Reinforcement Learning

PhD course @ “Dottorato di ingegneria industriale e dell’informazione”
Trieste, 2024

Simone Silvetti
Research & Development

Who am I?

Simone Silvetti (silvetti@esteco.com)

- ➔ Studied mathematics in Rome
- ➔ Phd in Computer Science @ Udine
- ➔ Currently working in ESTECO

application of quantitative formal methods and machine learning techniques to Verification and Model-based Testing of Complex Systems

Who am I?

Simone Silvetti (silvetti@esteco.com)

- ➔ Studied mathematics in Rome
- ➔ Phd in Computer Science @ Udine
- ➔ Currently working in ESTECO

application of quantitative formal methods and machine learning techniques to Verification and Model-based Testing of Complex Systems

Numerical Methods Group

multi-objective optimization algorithms, machine learning, object-oriented programming

Who am I?

Simone Silvetti (silvetti@esteco.com)

- ➔ Studied mathematics in Rome
- ➔ Phd in Computer Science @ Udine
- ➔ Currently working in ESTECO

application of quantitative formal methods and machine learning techniques to Verification and Model-based Testing of Complex Systems

Numerical Methods Group

multi-objective optimization algorithms, machine learning, object-oriented programming

Research and Development

process mining, research projects related to technology and domains useful for ESTECO products

Who am I?

Simone Silvetti (silvetti@esteco.com)

- ➔ Studied mathematics in Rome
- ➔ Phd in Computer Science @ Udine
- ➔ Currently working in ESTECO

application of quantitative formal methods and machine learning techniques to Verification and Model-based Testing of Complex Systems

Numerical Methods Group

multi-objective optimization algorithms, machine learning, object-oriented programming

I worked on “Inverse Reinforcement Learning” applied to autonomous driving

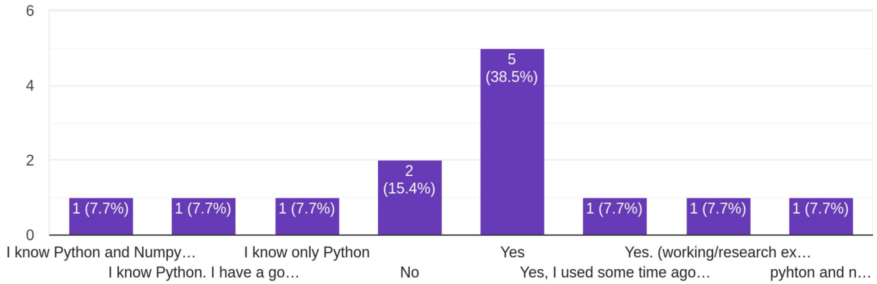
Research and Development

process mining, research projects related to technology and domains useful for ESTECO products

Who are you?

Do you know Python? Numpy, Scipy?

13 responses



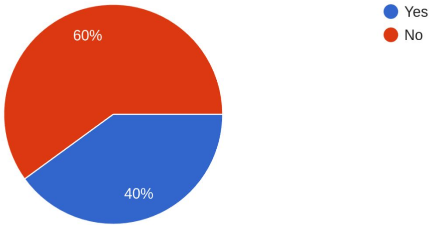
During your studies have you participated in courses of Reinforcement Learning? If yes, which topics have you covered?

13 responses

- No
- Only partially
- I did not partecipate to any course.
- I have never participated in a course about Reinforcement Learning.
- I have never participated at any course of bayesian optimization
- no
- Foundations

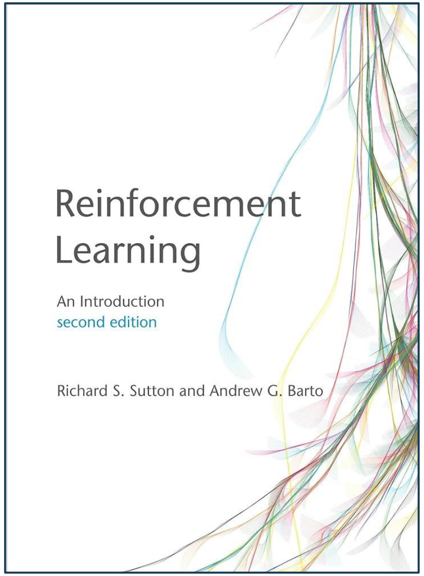
Will you follow the "Learning-based Controllers and the Reality Gap" course?

5 responses



Reference

A book from Sutton et al.

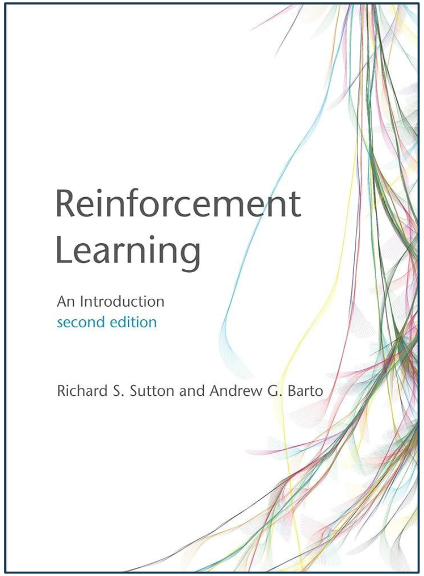


Free available here!

http://incompleteideas.net/book/the-book-2nd.html

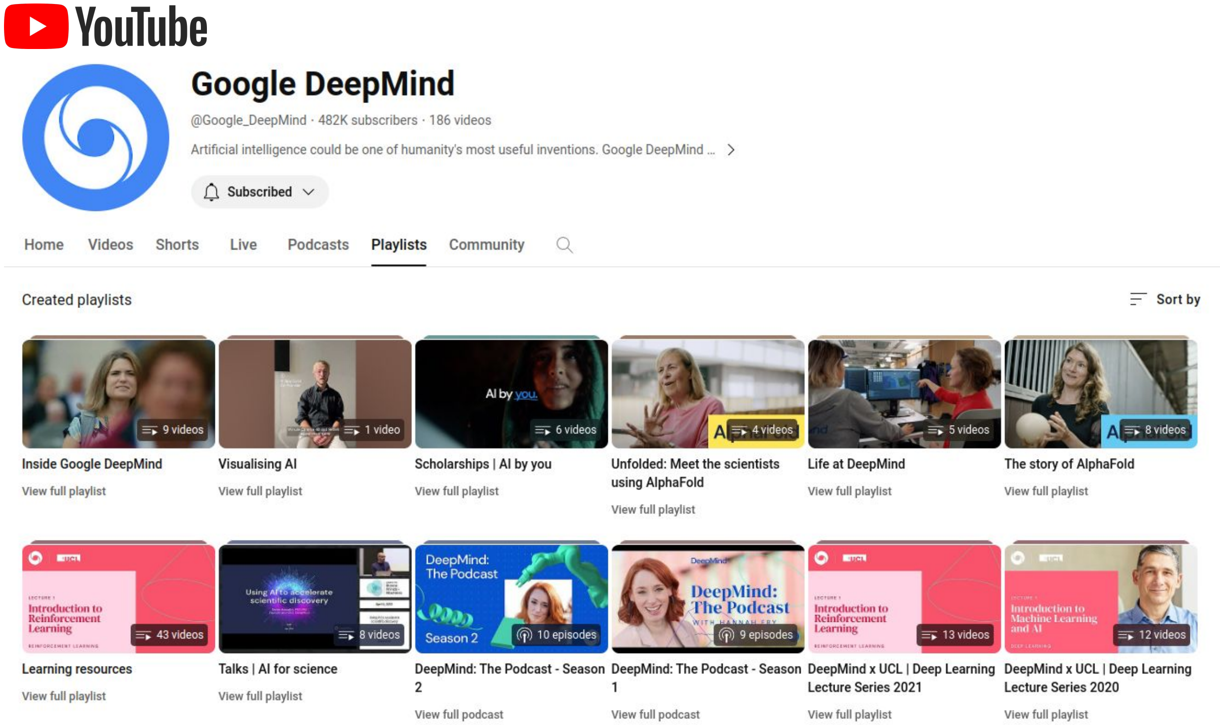
Reference

A book from Sutton et al.



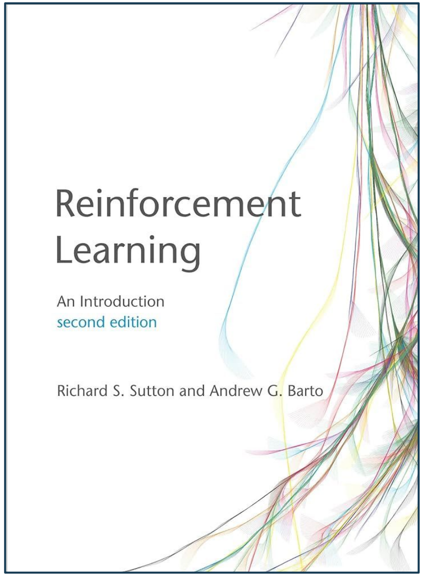
Free available here!

http://incompleteideas.net/book/the-book-2nd.html



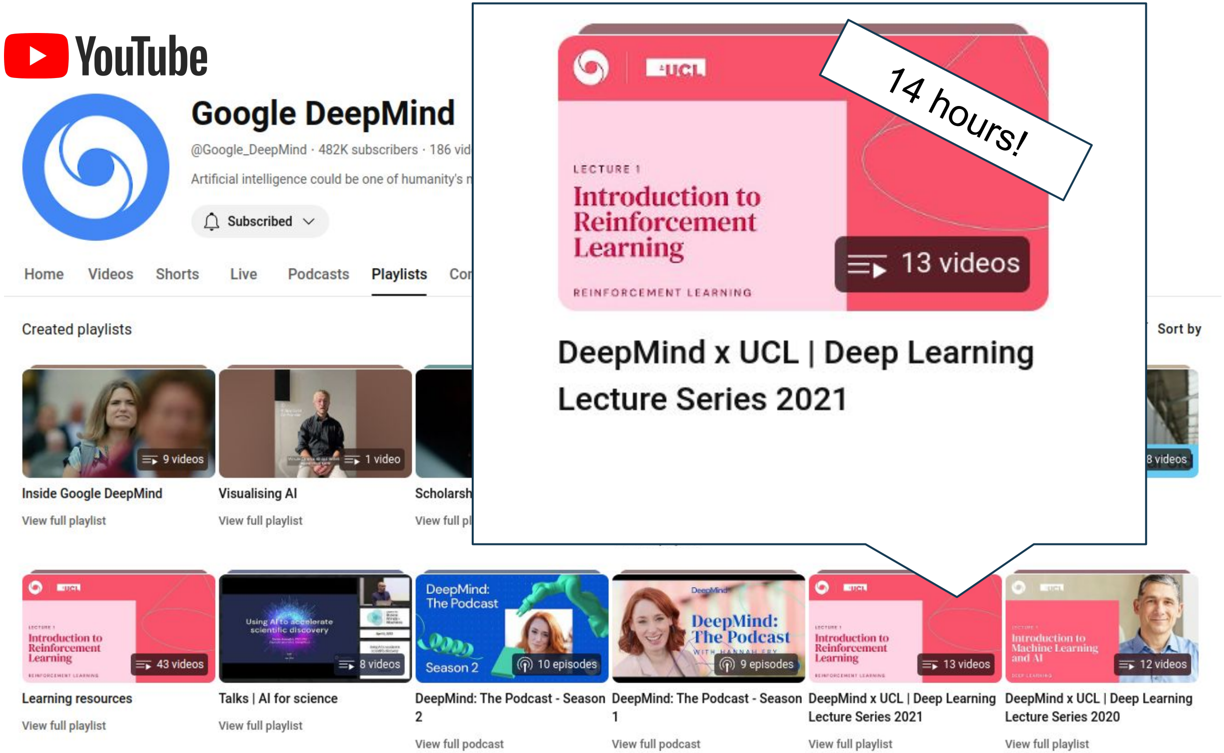
Reference

A book from Sutton et al.



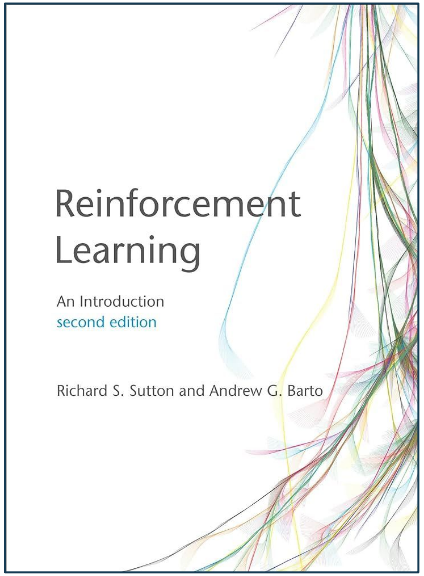
Free available here!

http://incompleteideas.net/book/the-book-2nd.html



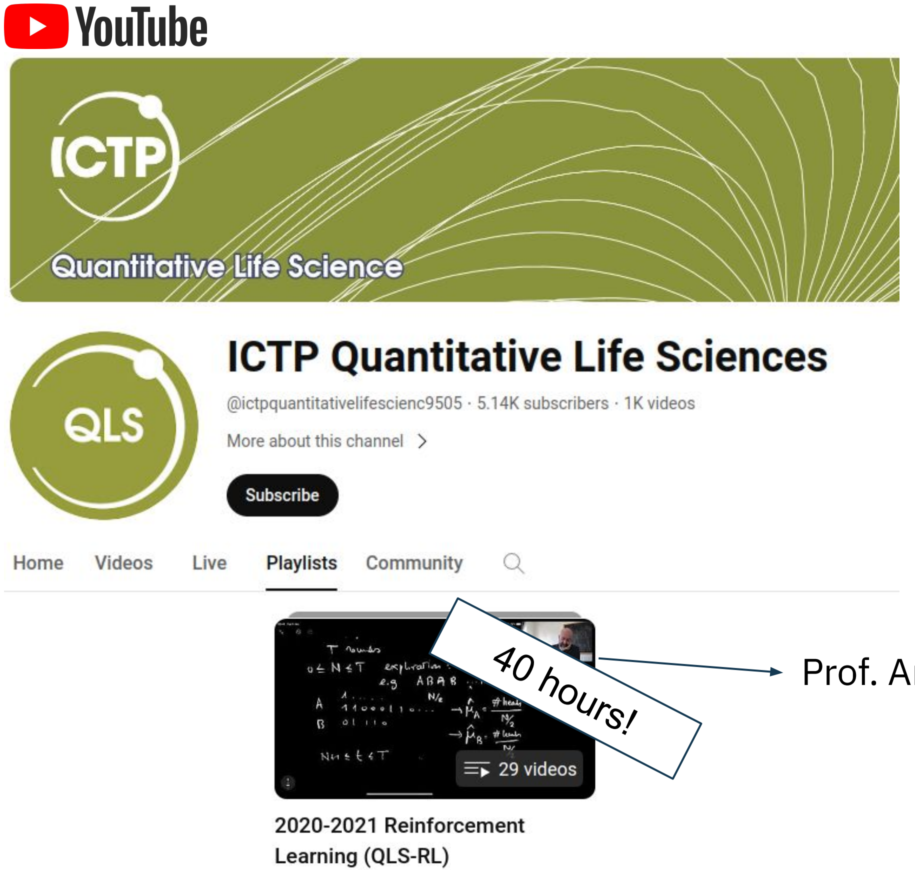
Reference

A book from Sutton et al.



Free available here!

http://incompleteideas.net/book/the-book-2nd.html

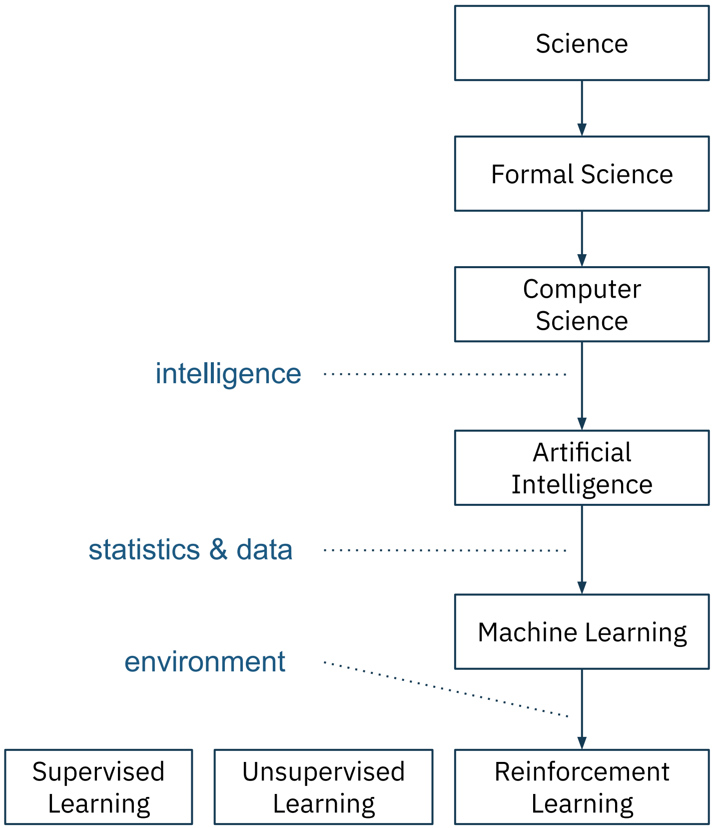


Prof. Antonio Celani

Introduction

What is Reinforcement Learning?

A map



the systematic study of physical and natural world through observation, experimentation, and the testing of theories against the evidence obtained

uses formal systems to generate knowledge

is the study of computation, information and automation

enabling machines to perceive their environment and uses learning and intelligence to take actions that maximize their chances of achieving defined goals

development and study of **statistical algorithms** that can learn from data and **generalize** to unseen data, and thus perform tasks without explicit instructions.

technique that trains software to make decisions to achieve the most optimal results

A definition

Reinforcement Learning

technique that trains software to make decisions to achieve the most optimal results

A definition

Reinforcement Learning

technique that trains ~~software~~ to ~~make decisions~~ to ~~achieve the most optimal results~~

A definition

Reinforcement Learning

technique that trains **agents** to ~~make decisions~~ to ~~achieve the most optimal results~~

A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to ~~achieve the most optimal results~~

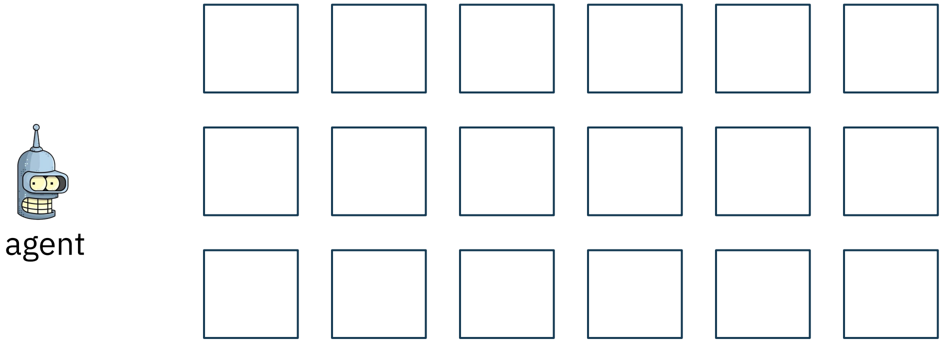
A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**

A definition

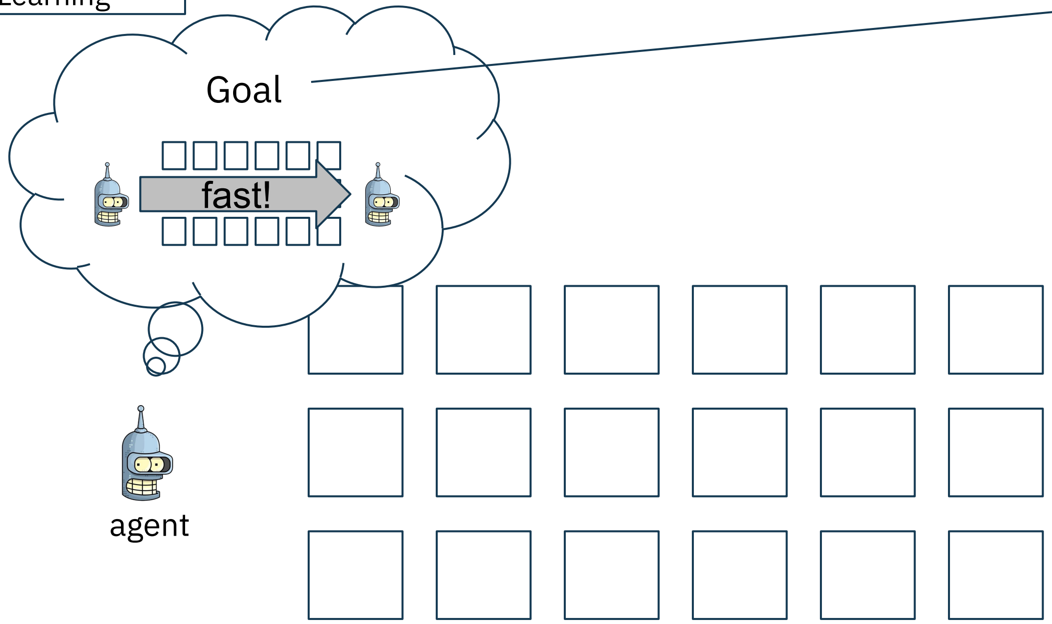
Reinforcement Learning	technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

Reinforcement Learning

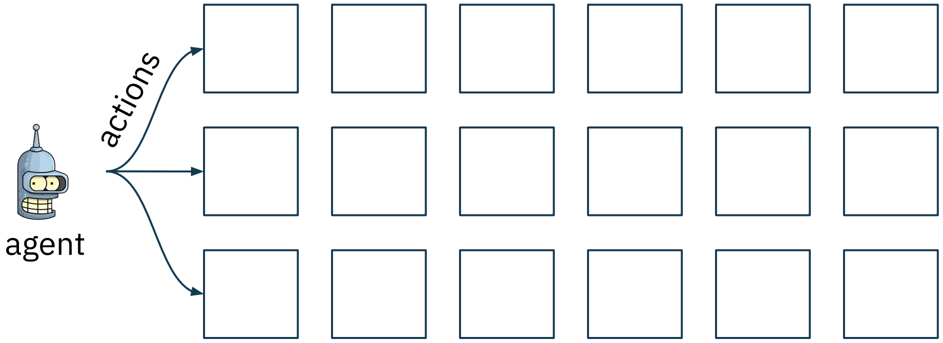
technique that trains **agents** to **map states into actions** to **maximize a <u>cumulative reward</u>**



A definition

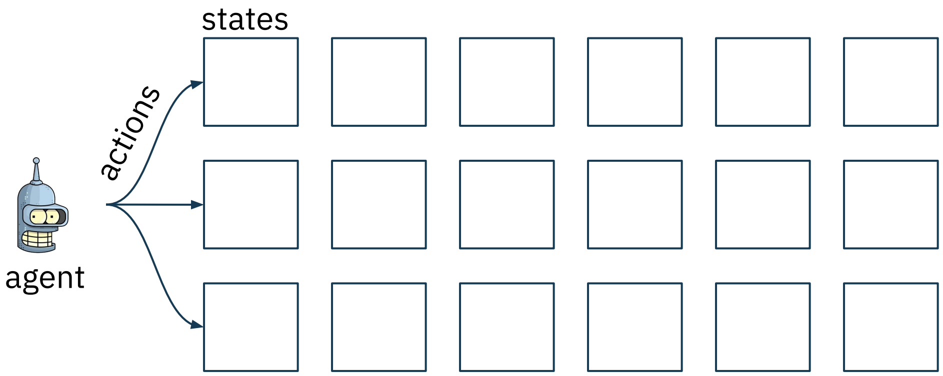
Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



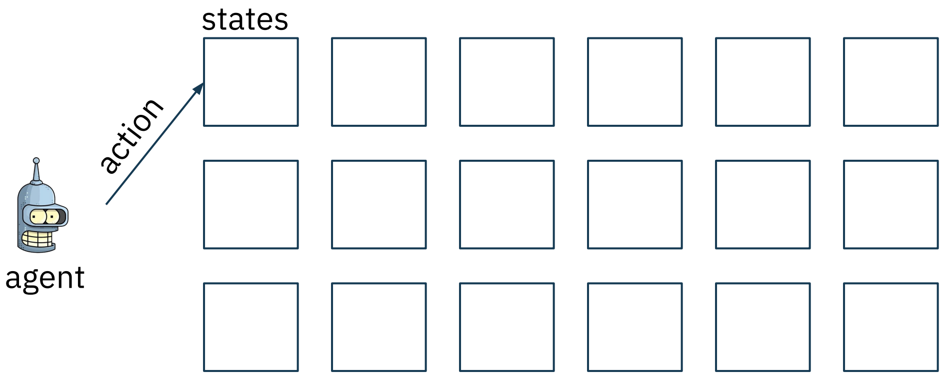
A definition

Reinforcement Learning	technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

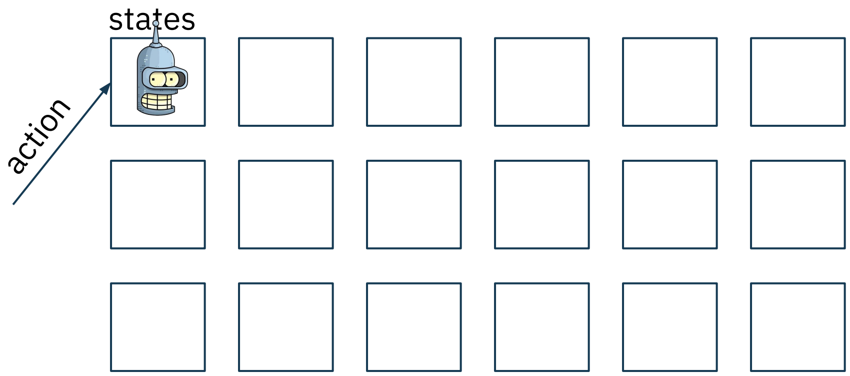
Reinforcement Learning: technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

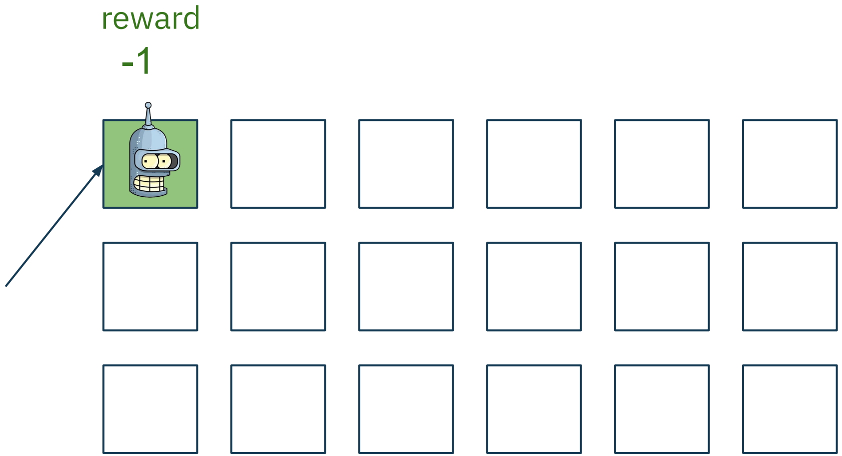
Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

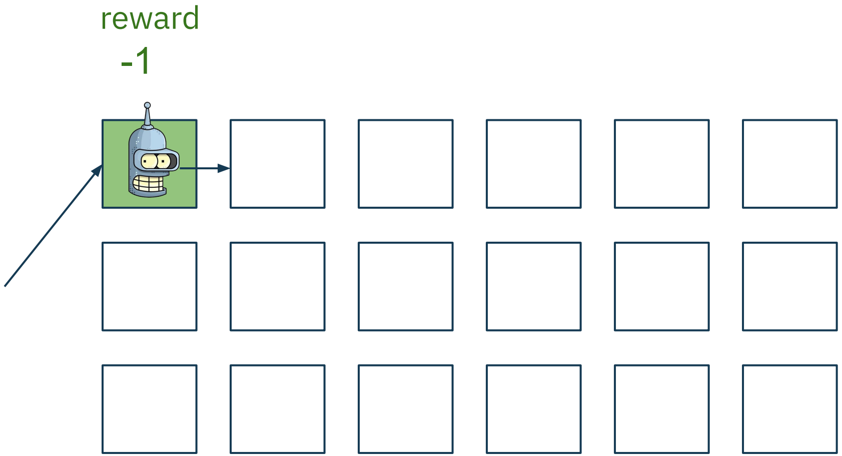
Reinforcement Learning	technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

Reinforcement Learning

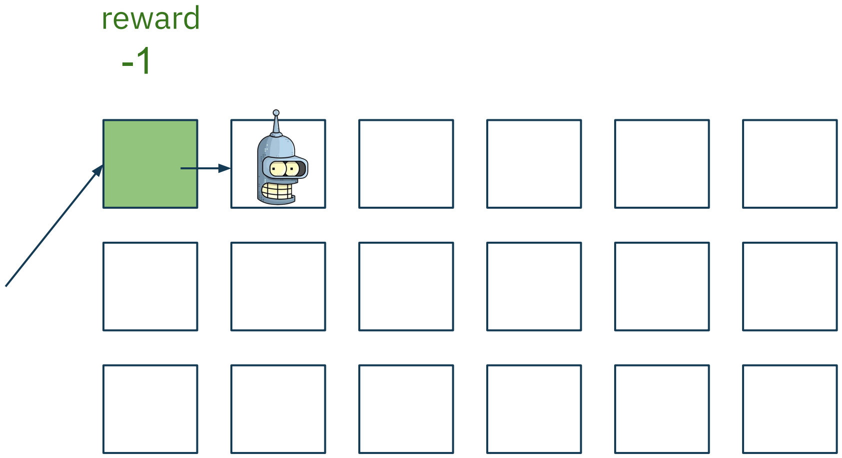
technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



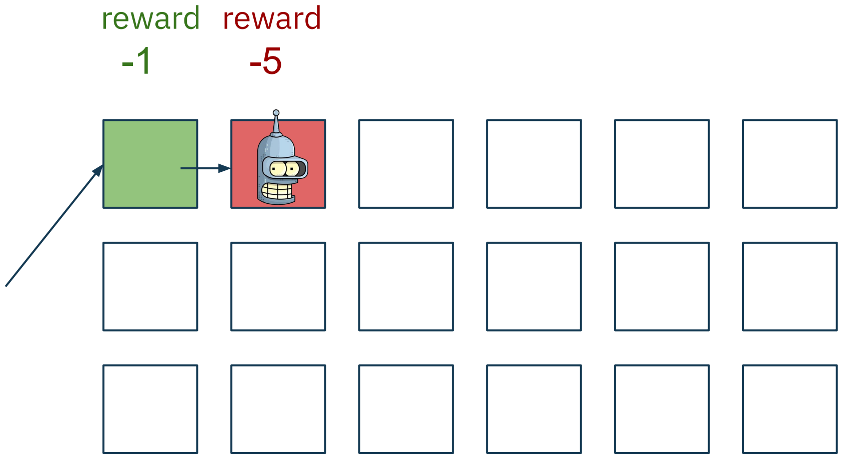
A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**

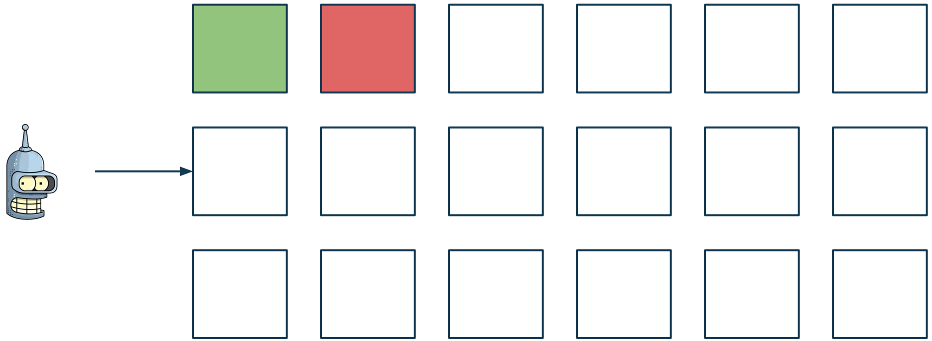
Cumulative Reward

-6



A definition

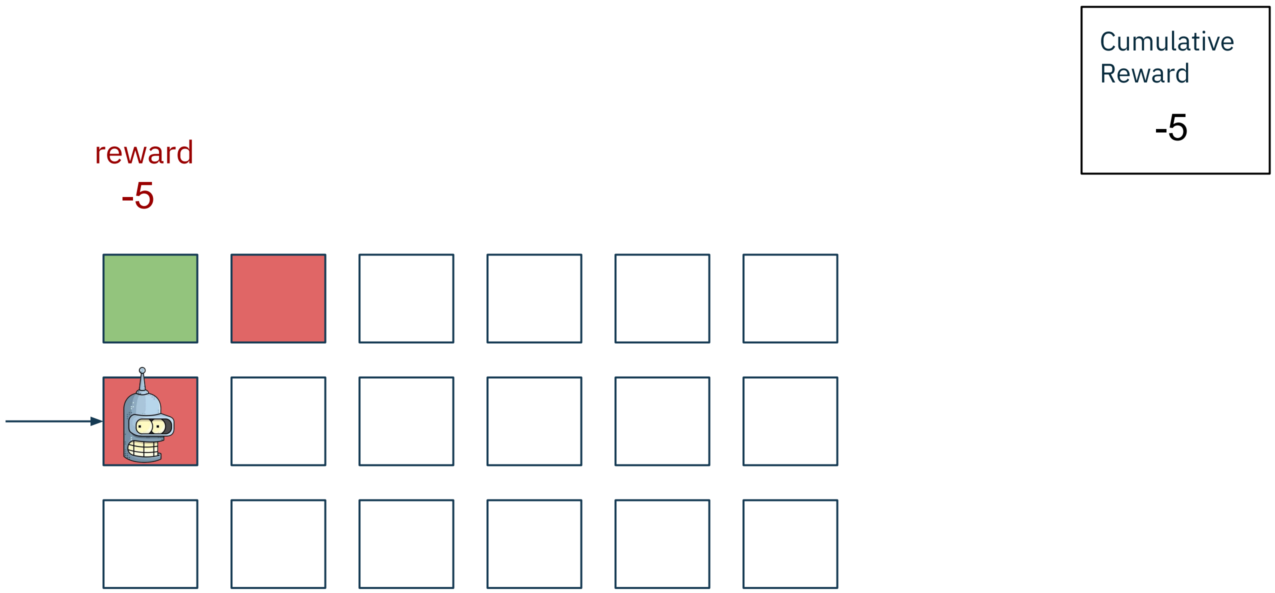
Reinforcement Learning	technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

Reinforcement Learning

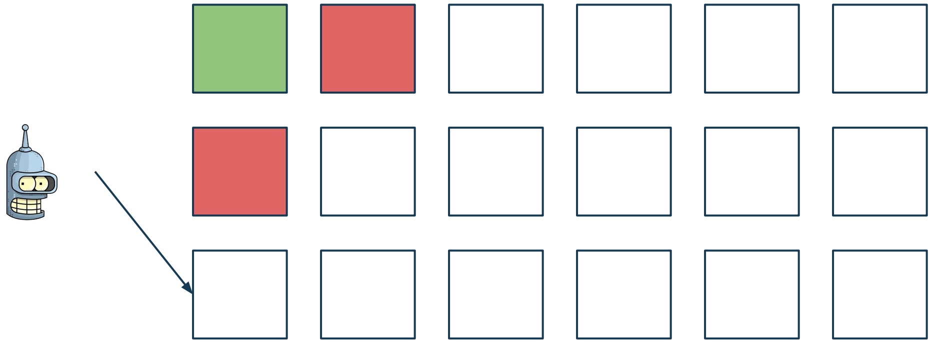
technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**



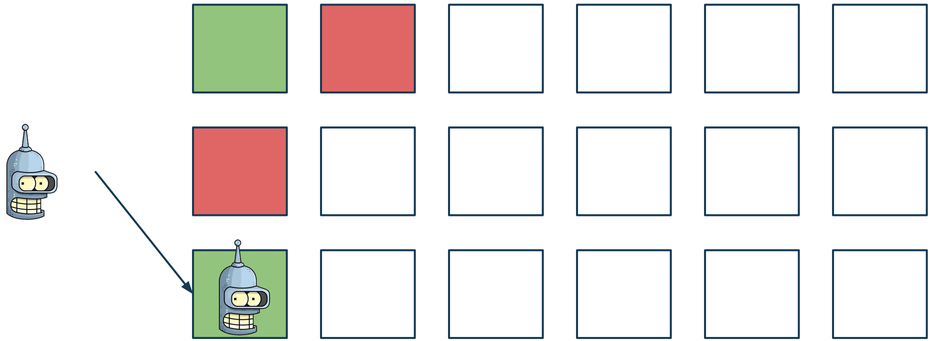
A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**

Cumulative Reward

-1



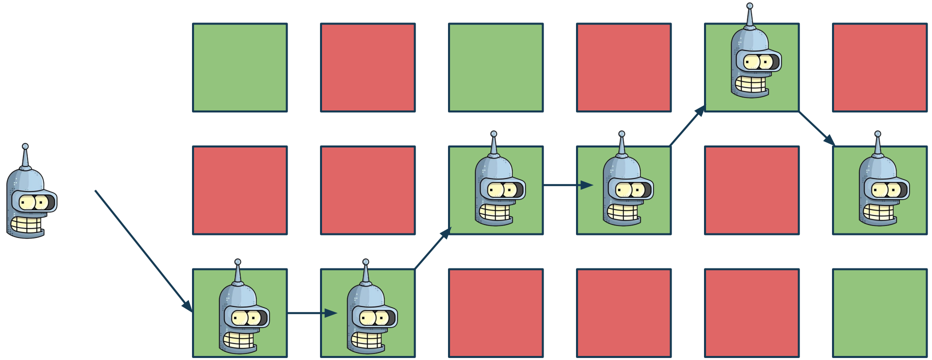
A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**

Cumulative Reward

-6



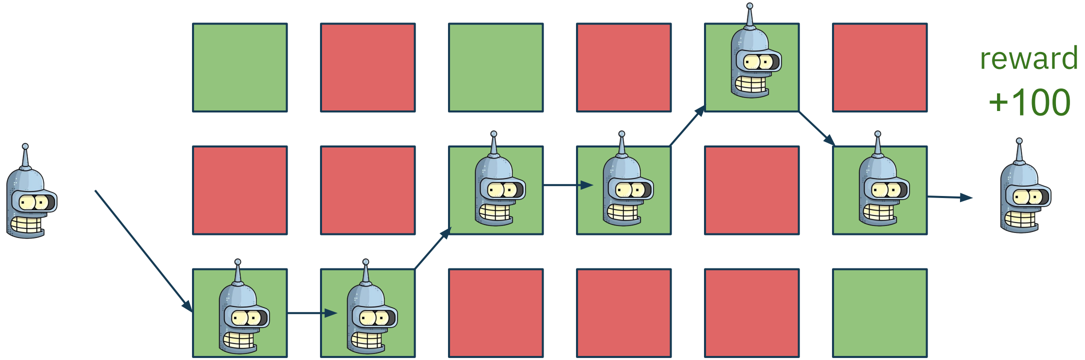
A definition

Reinforcement Learning

technique that trains **agents** to **map states into actions** to **maximize a cumulative reward**

Cumulative Reward

94



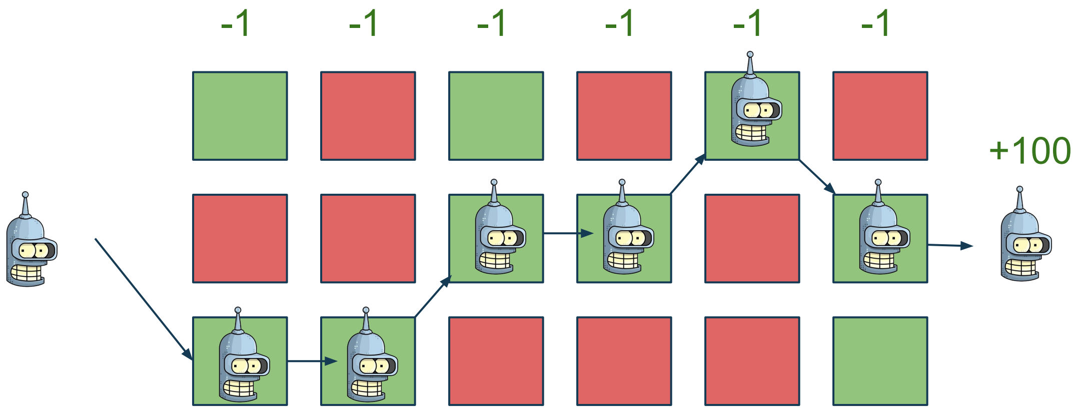
On reward

Goal and reward coherence

we want the agent goes as fast as possible from A to B. We need to choose an appropriate reward signal!

Cumulative Reward

94



On reward

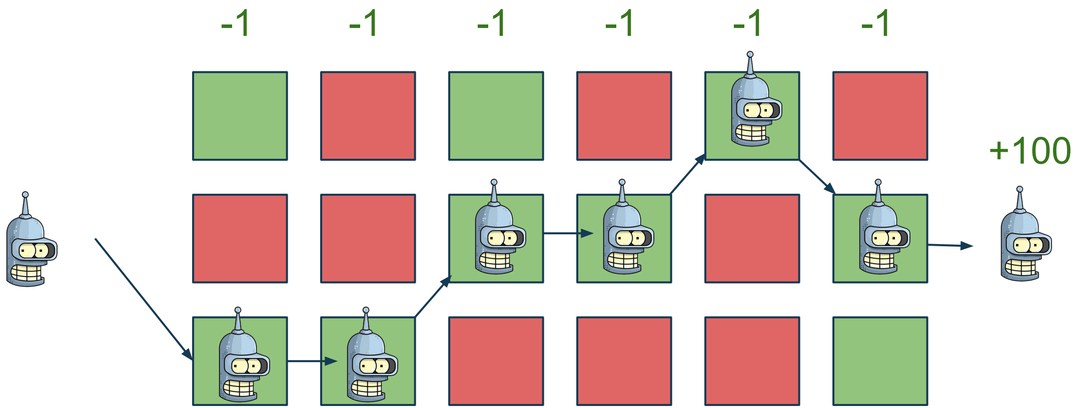
Goal and reward coherence

we want the agent goes as fast as possible from A to B. We need to choose an appropriate reward signal!

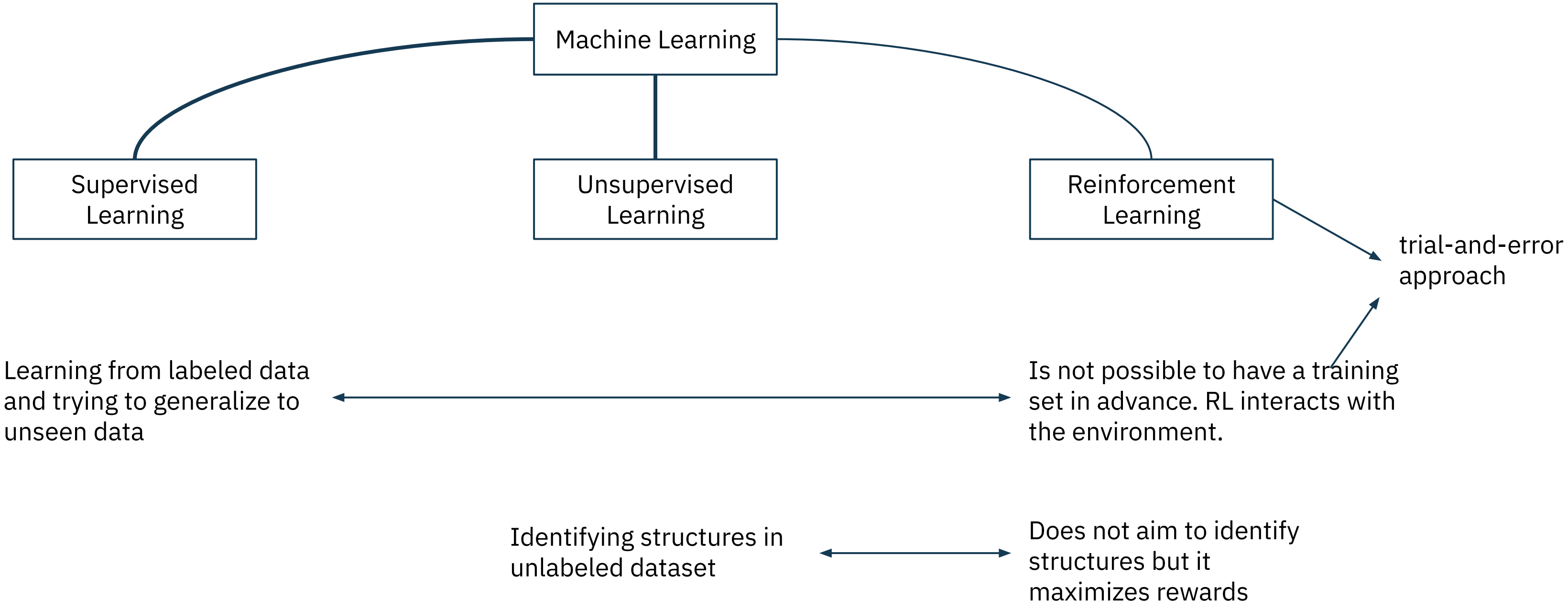
Cumulative Reward

94

Why negative values?



A definition



Elements of RL

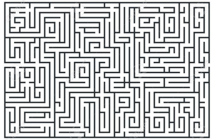
Mathematical definition

List of the ingredients

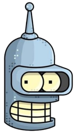
Elements of Reinforcement Learning

Environment	usually exposes uncertainty
Agent	is a decision-maker and has a goal
Policy (of the agent)	defines the agent behaviour
Reward signal	defines the goal of the agent
Model of the environment	exists model-free and model-based RL algorithms
Value function (of a policy)	quantifies how good is for an agent to be in a specific state (if it follows that policy)

Observability

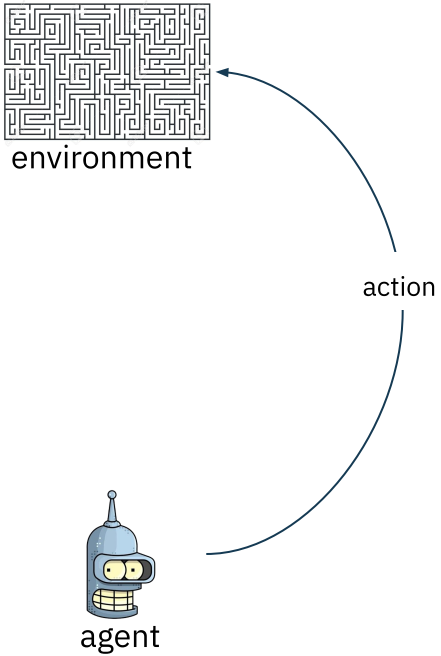


environment

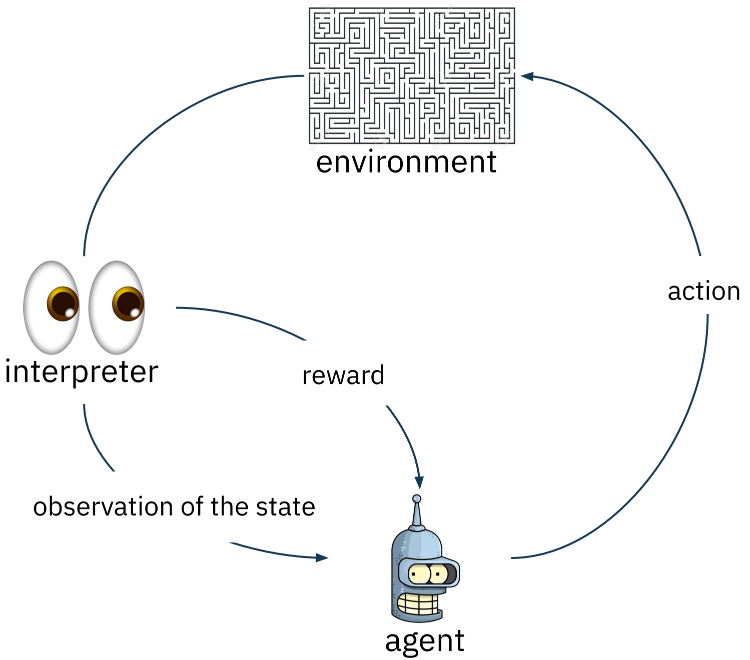


agent

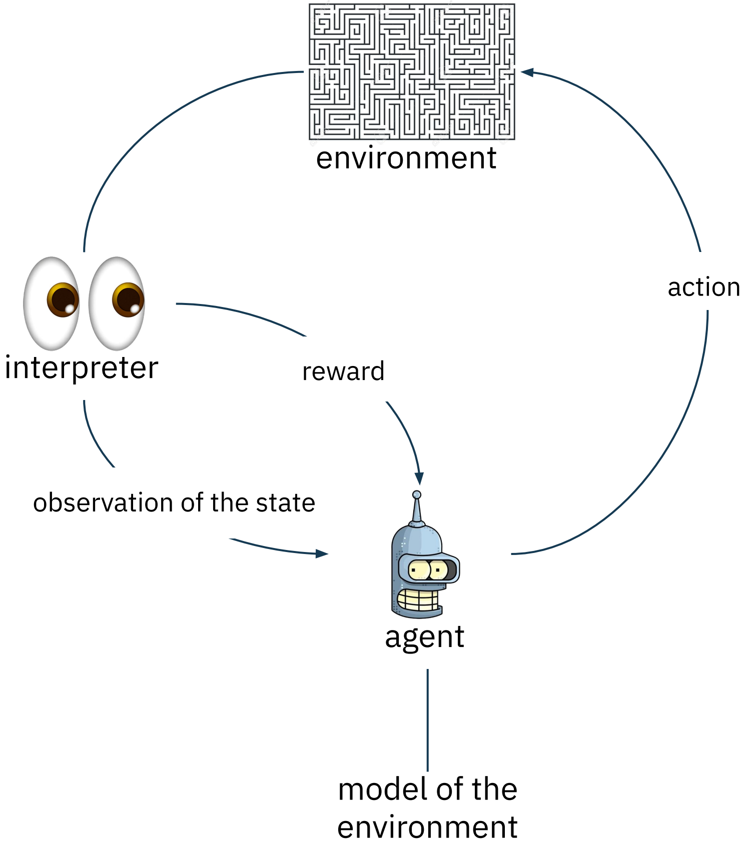
Observability



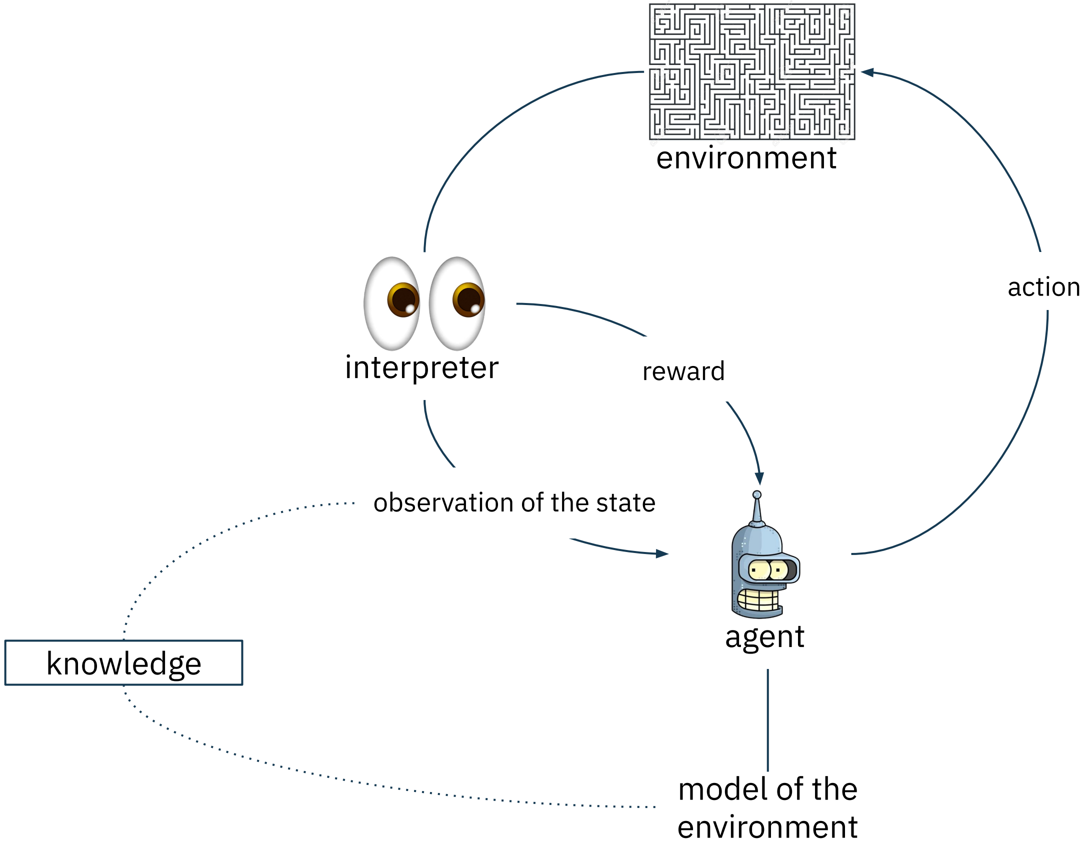
Observability



Observability



Observability



Observability

RL = learning + prediction + controlling

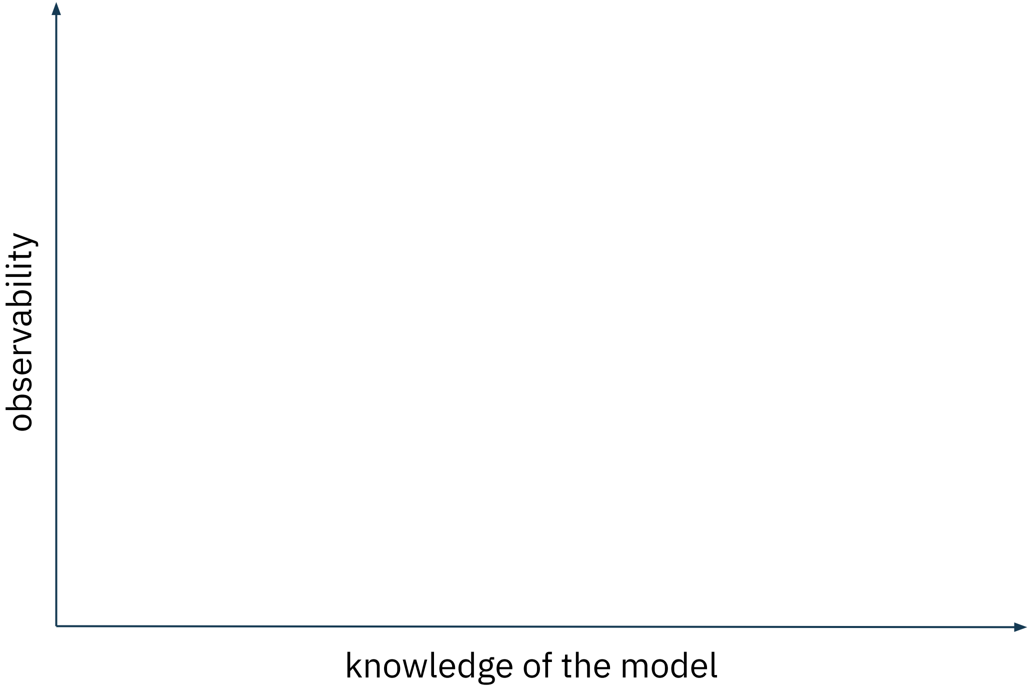
Building a model of the environment

Knowing the cumulative reward I'll get following a policy

Discovering the best action

Knowledge of the environment

The two axes of knowledge

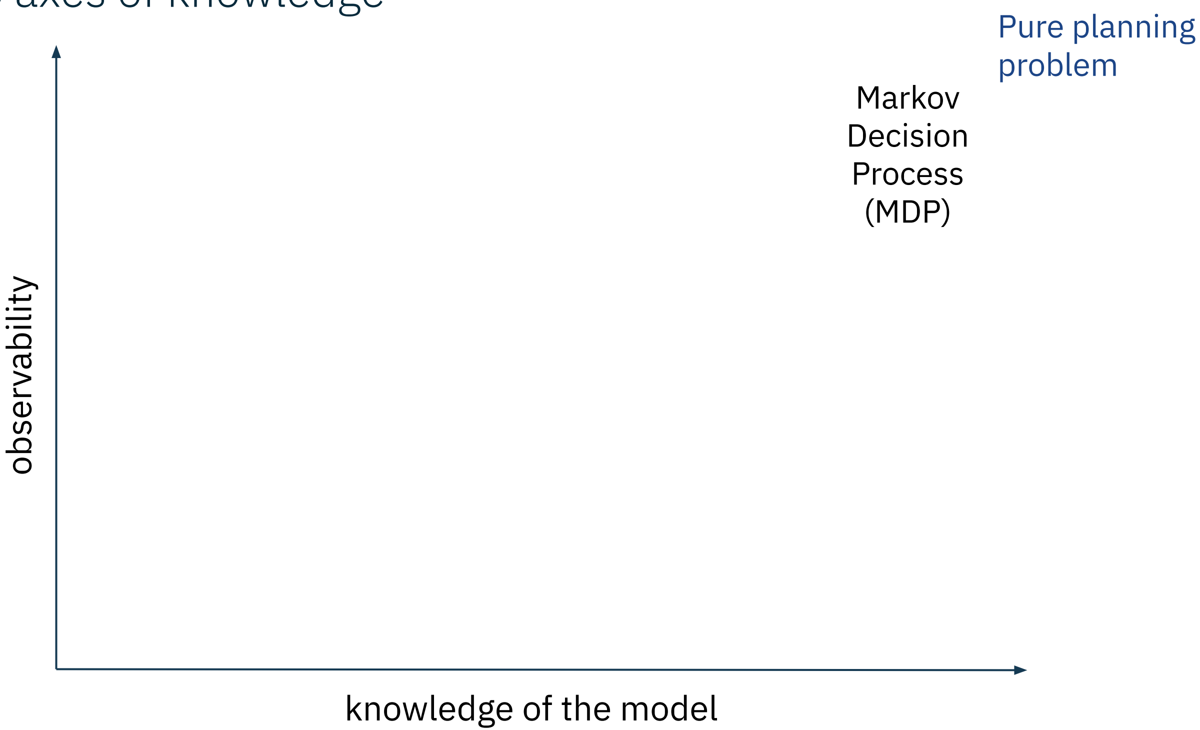


Empirical knowledge ⟺ Epistemic knowledge

Knowledge of the environment

The two axes of knowledge

<u>Markovian process</u>
only matter knowledge
of the actual state



Empirical knowledge ⟺ Epistemic knowledge

Knowledge of the environment

The two axes of knowledge

<u>Markovian process</u>
only matter knowledge of the actual state

Pure planning problem

Markov Decision Process (MDP)

inference

Partially Observable MDP

Planning with uncertainty

We have model but some params are unknown

observability

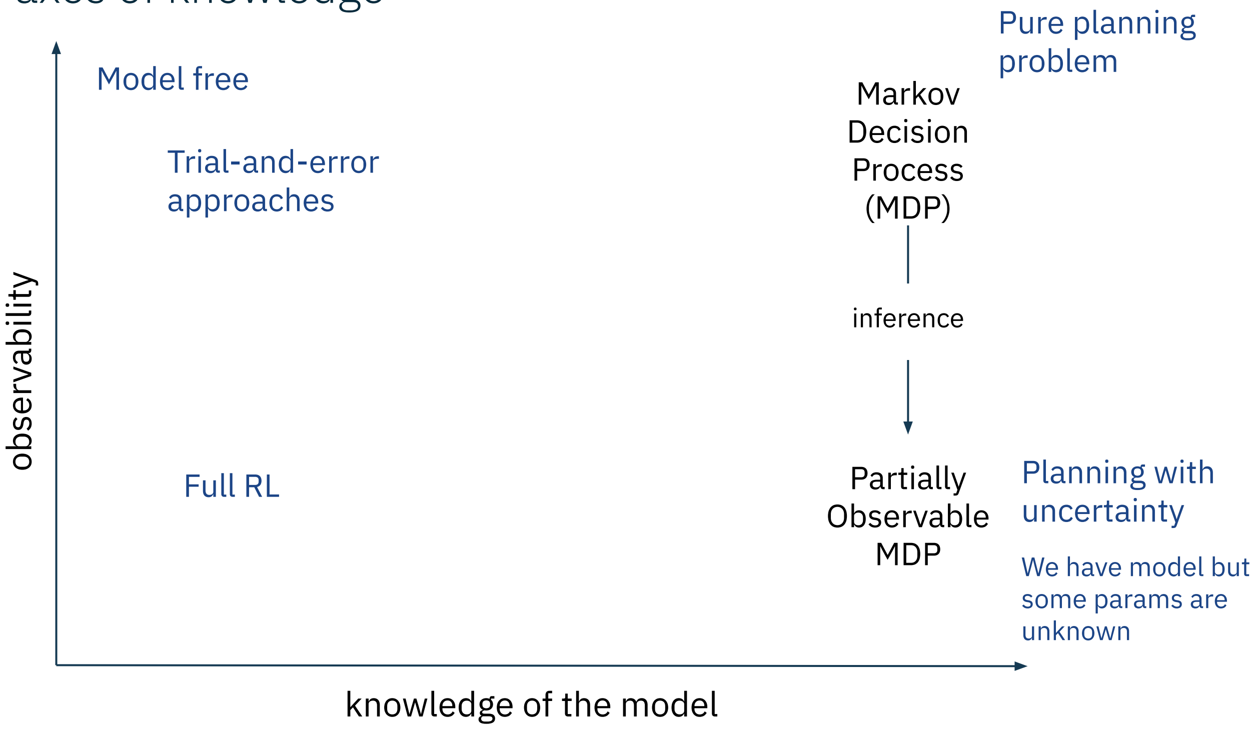
knowledge of the model

Empirical knowledge ⟺ Epistemic knowledge

Markovian process
only matter knowledge of the actual state

Knowledge of the environment

The two axes of knowledge

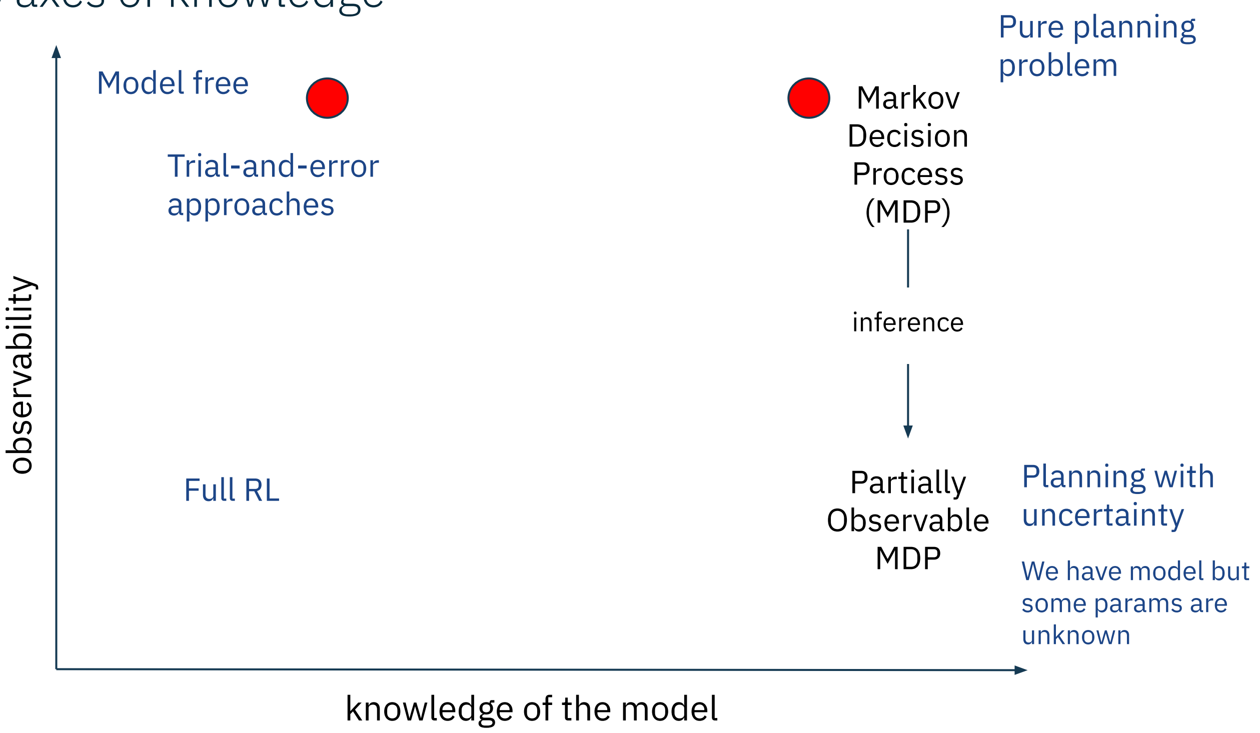


Empirical knowledge ⟺ Epistemic knowledge

Knowledge of the environment

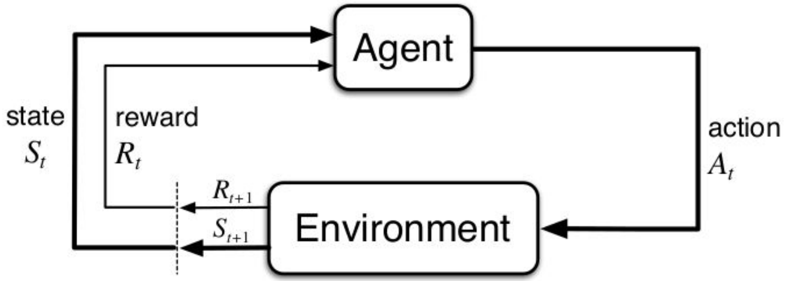
The two axes of knowledge

<u>Markovian process</u>
only matter knowledge
of the actual state



Empirical knowledge ⟺ Epistemic knowledge

(finite) Markov Decision Process



trajectory

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \dots$$

dynamics

$$p(s', r \mid s, a) \doteq \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

$$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r \mid s, a) = 1, \text{ for all } s \in \mathcal{S}, a \in \mathcal{A}(s)$$

Perfect knowledge
of the model

(finite) Markov Decision Process

trajectory

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots$$

dynamics

$$p(s', r \mid s, a) \doteq \Pr\{S_t = s', R_t = r \mid S_{t-1} = s, A_{t-1} = a\}$$

state-transition probability

$$p(s' \mid s, a) \doteq \Pr\{S_t = s' \mid S_{t-1} = s, A_{t-1} = a\} = \sum_{r \in \mathcal{R}} p(s', r \mid s, a)$$

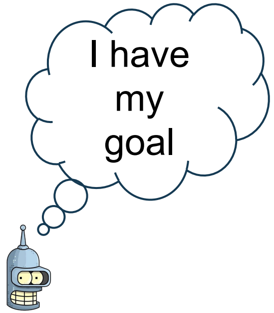
expected reward (I)

$$r(s, a) \doteq \mathbb{E}[R_t \mid S_{t-1} = s, A_{t-1} = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r \mid s, a)$$

expected reward (II)

$$r(s, a, s') \doteq \mathbb{E}[R_t \mid S_{t-1} = s, A_{t-1} = a, S_t = s'] = \sum_{r \in \mathcal{R}} r \frac{p(s', r \mid s, a)}{p(s' \mid s, a)}$$

Reward signal



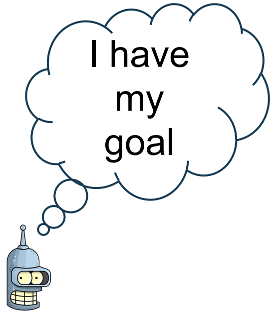
Reward hypothesis: that all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).

Reward R_{t+1}

Return $G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$

Discounted Return $G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

Reward signal



Reward hypothesis: that all of what we mean by goals and purposes can be well thought of as the maximization of the expected value of the cumulative sum of a received scalar signal (called reward).

Short-term view

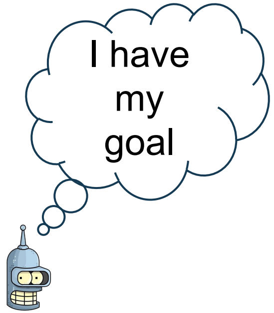
Reward R_{t+1}

Long-term view

Return $G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$

Discounted Return $G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

Reward signal



Reward hypothesis: that all of what we mean by <u>goals and purposes</u> can be well thought of as the <u>maximization of the expected value of the cumulative sum of a received scalar signal (called reward).</u>

Short-term view

Long-term view

Reward R_{t+1}

Return $G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$

Discounted Return $G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

$$\Downarrow$$

$$\boxed{G_t = R_{t+1} + \gamma G_{t+1}}$$

RECURSIVE DEFINITION

Policy

Policy

is a mapping from states to probabilities of selecting each possible action

$$\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$$

If we are at time t, $\pi(a|s)$ is the probability of having $A_t = a \wedge S_t = s$

can be *deterministic*

Value function

Value Function

is a function that quantify how good is to be on a state and follows a specific policy

$$v_\pi : \mathcal{S} \to \mathbb{R}$$

state-value function

$$v_\pi(s) \doteq \mathbb{E}_\pi[G_t \mid S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s\right], \text{ for all } s \in \mathcal{S}$$

action-value function

$$q_\pi(s,a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \,\middle|\, S_t = s, A_t = a\right]$$

Solving a RL problem

find a policy that achieves the maximum reward over the long run

optimal policy

$\pi_* \succeq \pi \quad \forall \pi \in$ policies

Solving a RL problem

find a policy that achieves the maximum reward over the long run

optimal policy

$\pi_* \succeq \pi \quad \forall \pi \in$ policies

$$\pi' \succeq \pi \iff \forall s \in \mathcal{S},\ v_{\pi'}(s) \geq v_\pi(s)$$

Solving a RL problem

find a policy that achieves the maximum reward over the long run

optimal policy

$\pi_* \succeq \pi \quad \forall \pi \in$ policies

$$\pi' \succeq \pi \iff \forall s \in \mathcal{S},\ v_{\pi'}(s) \geq v_\pi(s)$$

optimal state-value function

$$v_*(s) \doteq \max_\pi v_\pi(s)$$

optimal action-value function

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

Solving a RL problem

find a policy that achieves the maximum reward over the long run

optimal policy

$$\pi_* \succeq \pi \quad \forall \pi \in \text{policies}$$

$$\pi' \succeq \pi \iff \forall s \in \mathcal{S},\ v_{\pi'}(s) \geq v_\pi(s)$$

optimal state-value function

$$v_*(s) \doteq \max_\pi v_\pi(s)$$

optimal action-value function

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

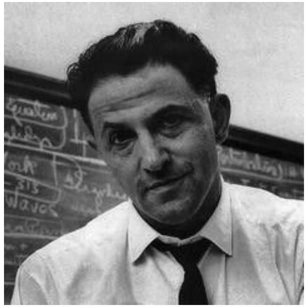
$$q_*(s,a) = \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a]$$

Dynamic Programming

How to solve MDP problems

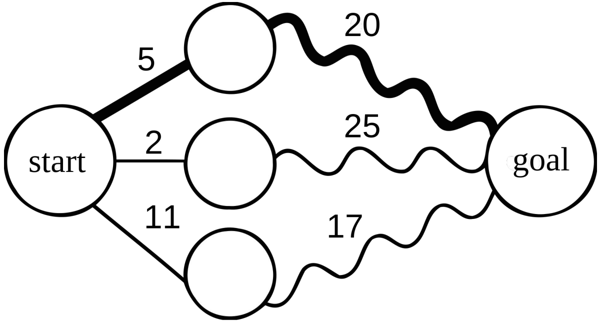
Dynamic Programming

Mr. Richard Ernest Bellman



Bellman, 1950s

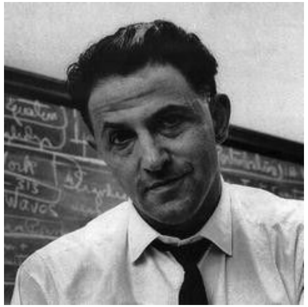
Algorithm paradigm useful to solve a specific class of problems that can be decomposed in sub-problems in recursive way



Dynamic Programming

In the RL context

Collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as a MDP.



Bellman, 1950s

Key idea: use value function to organize and structure the search of optimal policies

Consistency relation of state-value function

$$
\begin{aligned}
v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s'} \sum_r p(s', r|s, a) \Big[r + \gamma \mathbb{E}_\pi[G_{t+1}|S_{t+1} = s'] \Big] \\
&= \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) \Big[r + \gamma v_\pi(s') \Big], \quad \text{for all } s \in \mathcal{S}.
\end{aligned}
$$

Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

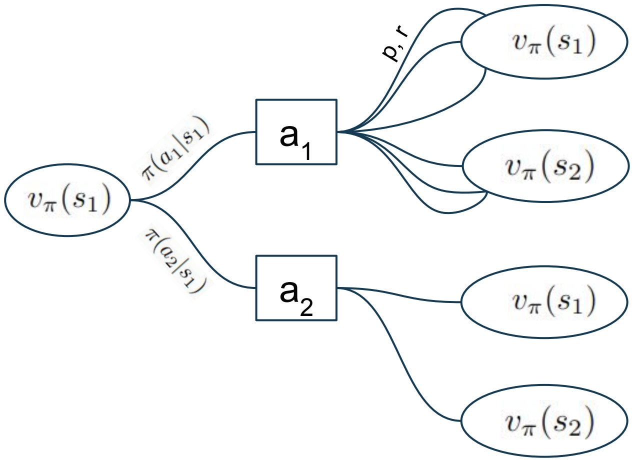
$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right]$$

Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\pi(s')\right]$$

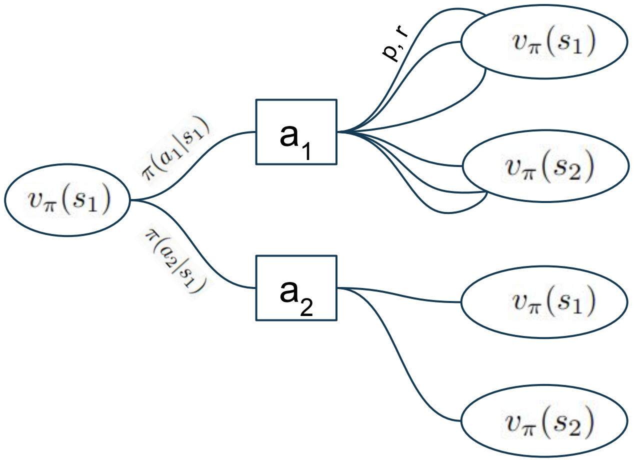


Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right]$$

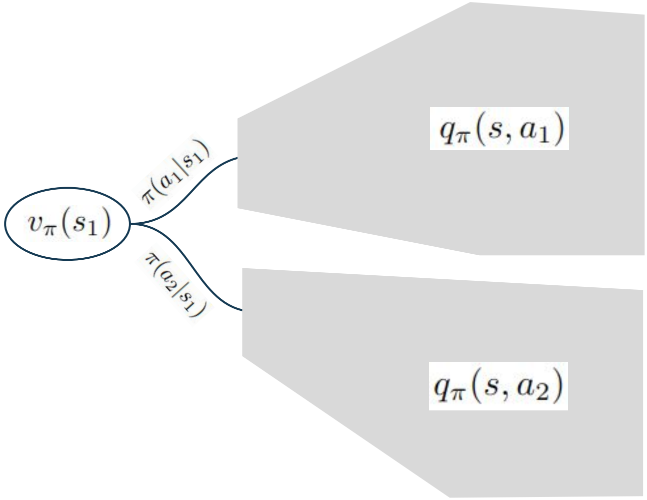


Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \left[r + \gamma v_\pi(s')\right]$$

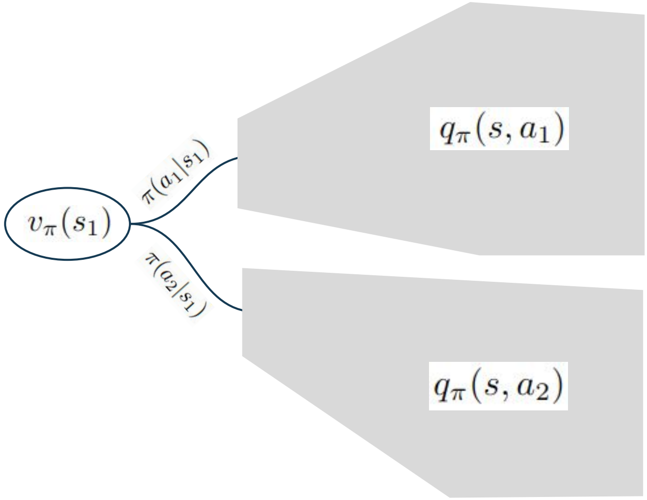


Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$$



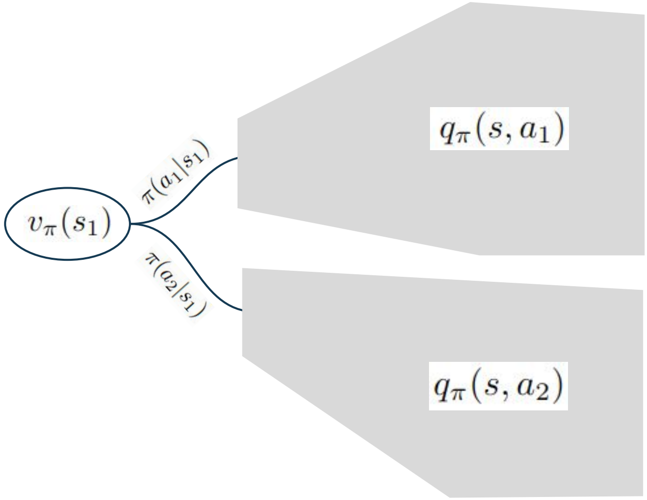
Dynamic Programming

Towards the Bellman Equation

What about the optimal policy and the optimal state-value function?

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$$



Dynamic Programming

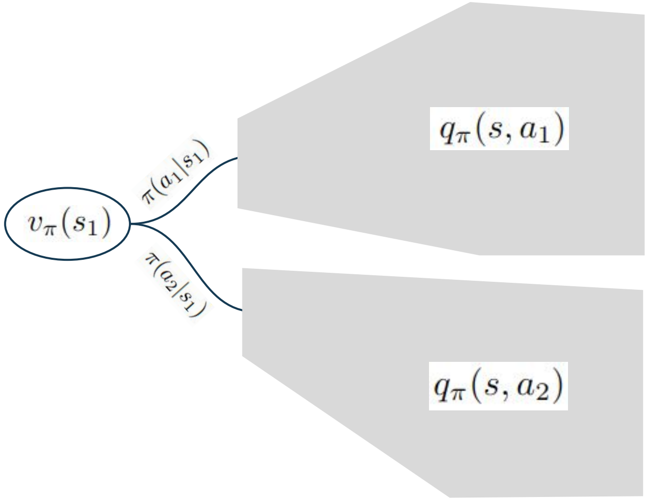
Towards the Bellman Equation

What about the optimal policy and the optimal state-value function?

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$$

It's an average



Dynamic Programming

Towards the Bellman Equation

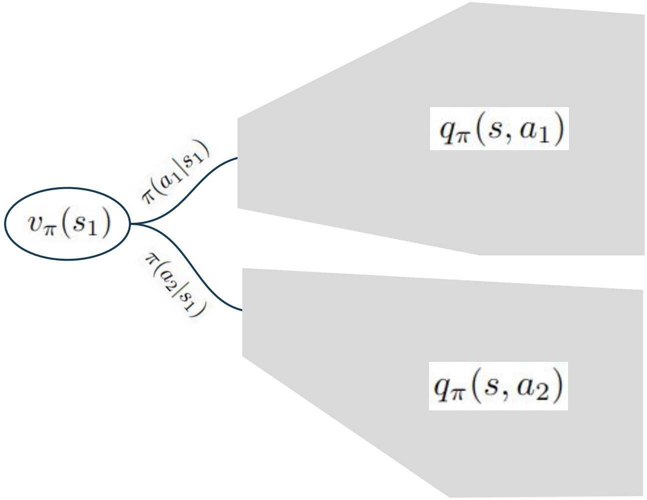
What about the optimal policy and the optimal state-value function?

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s,a)$$

It's an average

The optimal policy is a policy so it should satisfy the consistency relation



Dynamic Programming

Towards the Bellman Equation

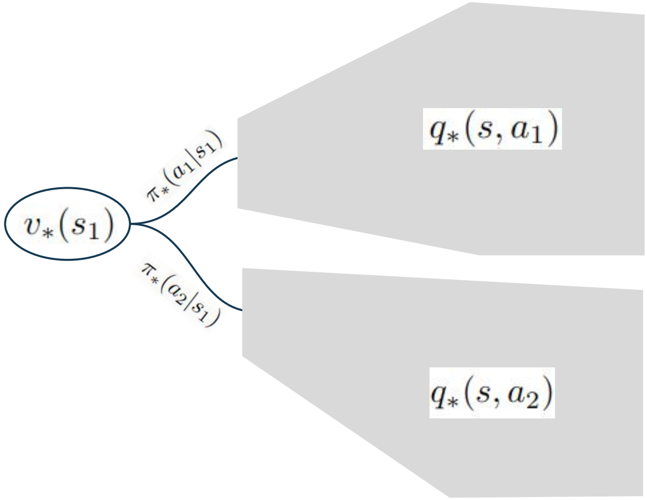
What about the optimal policy and the optimal state-value function?

Consistency relation of state-value function

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

It’s an average

The optimal policy is a policy so it should satisfy the consistency relation

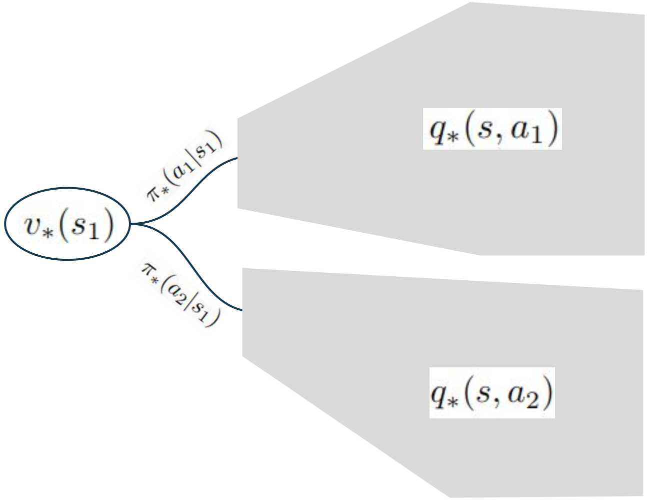


Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

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



What about the optimal policy and the optimal state-value function?

It's an average

The optimal policy is a policy so it should satisfy the consistency relation

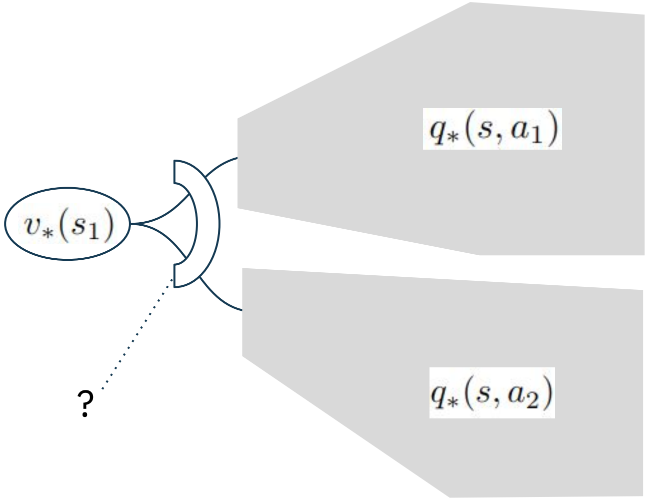
The optimal policy is *optimal*

Dynamic Programming

Towards the Bellman Equation

Consistency relation of state-value function

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



What about the optimal policy and the optimal state-value function?

It's an average

The optimal policy is a policy so it should satisfy the consistency relation

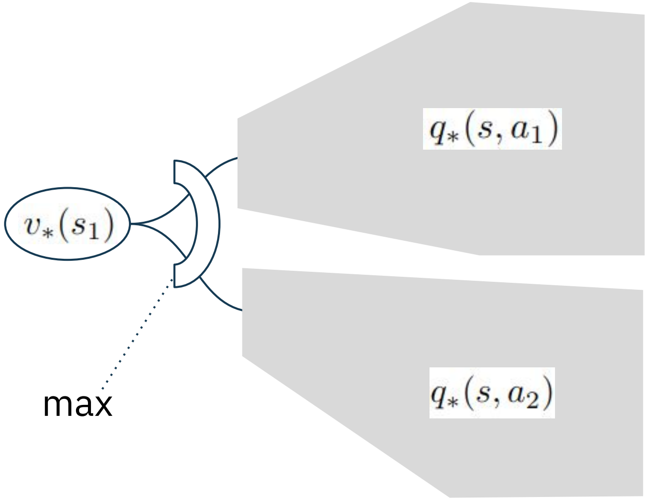
The optimal policy is *optimal*

Dynamic Programming

Bellman Equation

Bellman equation

$$v_*(s) = \max_a q_*(s,a)$$



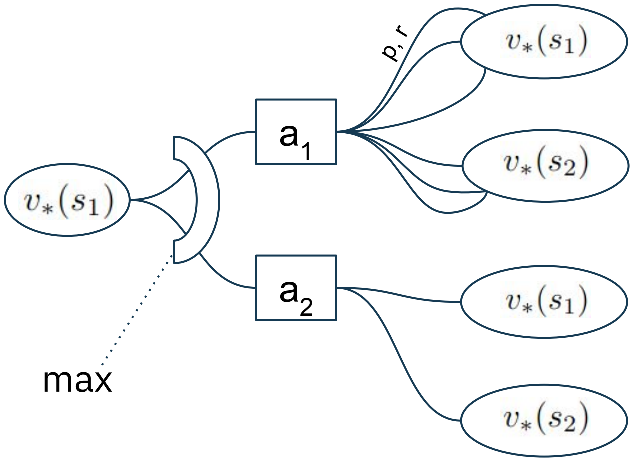
Dynamic Programming

Bellman Equation

Bellman equation

$$v_*(s) = \max_a q_*(s,a)$$

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')]$$



Dynamic Programming

Bellman Equation

Bellman equation

$$v_*(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_*(s')]$$

$$q_*(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma \max_{a'} q_*(s', a')]$$

Dynamic Programming

How to find the optimal policy?

$$\pi \xrightarrow{\text{Iterative procedure}} \pi^*$$

Dynamic Programming

How to find the optimal policy?

Consistency relation of the state-value function

$$\pi \xrightarrow{\text{Policy evaluation}} v_{\pi} \longrightarrow \pi'$$

Dynamic Programming

How to find the optimal policy?

Bellman intuition

Consistency relation of the state-value function

$$\pi \xrightarrow{\text{Policy evaluation}} v_{\pi} \xrightarrow{\text{Policy improvement}} \pi'$$

Dynamic Programming

How to find the optimal policy?

Bellman intuition

Consistency relation of the state-value function

$$\begin{tikzcd} \pi \arrow[r, "\text{Policy evaluation}"] & v_\pi \arrow[r, "\text{Policy improvement}"] & \pi' \arrow[ll, bend left, "\text{Policy Iteration}"] \end{tikzcd}$$

Dynamic Programming

How to find the optimal policy?

Bellman intuition

Consistency relation of the state-value function

$$\begin{tikzcd} \pi \arrow[r, "\text{Policy evaluation}"] & v_\pi \arrow[r, "\text{Policy improvement}"] & \pi' \arrow[ll, bend left] \end{tikzcd}$$

Policy Iteration

$$\pi_0 \xrightarrow{\mathrm{E}} v_{\pi_0} \xrightarrow{\mathrm{I}} \pi_1 \xrightarrow{\mathrm{E}} v_{\pi_1} \xrightarrow{\mathrm{I}} \pi_2 \xrightarrow{\mathrm{E}} \cdots \xrightarrow{\mathrm{I}} \pi_* \xrightarrow{\mathrm{E}} v_*$$

Does it converge? Yes

Dynamic Programming

Policy evaluation

$$\pi \xrightarrow{\text{Policy evaluation}} v_\pi$$

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right]$$

Dynamic Programming

Policy evaluation

$$\pi \xrightarrow{\text{Policy evaluation}} v_\pi$$

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right]$$

Iterative policy evaluation

$$v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_k(s')]$$

Dynamic Programming

Policy evaluation

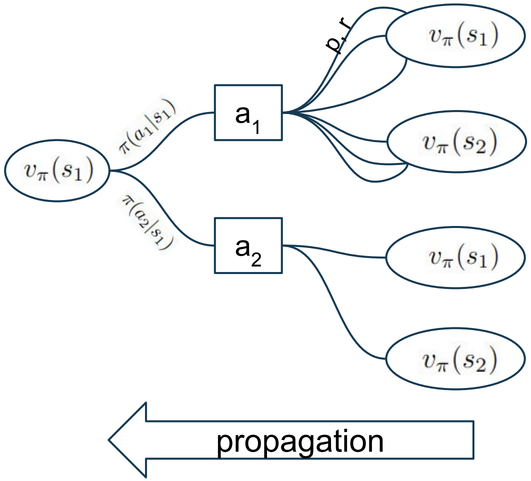
$$\pi \xrightarrow{\text{Policy evaluation}} v_\pi$$

Consistency relation of state-value function

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\pi(s')\right]$$

Iterative policy evaluation

$$v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_k(s')]$$



<u>2 ways of updating</u>: *in-place* vs *two arrays version*

Faster, depends on ordering of update

Policy Evaluation

Algorithm

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Policy Evaluation

Algorithm

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$ consistency relation
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Policy Evaluation

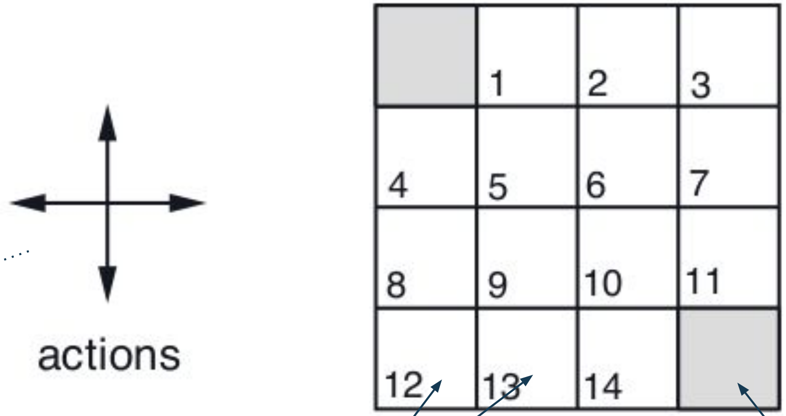
Algorithm

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s',r \,|\, s,a) \big[r + \gamma V(s')\big]$ consistency relation
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ Stability of state-value function

Policy Evaluation
Example



Uniform Policy

$R_t = -1$
on all transitions

non-terminal state

terminal state

Policy Evaluation

Example - 1st iteration

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

0.0			
			0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

Policy Evaluation

Example - 1st iteration

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

0.0			
			0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[\underbrace{r}_{-1} + \underbrace{\gamma v_\pi(s')}_{0}]$$

Policy Evaluation

Example - 1st iteration

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

$\Rightarrow$

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

Policy Evaluation

Example - 1st iteration

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

0.0	?		
?			
			?
		?	0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

Policy Evaluation

Example - 2nd iteration

← -1/3 ✚

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

⇒

0.0	?		
?			
			?
		?	0.0

$$v_\pi(s) = \sum_a \underbrace{\pi(a|s)}_{1/3} \sum_{s',r} \underbrace{p(s',r|s,a)}_{1}[\underbrace{r}_{-1} + \gamma \underbrace{v_\pi(s')}_{0}]$$

Policy Evaluation

Example - 2nd iteration

← -1/3 +

↓ -2/3 +

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

⇒

0.0	?		
?			
			?
		?	0.0

$$v_\pi(s) = \sum_a \underbrace{\pi(a|s)}_{1/3} \sum_{s',r} \underbrace{p(s',r|s,a)}_{1}[\underbrace{r}_{-1} + \gamma \underbrace{v_\pi(s')}_{-1}]$$

Policy Evaluation

Example - 2nd iteration

← -1/3 +

↓ -2/3 +

→ -2/3 +

$\boxed{-1.7}$

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

0.0	?		
?			
			?
		?	0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

Policy Evaluation

Example - 2nd iteration

$\leftarrow$ -1/3 +

$\downarrow$ -2/3 +

$\rightarrow$ -2/3 +

$\boxed{-1.7}$

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

$\Rightarrow$

0.0	-1.7		
-1.7			
			-1.7
		-1.7	0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

Policy Evaluation

Example - 2nd iteration

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

$\Rightarrow$

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

Policy Evaluation

Example - until the end

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

1

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

2

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

3

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

∞

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Policy Evaluation

Example - until the end

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

1

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

2

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

3

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

∞

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Policy Evaluation

Example - until the end

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

1

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

2

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

3

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

∞

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	$\boxed{-20.}$	-18.	-14.
-22.	-20.	-14.	0.0

Policy Evaluation

Example - until the end

0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0

1

0.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	-1.0
-1.0	-1.0	-1.0	0.0

2

0.0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0.0

3

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0

∞

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Policy Improvement

How to find better policies

$$v_\pi \xrightarrow{\text{Policy improvement}} \pi'$$

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0

Policy improvement theorem

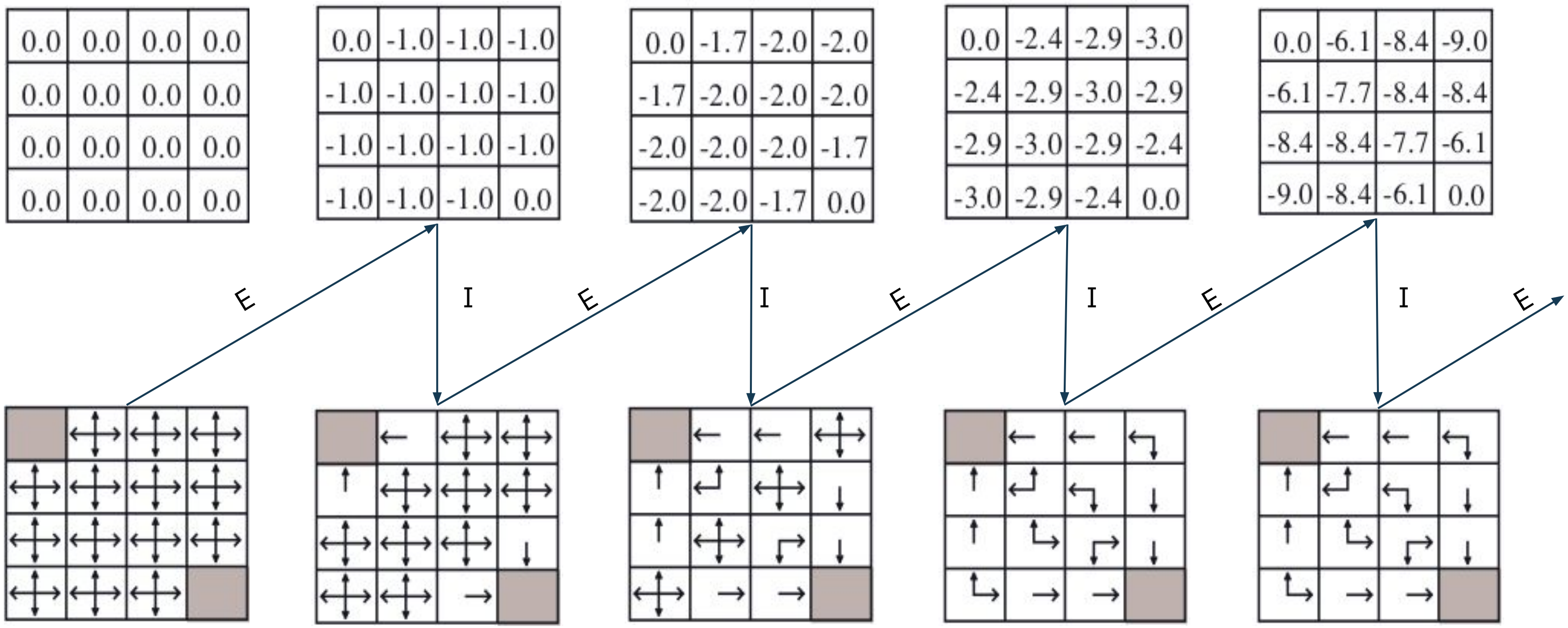
$$q_\pi(s, \pi'(s)) \geq v_\pi(s),\ \forall s \in \mathcal{S} \Rightarrow v_{\pi'}(s) \geq v_\pi(s),\ \forall s \in \mathcal{S}$$

Greedy policy approach

$$\begin{aligned}
\pi'(s) &\doteq \arg\max_a q_\pi(s,a) \\
&= \arg\max_a \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \arg\max_a \sum_{s',r} p(s', r \mid s, a)\Big[r + \gamma v_\pi(s')\Big],
\end{aligned}$$

Policy Iteration

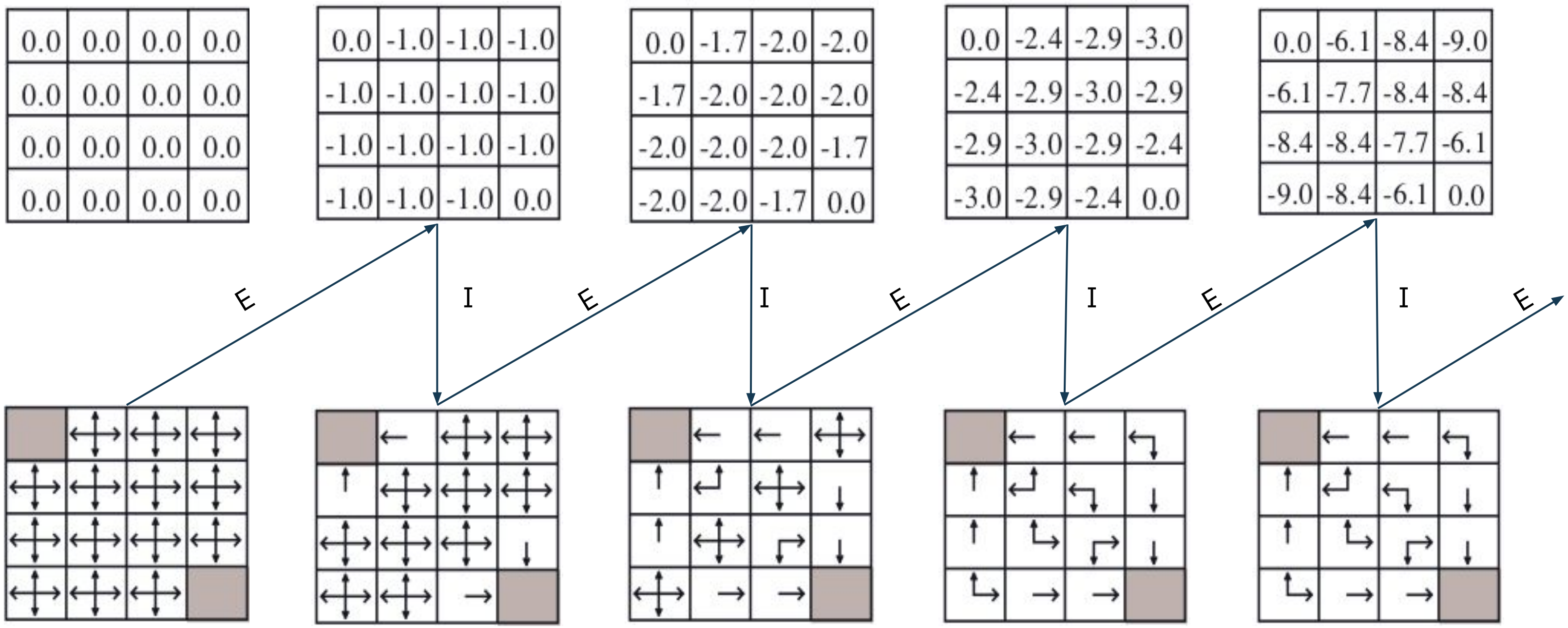
Example



Policy Iteration

Example

propagation effect

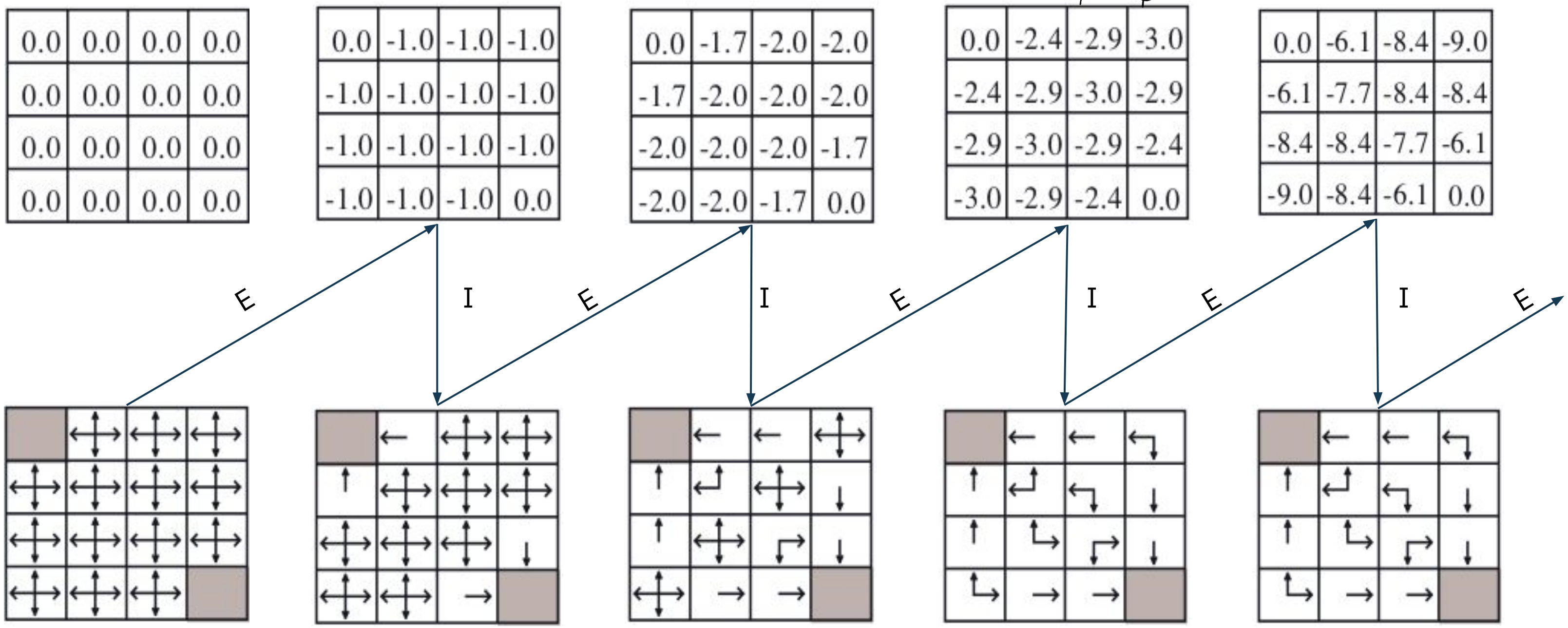


policy convergence

Policy Iteration

Example

propagation effect



policy convergence

Policy Iteration

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\quad \Delta \leftarrow 0$
   $\quad$ Loop for each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad V(s) \leftarrow \sum_{s',r} p(s',r|s,\pi(s))\big[r + \gamma V(s')\big]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
   $\quad$ If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Policy Iteration

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   - $\Delta \leftarrow 0$
   - Loop for each $s \in \mathcal{S}$:
     - $v \leftarrow V(s)$
     - $V(s) \leftarrow \sum_{s',r} p(s', r | s, \pi(s))\big[r + \gamma V(s')\big]$
     - $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   - *old-action* $\leftarrow \pi(s)$
   - $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r | s, a)\big[r + \gamma V(s')\big]$
   - If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*

   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Value Iteration

Solving efficiently the Policy Iteration

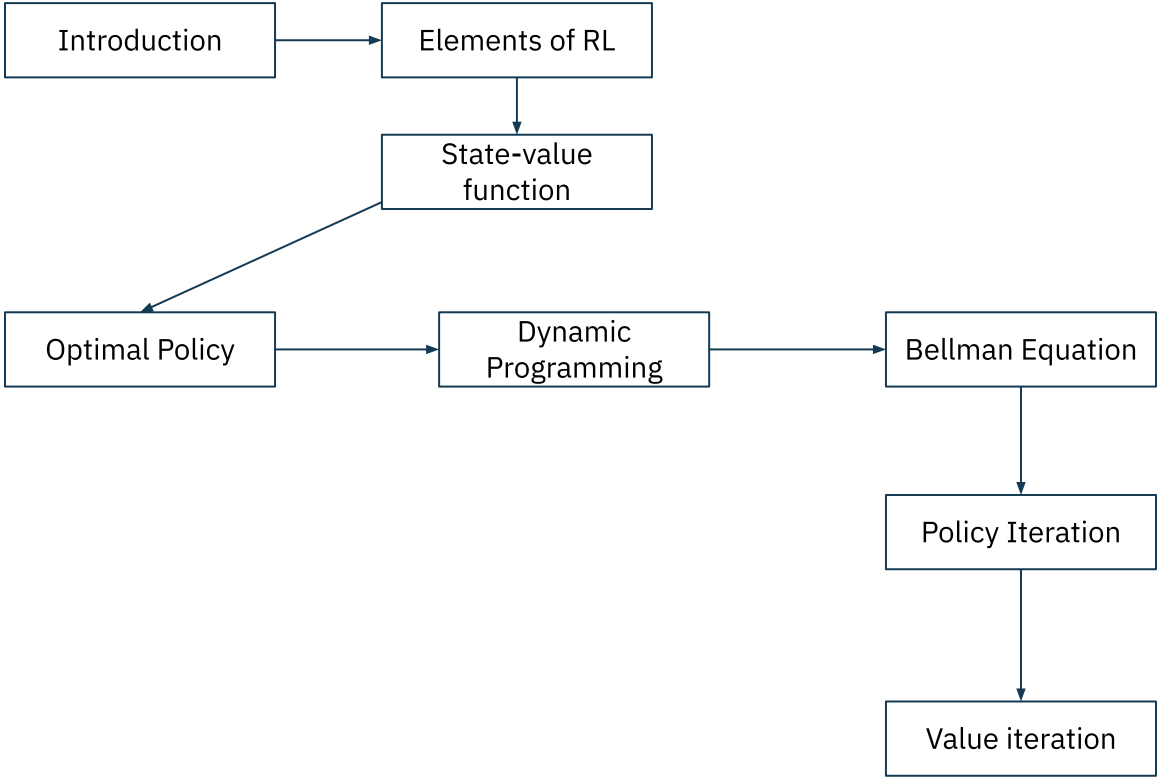
Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$

Recap


Introduction
Elements of RL
State-value function
Optimal Policy
Dynamic Programming
Bellman Equation
Policy Iteration
Value iteration

Thank you

esteco.com