal:

$$\forall s, \quad V_{\pi^\star}(s) = \max_\pi V_\pi(s)$$

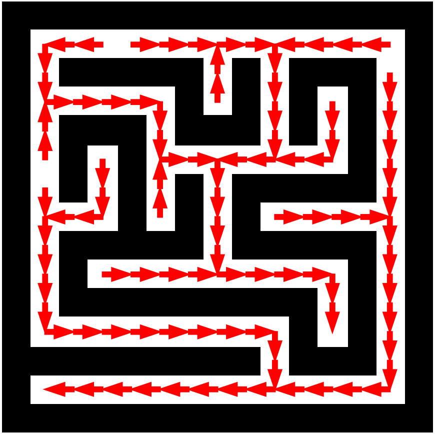
Note that:

- The policy $\pi^\star$ is **deterministic**
- The optimal policy is **not unique**.

Toy example: Maze (optimal policy)



Toy example: Pac-Man (optimal policy)



Outline

1. Markov decision process
2. **Dynamic programming** $\rightarrow$ Policy iteration, Value iteration
3. Online estimation
4. Online control

Policy improvement

Given some policy π, let π' the policy defined by:

$$\pi'(s) = a^\star \in \arg\max_a \mathrm{E}(r_0 + \gamma V_\pi(s_1) |\ s_0 = s, a_0 = a)$$

Proposition

The policy π' is better than π:

$$\forall s, \quad V_{\pi'}(s) \geq V_\pi(s)$$

Policy iteration

Algorithm

Starting from some arbitrary **policy** $\pi = \pi_0$, iterate until convergence:

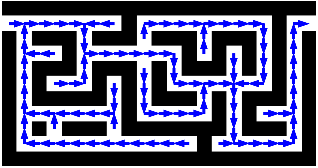
1. **Evaluate** the policy (by solving Bellman's equation)
2. **Improve** the policy:

$$\forall s, \quad \pi(s) \leftarrow \arg\max_a \mathrm{E}(r_0 + \gamma V_\pi(s_1) |\ s, a)$$

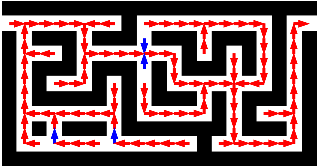
- The sequence $\pi_0, \pi_1, \pi_2, \ldots$ is **monotonic** and converges in **finite time** (for finite number of states and actions).
- The limit is an **optimal policy**.

Toy example: Maze (policy iteration)

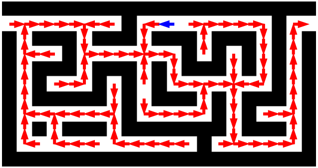
$n = 1$



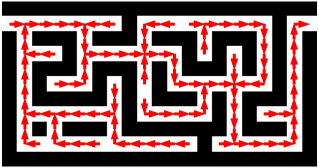
$n = 2$



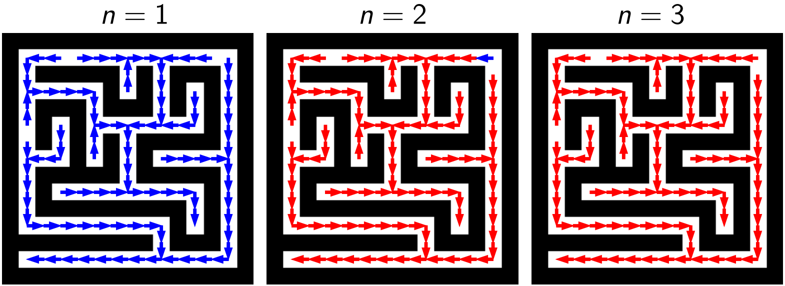
$n = 5$



$n = 6$



Toy example: Pac-Man (policy iteration)



Practical considerations

- The step of **policy evaluation** is time-consuming (solution of Bellman's equation)
- Do we need the **exact** solution?
  No, since it is used only to **improve** the policy!
- The number of iterations can be limited to some value k
  How to set k? Why not... $k = 1$?

Value Iteration

Algorithm

Starting from some arbitrary **value** function $V = V_0$,
iterate until convergence:

$$\forall s, \quad V(s) \leftarrow \max_a \mathrm{E}(r_0 + \gamma V(s_1) | \, s, a)$$

- The sequence $V_0, V_1, V_2, \ldots$ converges in **finite time** (for finite number of states and actions).
- The **limit** solves Bellman's optimality equation.
- The corresponding policy is **optimal**.

Outline

1. Markov decision process
2. Dynamic programming
3. **Online estimation** $\rightarrow$ Monte-Carlo, TD learning
4. Online control

Temporal difference

Assume you want to estimate the **empirical mean** of some data stream $x_1, x_2, \ldots$

Two strategies:

- Store the sum:
  For each new sample x_t,

$$S \leftarrow S + x_t \quad X \leftarrow \frac{S}{t}$$

- Update with the **temporal difference**:
  For each new sample x_t,

$$X \leftarrow X + \alpha(x_t - X) \quad \alpha = \frac{1}{t}$$

Note: Most often, the learning rate α is fixed!

MC learning

Idea: Evaluate the **value** function of some policy π using **episodes** $s_0, s_1, \ldots, s_T$ (assuming the presence **terminal** states)

Let:

$$
\begin{aligned}
&G_0 = r_0 + \gamma r_1 + \ldots + \gamma^{T-1} r_{T-1} \\
&G_1 = r_1 + \gamma r_2 + \ldots + \gamma^{T-2} r_{T-1} \\
&\ldots \\
&G_{T-1} = r_{T-1}
\end{aligned}
$$

MC updates

$$\forall t, \quad V(s_t) \stackrel{\alpha}{\leftarrow} G_t$$

TD learning

Idea: Online estimation of the **value** function of some policy π (no need for **terminal** states)

TD updates

$$\forall t, \quad V(s_t) \xleftarrow{\alpha} r_t + \gamma V(s_{t+1})$$

cf. Bellman's equation

$$\forall s, \quad V(s) = \mathrm{E}_\pi(r_t + \gamma V(s_{t+1}) |\ s_t = s)$$

From estimation to control

Let V be the estimate of the value function of some policy π

Policy improvement

$$\pi(s) \leftarrow \arg\max_a \mathrm{E}(r_0 + \gamma V(s_1) | \; s_0 = s, a_0 = a)$$

Outline

1. Markov decision process
2. Dynamic programming
3. Online estimation
4. **Online control** $\rightarrow$ SARSA, Q-learning

Action-value function

Recall the **value** function:

$$\forall s, \quad V_\pi(s) = \mathrm{E}_\pi(G | s_0 = s)$$

Without model, how to predict the impact of action a in state s?

Definition

The **action-value** function is the expected gain from each state-action pair:

$$Q_\pi(s, a) = \mathrm{E}_\pi(G | s_0 = s, a_0 = a)$$

Note that:

$$Q_\pi(s, a) = \mathrm{E}(r_0 + \gamma V_\pi(s_1) | s_0 = s, a_0 = a)$$

Bellman's equation for the action-value function

Proposition

The action-value function Q_π of any policy π is the unique solution to the **fixed-point equation**:

$$\forall s, a, \quad Q(s,a) = \mathrm{E}_\pi(r_0 + \gamma Q(s_1, a_1) |\ s_0 = s, a_0 = a)$$

Optimal policy

By definition, the optimal policy is:

$$\pi^\star(s) = a^\star \in \arg\max Q_\star(s, a)$$

with

$$Q_\star(s, a) = \mathrm{E}(r_0 + \gamma V_\star(s_1) | s_0 = s, a_0 = a)$$

Bellman's optimality theorem

$$\forall s, a, \quad Q_{\pi^\star}(s, a) = \max_\pi Q_\pi(s, a)$$

Bellman's optimality equations

Recall that $V_\star$ is the unique solution to:

$$\forall s, \quad V(s) = \max_a \mathrm{E}(r_0 + \gamma V(s_1) |\ s_0 = s, a_0 = a)$$

Proposition

There is a unique solution $Q_\star$ to the **fixed-point equation**:

$$\forall s, a, \quad Q(s,a) = \mathrm{E}(r_0 + \gamma \max_{a'} Q(s_1, a') |\ s_0 = s, a_0 = a)$$

Exploration vs exploitation

- Pure exploitation $\longrightarrow$ **greedy**

$$\pi(s) \leftarrow \arg\max_a Q(s,a)$$

- Pure exploration $\longrightarrow$ **random**

$$\pi(s) \leftarrow \text{random}$$

- Exploration-exploitation trade-off $\longrightarrow$ e.g., ε-**greedy**

$$\pi(s) \leftarrow \begin{cases} a^\star \in \arg\max_a Q(s,a) & \text{with probability } 1-\varepsilon \\ \text{random} & \text{with probability } \varepsilon \end{cases}$$

ε-greedy policy improvement

Let π be an ε-greedy policy and:

$$\pi'(s) \leftarrow \begin{cases} a^\star \in \arg\max_a Q_\pi(s,a) & \text{with probability } 1-\varepsilon' \\ \text{random} & \text{with probability } \varepsilon' \end{cases}$$

Proposition

If $\varepsilon' \leq \varepsilon$, the policy π' is better than π:

$$\forall s, \quad V_{\pi'}(s) \geq V_\pi(s)$$

Estimation of the action-value function

MC updates

Given some sequence $s_0, a_0, s_1, a_1, \ldots, s_T$,

$$\forall t, \quad Q(s_t, a_t) \stackrel{\alpha}{\leftarrow} G_t$$

TD updates

$$\forall t, \quad Q(s_t, a_t) \stackrel{\alpha}{\leftarrow} r_t + \gamma Q(s_{t+1}, a_{t+1})$$

cf. Bellman's equation

$$\forall s, a, \quad Q(s, a) = \mathrm{E}(r_t + \gamma Q(s_{t+1}, a_{t+1}) |\ s_t = s, a_t = a)$$

SARSA

Key idea: Learn the **optimal** action-value function $Q_\star$ online, using the ε-greedy policy:

$$\pi(s) \leftarrow \begin{cases} a^\star \in \arg\max_a Q(s,a) & \text{with probability } 1-\varepsilon \\ \text{random} & \text{with probability } \varepsilon \end{cases}$$

TD updates

$$\forall t, \quad Q(s_t, a_t) \overset{\alpha}{\leftarrow} r_t + \gamma Q(s_{t+1}, a_{t+1})$$

Q-learning

Key idea: Learn the **optimal** action-value function $Q_\star$ online, using the ε-greedy policy for **control** and the greedy policy for **estimation**.

TD updates

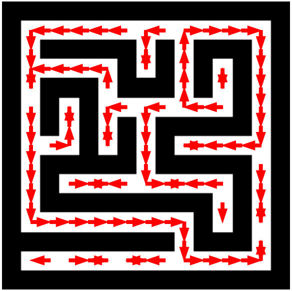
$$\forall t, \quad Q(s_t, a_t) \stackrel{\alpha}{\leftarrow} r_t + \gamma \max_a Q(s_{t+1}, a)$$

cf. Bellman's **optimality** equation

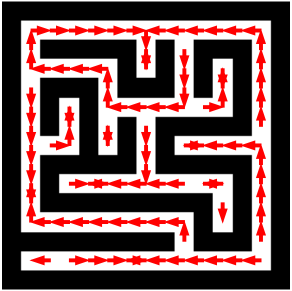
$$\forall s, a, \quad Q(s, a) = \mathrm{E}(r_t + \gamma \max_{a'} Q(s_{t+1}, a') |\ s_t = s, a_t = a)$$

Toy example: Pac-Man ($n = 1000$)

SARSA

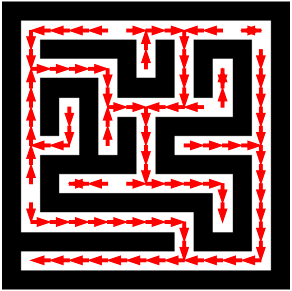


Q-learning

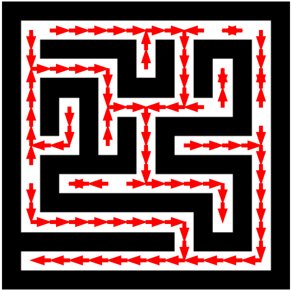


Toy example: Pac-Man ($n = 100,000$)

SARSA



Q-learning



Outline

1. Markov decision process
2. Dynamic programming $\rightarrow$ Policy Iteration, Value Iteration
3. Online estimation $\rightarrow$ Monte-Carlo, TD learning
4. Online control $\rightarrow$ SARSA, Q-learning

References

Olivier Sigaud
Course on Reinforcement Learning (slides, videos)

David Silver
Course on Reinforcement Learning (slides)

Richard Sutton and Andrew Barto
Book Reinforcement Learning: An Introduction (2015)