Reinforcement Learning (RL)

Chapter 1:

Overview of Reinforcement Learning

Saeed Saeedvand, Ph.D.

Course Topics in General

- ✓ Introduction to Reinforcement Learning and its applications
- ✓ Multi-armed bandits
- ✓ Markov Decision Processes (MDP)
- ✓ Value Iteration
- ✓ Policy Iteration
- ✓ Monte Carlo algorithms
- ✓ Temporal Difference (TD) evaluation
- ✓ Q-Learning
- ✓ SARSA
- ✓ ESARSA
- ✓ Introduction to DRL
- ✓ Policy Gradient
- ✓ Actor-Critic
- ✓ State-of-the-art Algorithms

Assessment

- ✓**Assignments and homework:** 30%
- ✓**Project/Research**: 30%.
- ✓**Midterm :** 15%
- ✓**Exam**: 25%

Slides and class materials: Moodle

Contact me: saeedvand@ntnu.edu.tw

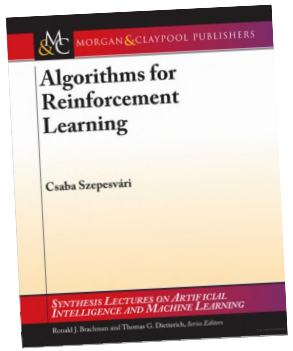
Textbooks

✓**Reinforcement Learning:** An Introduction

[Richard S. Sutton and Andrew G. Barto, Second edition]

http://www.incompleteideas.net/book/ebook/the-book.html

✓**Algorithms for Reinforcement Learning,** Csaba [Szepesvari, 2009]



Contents

Chapter 1:

- ✓ Introduction to Reinforcement Learning (RL)
- ✓ History of reinforcement learning
- ✓ Reinforcement Learning Differences with ML
- ✓ Some application examples
- ✓ Main sub elements of a RL
- ✓ Formal presentation of Fundamentals

Aim of this chapter:

- ✓ Understand concepts of a Reinforcement Learning (RL) and discuss important elements of RL.

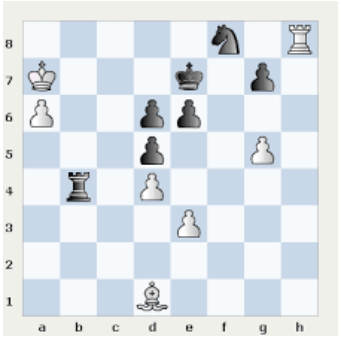
Reinforcement Learning

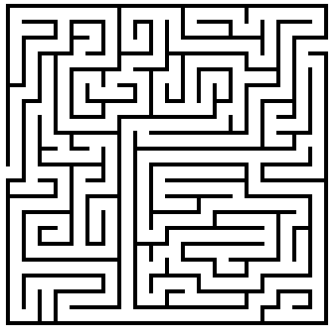
✓ RL is **the science of decision making**

✓ **Works** for all problems and games **where decision sequences are important**:

- Chess, maze, path planning, industrial robot arm, soccer robot, sweeper robots, etc.
- We may do not have complete model of environment and state transitions (RL will learn it)

✓ RL has tremendous popularity in the last decade

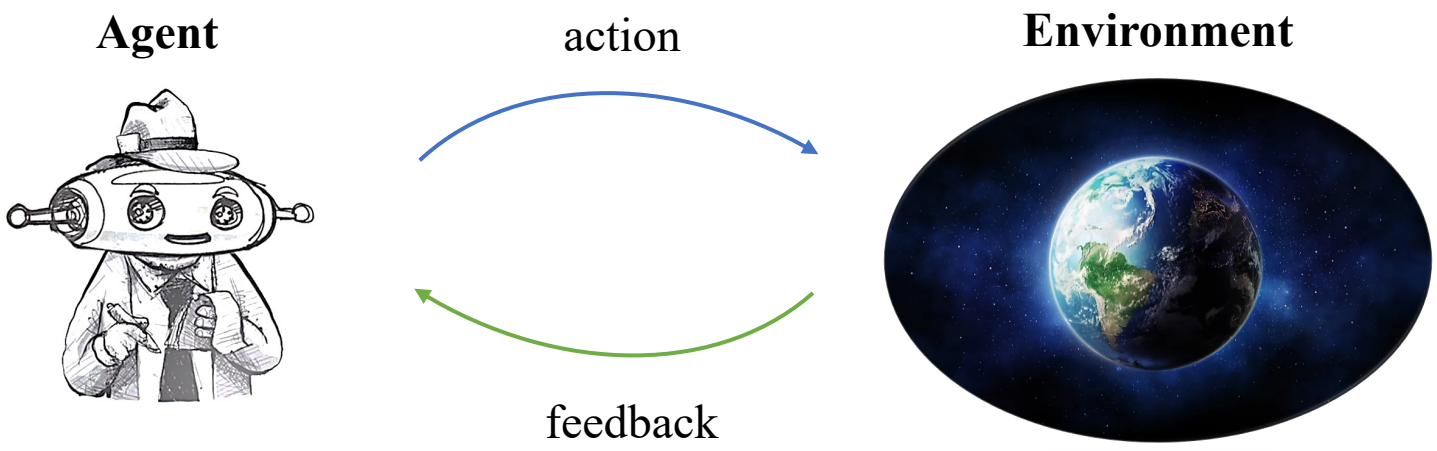




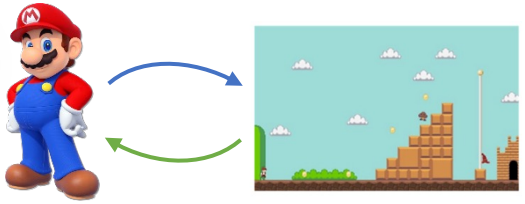
Agents

What is an Agent?

- ✓ Anything that can control its surroundings through sensors, understand and act in that environment.



Simulation



Reinforcement Learning

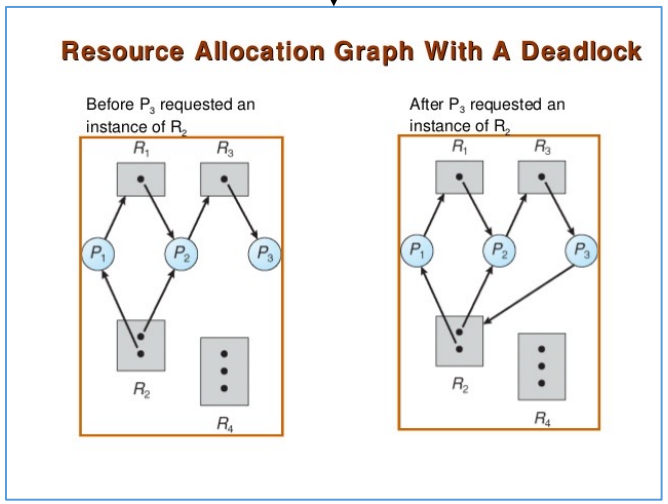
- ✓ The agent learns how to move from **initial state** to **Goal state** or (target state) and what is the best Action in each state.

With dynamic nature!

Reinforcement Learning

- ✓ **RL:** A type of learning that is very **useful** in robotics because the robot learns exactly **which action is best in each state**.

- ✓ **Applications:** From resource allocation in the operating system to unmanned helicopter control and ...



Reinforcement Learning (Example)

- ✓ **A practical Example**

Reinforcement learning

History of reinforcement learning

Three concurrent threads

1) **Learning by trial and error**:
   - ✓ Originated in the psychology of animal learning
   - ✓ Earliest work in AI resulted revealing of RL (early 1980s).

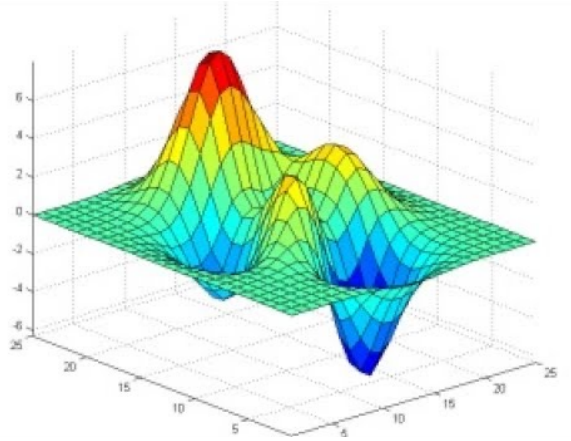
2) **The problem of optimal control** and its solution using **Dynamic Programming (DP)**
   - ✓ The research in 'optimal control' began in the 1950's

3) **Temporal Difference Learning Methods**
   - ✓ Inspired from mathematical differentiation
   - ✓ Aiming to derive a prediction from a set of known variables

Reinforcement learning

Optimal Control

- ✓ It is a **branch of mathematical optimization**
- ✓ It describe the problem of **designing a controller to minimize or maximize** a measure of a dynamical system's behavior over time
- ✓ It is to **optimize objective function** for chosen actions



What is the Dynamic Programming (DP)

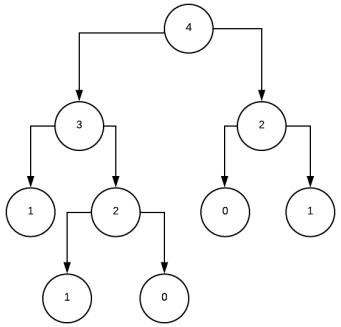
✓ The method was **developed by Richard Bellman** in the 1950s

a) **Solving a complex problem** by first breaking into a collection of simpler subproblems

b) Solving each subproblem just once

c) Then **storing** their solutions to **avoid repetitive computations (**as in recursive algorithm is happens but different).

Example:



Fibonacci

```
def Fibonacci(n):
    Fib = [None] * (n + 1)

    Fib[0], Fib[1] = 0, 1
    print(Fib)

    for i in range(2, n + 1):
        Fib[i] = Fib[i - 1] + Fib[i - 2]
    return Fib[n]

print("Result: ", Fibonacci(6))
```

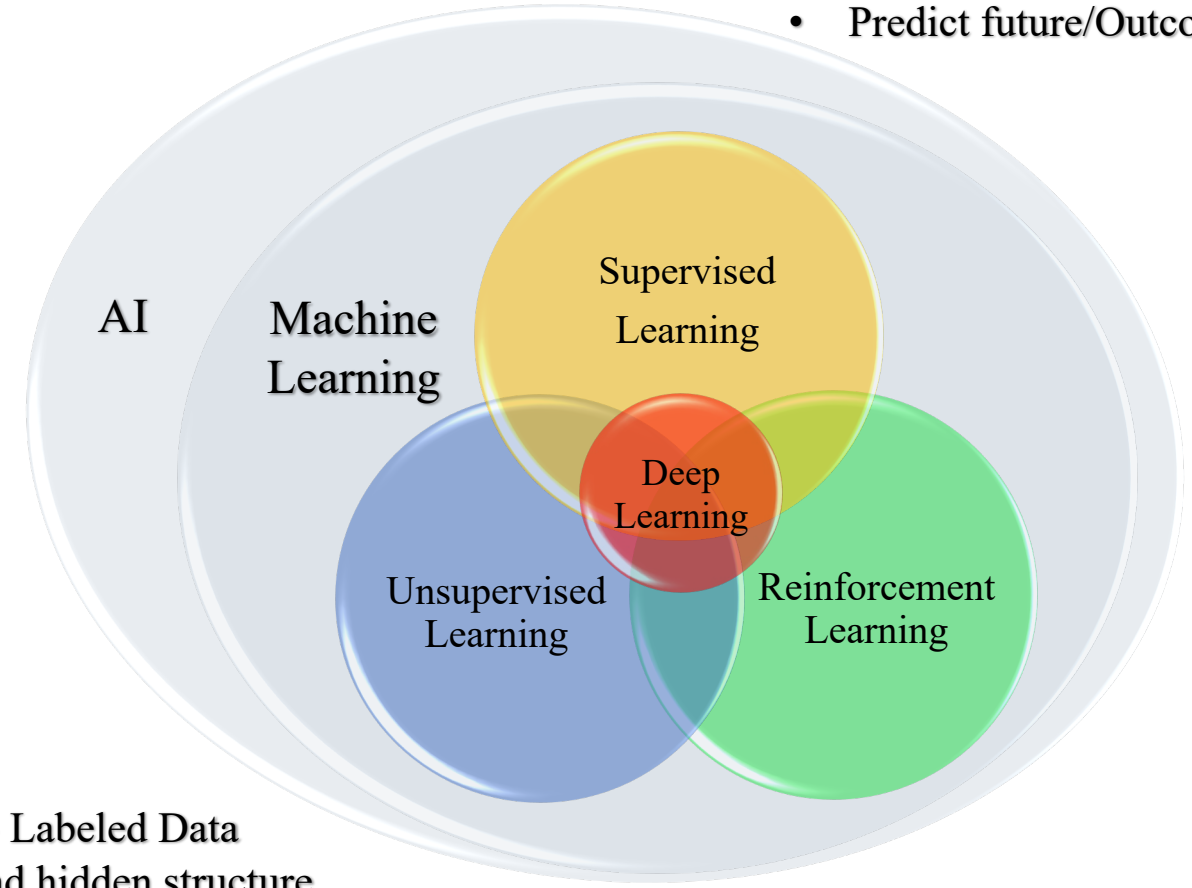
What is the Machine Learning

Machine Learning (ML)

ML is a type of AI that:

- Allows us to predict outcomes **without being explicitly programmed**.
- Building methods **that "learn" a model** based on the **sample data**

- Labeled Data
- Predict future/Outcome



- No Labeled Data
- Find hidden structure
- e.g. K-means clustering

- Learn in interaction mode
- Reward
- Decision process

Reinforcement Learning Differences with ML

Supervised learning:

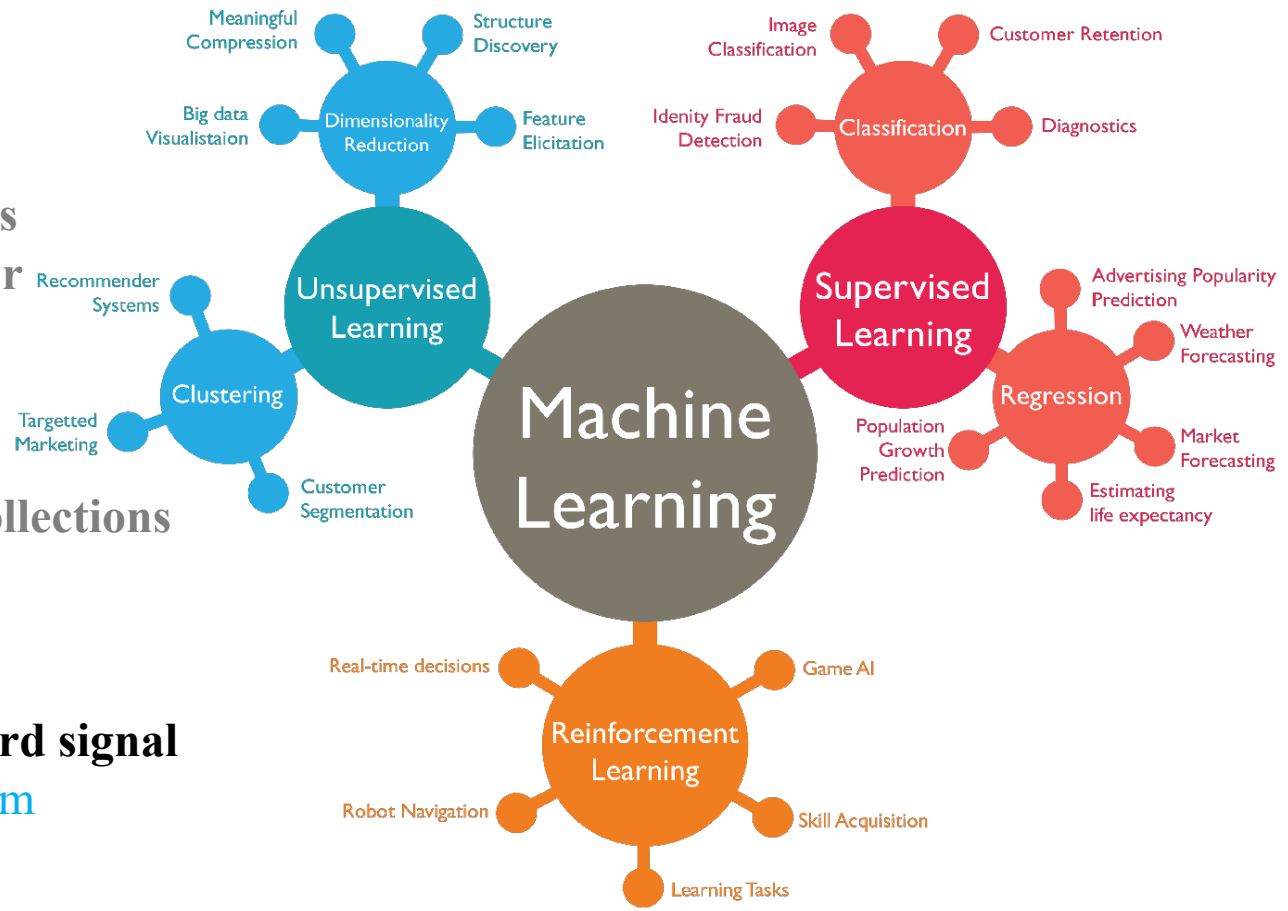
- ✓ Learning from a **training set of labeled examples** provided by a knowledgeable **external supervisor**

Unsupervised learning:

- ✓ Is typically about **finding structure hidden in collections of unlabeled data**.

Reinforcement learning

- ✓ Reinforcement learning is **trying to maximize a reward signal** instead of trying to find hidden structure (Learning from interaction)



We therefore consider reinforcement learning to be a **third** machine learning paradigm

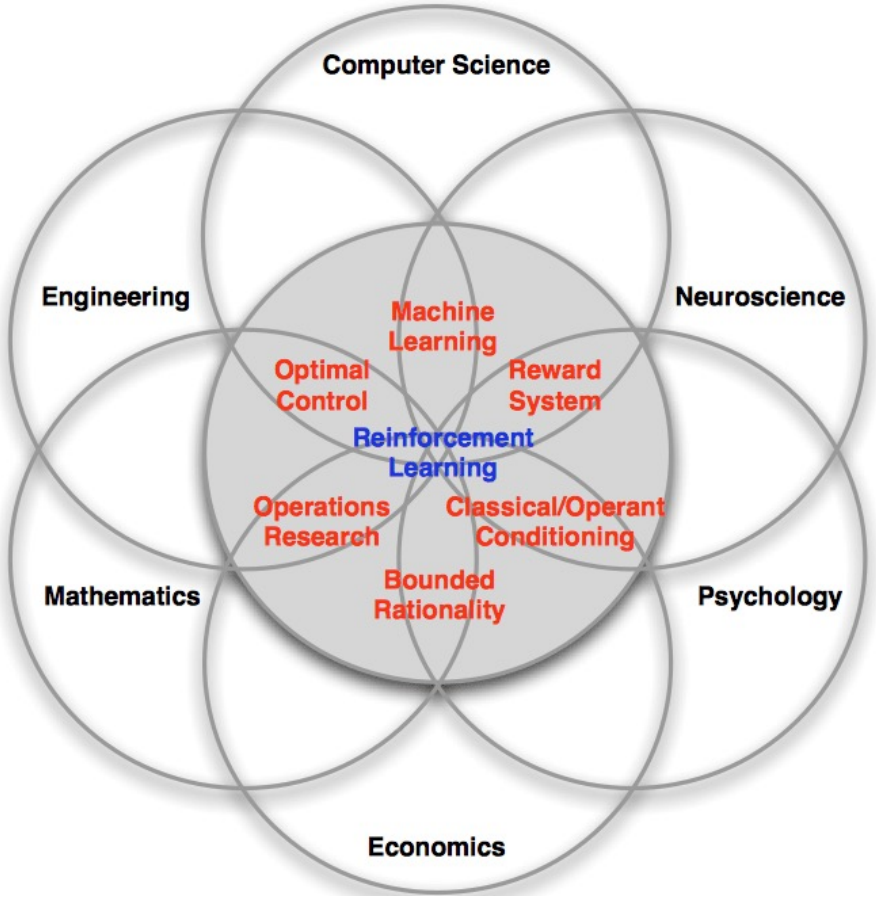
Reinforcement learning

Reinforcement learning main characteristics:

- ✓There is no expert supervision during training
- ✓Environment is usually stochastic
- ✓The model of environment can be incomplete
- ✓Delayed feedback or partial feedback is possible

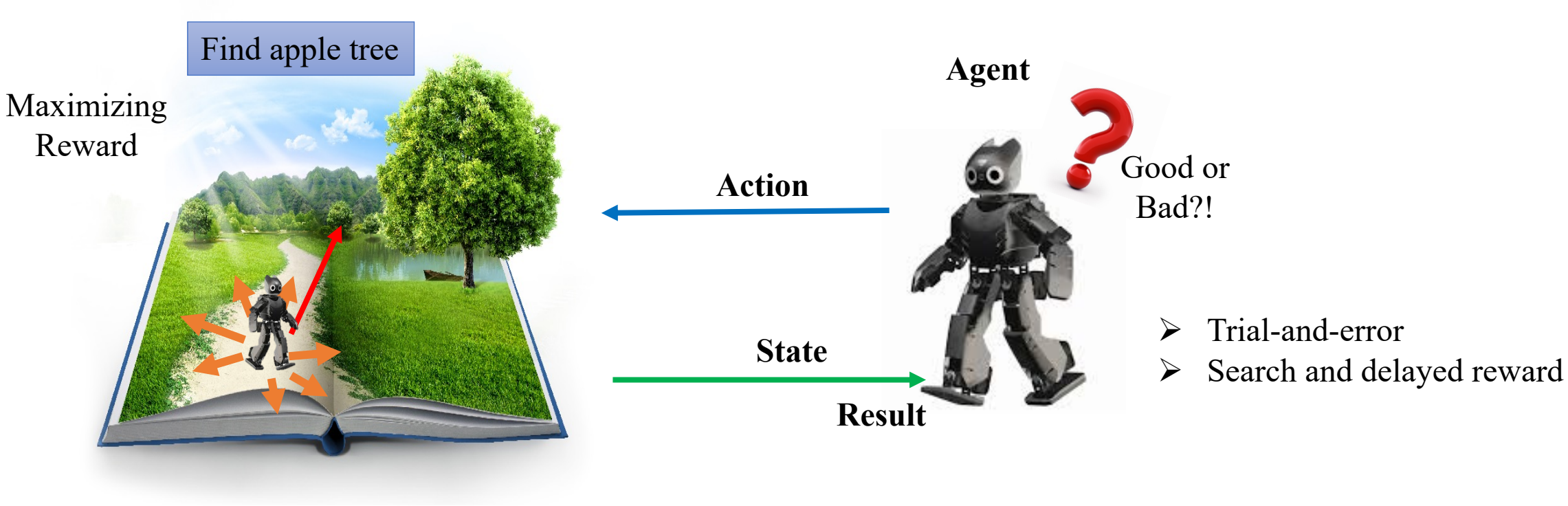
Reinforcement learning application domains

- ✓Computer Science
- ✓Engineering
- ✓Mathematics
- ✓Economics
- ✓Neuroscience
- ✓Psychology



Reinforcement Learning

✓ The **agent takes** the **knowledge** of the environment through **interaction**, and the agent performs an action in the environment.



- Trial-and-error
- Search and delayed reward

Reinforcement learning

Definition

- ✓ RL **explicitly considers** the whole problem of **a goal-directed agent** interacting with an **uncertain environment.**
- ✓ **In solving different** examples we **can involve explicit goals that agent can judge progress.**
- ✓ The agent can **use its experience** to improve its performance over time.
  - ✓ For instance the **chess player refines the intuition** to evaluate positions, thereby improving the move

Reinforcement learning

Definition

- ✓ RL prefers actions that it **has tried in the past** and **found to be effective** to get more **reward**
- ✓ So the agent must **try a variety of actions** and **favor** those that appear to be **best**.
- ✓ **RL uses** the formal framework of **Markov Decision Processes (MDP)** to define the interaction between a learning agent and its environment
  - states, actions, and rewards

A new challenges that arise **only in** RL is tradeoff between **exploration and exploitation**

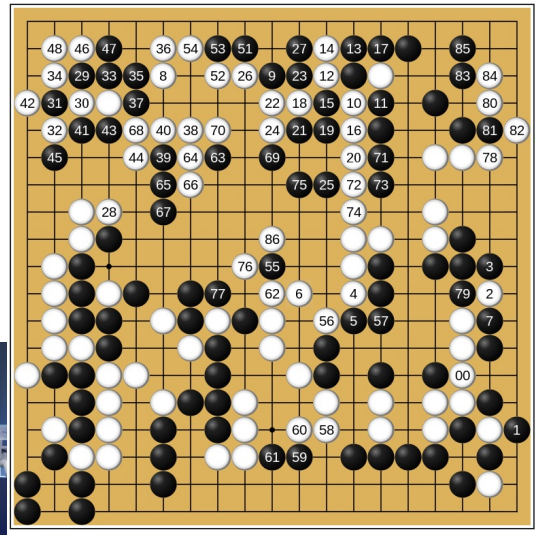
Reinforcement learning application examples

Game Playing

✓ **AlphaGo** and **AlphaGo Zero** are RL based algorithm that plays the **board game Go**, (DeepMind, Google).

In March 2016, RL beat Lee Sedol in a five-game match.

Environment?	Board and player
Agent?	Player
State?	Board Configuration
Action?	Placing a stone
Reward?	✓ Wining or Losing ✓ Current evaluation



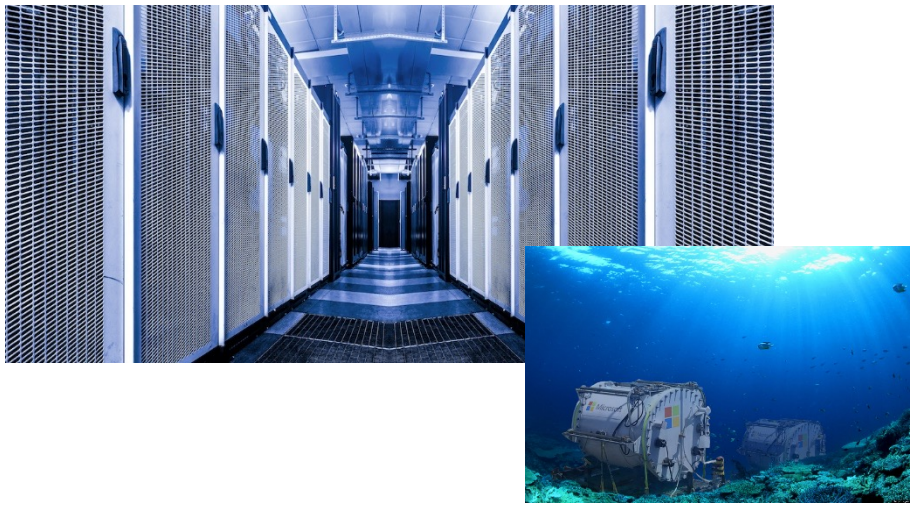
https://www.youtube.com/watch?v=WXuK6gekU1Y

Reinforcement learning application examples

Datacenters Cooling

- ✓ On of the **world's most challenging physical problems is energy consumption**
- ✓ Data centers (DCs) have a **large energy consumption** to **keep the servers running**
- ✓ **Google** data centers **using machine learning algorithms** have reduced the amount of energy for cooling by up to **40 percent** (**DeepMind)**

Environment?	All datacenters (DC)
Agent?	DC controller
State?	Energy consumption at time t
Action?	Move load, turn on, turn off
Reward?	Using less energy



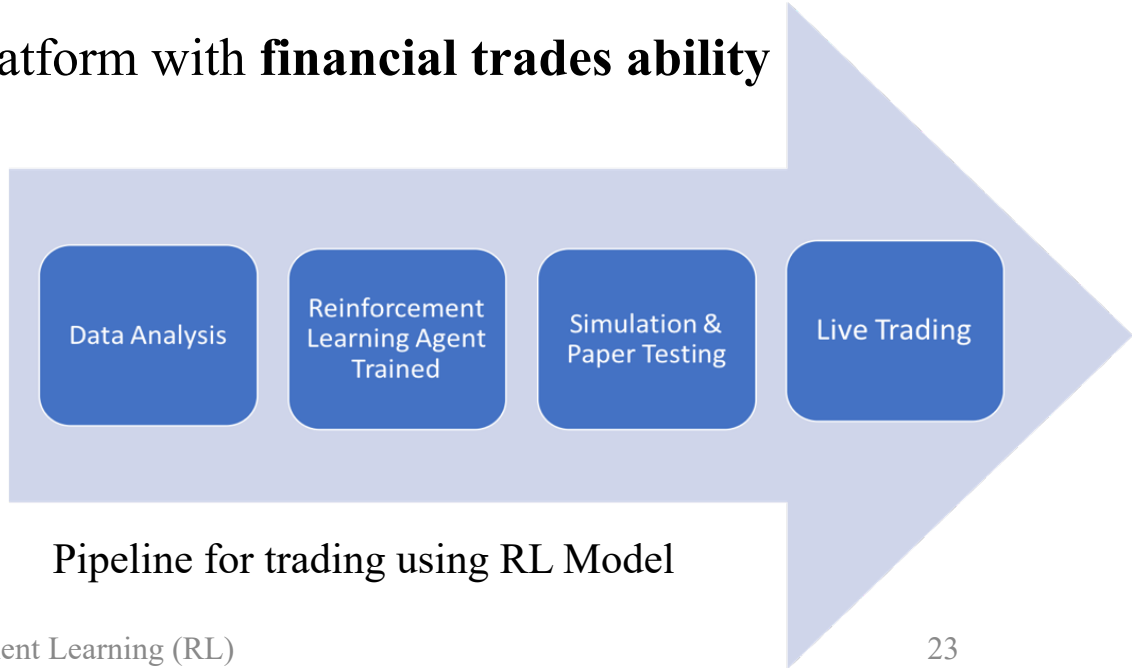
https://www.youtube.com/watch?v=lBeepqQBpvU

Reinforcement learning application examples

Trading and finance

- ✓ Supervised learning can be used for **predicting future sales stock prices but don't determine the action to take at a particular stock price**.
- ✓ RL agent **can decide on such a task**; whether to hold, buy, or sell.
- ✓ For example **IBM has a sophisticated** RL based platform with **financial trades ability**

Environment?	All traders and all Shares
Agent?	Trader
State?	A moment of the shares status
Action?	Sell, Buy, hold
Reward?	Profit



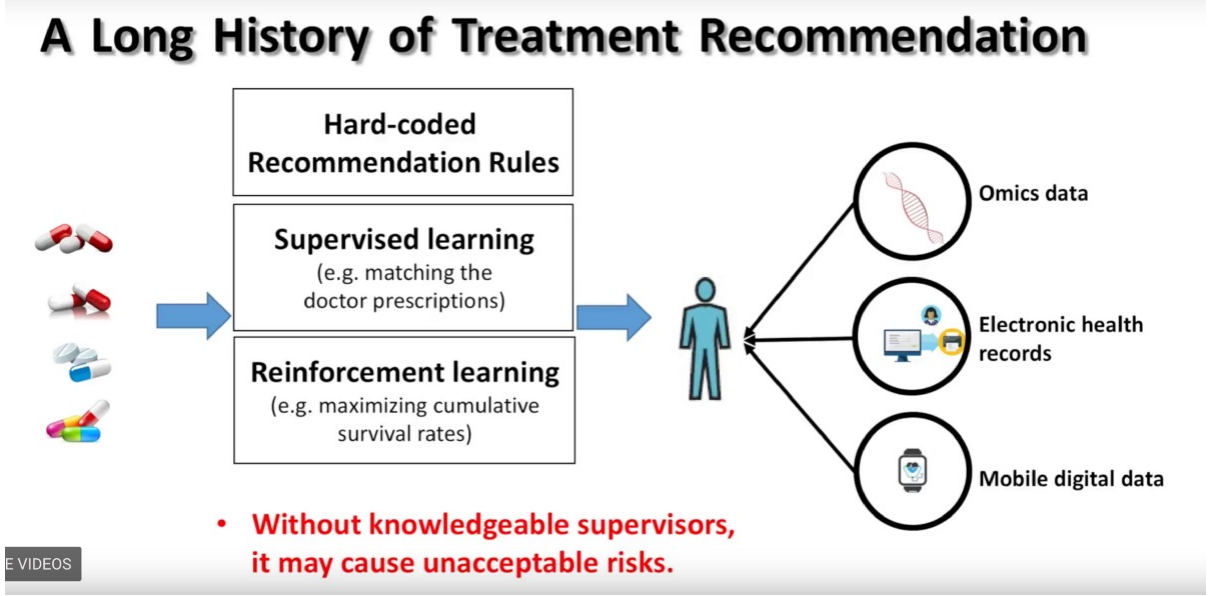
Pipeline for trading using RL Model

Reinforcement learning application examples

Healthcare

- ✓ **Healthcare** has seen a **significant tilt towards RL** in the past few years
- ✓ Implementing **dynamic treatment regimes** (**DTRs**) for patients suffering from **long-term illnesses**.
- ✓ **Choosing medicines is hard**.

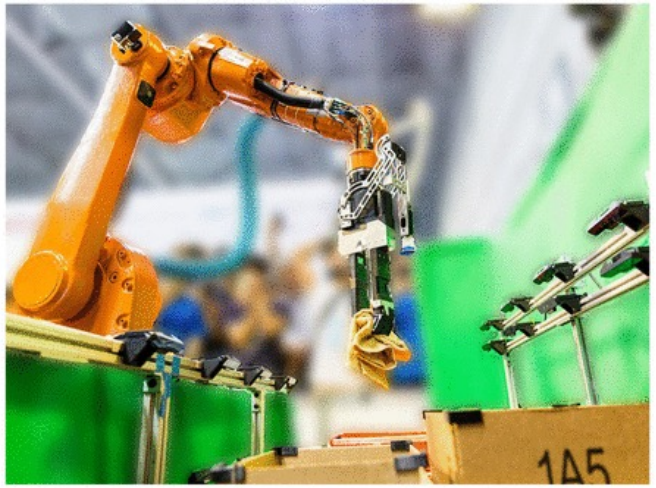
Environment?	Patients and all treatments
Agent?	Recommender
State?	Patient's current health status
Action?	Advising medicine
Reward?	Health improvement

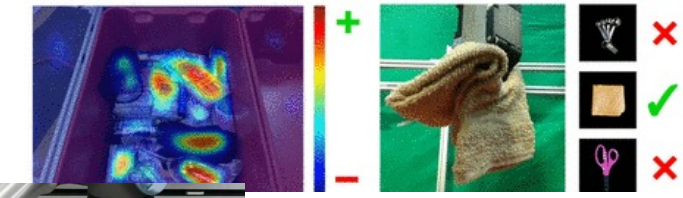


Reinforcement learning application examples

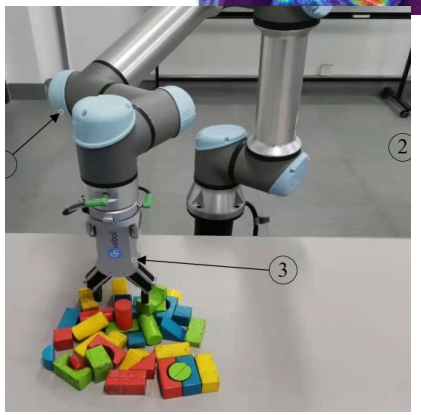
Robotics

- ✓ The robot to have ability to grasp various objects
- ✓ Almost in **all kind of robots we can use RL**





Environment?	Objects and robot's work space
Agent?	Robot
State?	Robot Joint's position and Object Pose
Action?	Robot's arm movements
Reward?	?



Reinforcement learning

✓Reinforcement learning is for learning:

- **What to do**
- **How to map each state to actions** (long and short term reward)

✓ The learner is not told which actions to take, but instead **must discover** which **actions** yield the **most reward** by **trying them**.

✓ Actions may **affect not only the immediate reward** but also the **next situations** and all **subsequent rewards**.

Reinforcement learning

Main sub elements of a RL

1) A policy

2) Reward signal

3) Value function

4) Model of the environment

Reinforcement learning

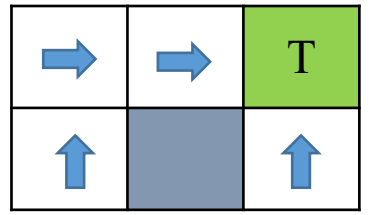
1 Policy

✓The policy (π) is the core of a RL agent which determines the behavior at a given time

✓So policy is a **mapping** the states of the environment **to actions**

What to do in each state

Environment



Reinforcement learning

1 Policy

- ✓In some cases the policy may be a **simple function** or **lookup table**
- ✓Policy **may involve extensive computation** such as a **search process**
- ✓Policies may be **stochastic**, specifying **probabilities** for each action

Important (modeling the error)

2 Reward

- ✓ A reward **signal defines the goal** of a reinforcement learning problem
- ✓ The reward signal **defines the good and bad events** for the agent
- ✓ The agent's **objective** is to **maximize the total reward** it receives over the long run.

3 Value Function

- ✓ The **value of a state** is the **total amount of reward** an agent can expect to accumulate in the future, starting from that state (**predictions of rewards)**.
- ✓ Values indicate the **long-term desirability** of states after taking into account the states
- ✓ **Reward signal only indicates immediate sense**
- ✓ **Value function specifies what is good in the long run**

Reinforcement learning

3 Value Function

✓ **A state** maybe always **have a low immediate reward** but still have a **high value**.

How it is possible?

Because it may followed by high rewarded states

✓ **Without rewards** there could be **no values**, and the only purpose of estimating values is to achieve more reward.

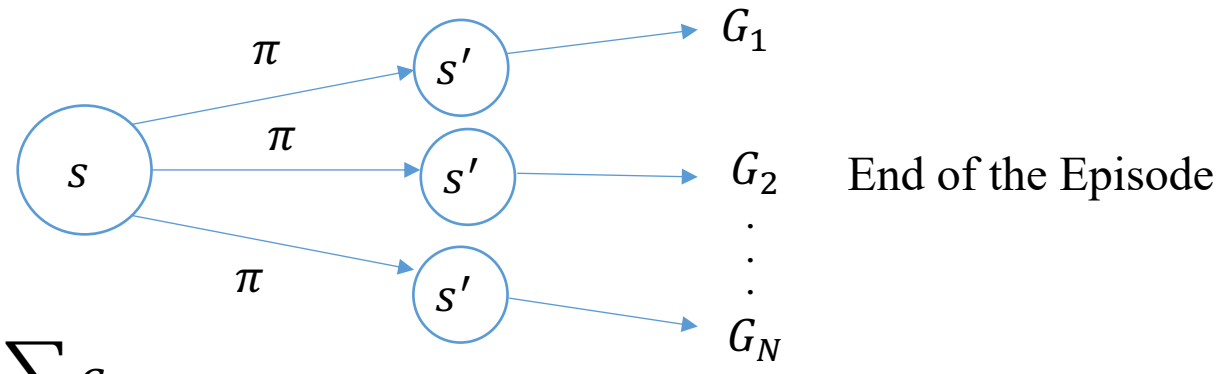
3 Value Function

✓ Better to choose actions that **bring highest value**, **not** **highest reward**

✓ **Much harder to determine values**

- Since they are estimation from the sequences of observations an agent makes over its entire lifetime (episode).

From state s run policy π multiple times (stochastic environment can cause to be in different next states s')



Average ⇒ $V_\pi(s) = \frac{1}{N}\sum G_i$

4 Model of the Environment

- ✓ It is to **mimics** the **behavior** of the environment, (how the environment will behave)
- ✓ By given a state and action, the **model** might **predict** the resultant next state and next reward.
- ✓ Models are **used for planning**, by which **considering possible future situations** **before they are actually experienced**.

4 Model of the Environment

- ✓ If a method for solving RL problems use models it is called **model-based** methods
- ✓ On the other hand, there are simpler **model-free** methods that are explicitly **trial-and-error** learners.
- ✓ Modern RL **spans the spectrum** from **low-level, trial-and-error** learning to **high-level**, deliberative planning

Reinforcement learning

Formal presentation of Fundamentals

- ✓ What is the **state** (s)?
  - ✓ Current state (s_t), next state (s_{t+1})

What is the **state space**?

- ✓ What is the **action** (a)?
  - ➢ An action that agent performs and moves from state s_t to s_{t+1}
- ✓ What is the **reward** (r or $\mathrm{R}(s, a)$)?
  - ➢ The result of taking action a at state s

Actions may **affect not only the immediate reward** but also the **next situations** and all **subsequent rewards**

Formal presentation of Fundamentals

- ✓ What is the **Episode?**
  - ➢ Performing whole sequence of state and action until reaching the terminate state
- ✓ What is the **Transition probability** $P(s'|s, a)$?
  - ➢ The probability of reaching s' if taking action at a_t state s_t
- ✓ What is the **Policy π(s, a)**?
  - ➢ Policy maps each state to an action (how the agent acts at each state).
  - ➢ It can be either **deterministic** or **stochastic**

$$a = \pi(s) \qquad \pi(a|s) = P[A_t = a \mid S_t = s]$$

Formal presentation of Fundamentals

✓ What is the **Return** (G_t)?

➢ It is the total future rewards at state s_t

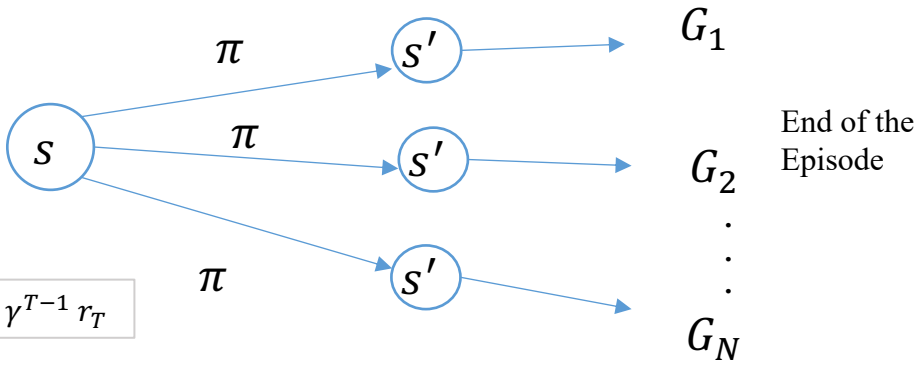
T is the end of the episode

$$G_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{T-1} r_T$$

Gamma (γ) is discounted future reward

Discounted reward by (γ^n) **describe the importance of the current state** (future rewards get lower effect).

Formal presentation of Fundamentals



Return:

$$G_t = r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{T-1} r_T$$

✓ What is the **Value V(s)**?

- **Expected return** for starting from state s or prediction of future reward for a state (in cases of **multi run** e.g. can be average)

Also known as:
State-Value Function

$$V(s) = \mathbb{E}[G_t | s_t = s] =$$
$$\mathbb{E}[r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \cdots + \gamma^{T-1} r_T | s_t = s]$$

Immediate reward

Discounted value of successor state

v(s) gives the long-term value of state s

Formal presentation of Fundamentals

Also known as:
Action-Value Function
Or even expected utility

✓ What is the **Q-value Q(s, a)**?

➢ It is the expected return for starting from **state *s* and taking action *a***

$$Q(s,a) = \mathbb{E}[G_t | s_t = s, a_t = a] = \mathbb{E}[r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \cdots | s_t = s, a_t = a]$$

$$Q(s,a) = \mathbb{E}[G_t] = \frac{1}{N}\sum_{n=1}^{N} G_t^n$$

Average of running different episodes

Formal presentation of Fundamentals

- ✓ What is the **Optimal Policy** $\boldsymbol{\pi^*(s)}$?
  - ➢ The best possible solution (Policy) for that state
- ✓ What is the **Optimal Value** $\mathbf{V^*(s)} = \boldsymbol{V_{\pi^*}}\ \boldsymbol{(s)}$?
  - ➢ The best possible/expected value for that states
- ✓ What is the **Optimal Q-Value** $\mathbf{Q^*(s, a)}$?
  - ➢ The best possible/expected Q-value for that states

Formal presentation of Fundamentals

✓ Definition of the **Optimal Value Functions**?

- **Optimal state-value function:** Maximum value function over all policies

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

- **Optimal action-value function:** Maximum value function over all policies

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

Reinforcement learning

Two fundamental tasks of reinforcement learning

Prediction Task

- ✓ We have a policy:
  - Goal is to **evaluate policy** by estimating the state-value or Q-value of running actions of given policy.

Evaluate the future

Control Task

- ✓ we don't know the policy, and the goal is:
  - To **find the optimal policy** aiming to collect maximum rewards

Optimize the future

Tabular Solution Methods

- ✓ The **core ideas** of RL algorithms in their **simplest forms**:
  - ➢ The state and action spaces are **small enough** for the **approximate value functions** to be represented as **arrays, or tables**.

- ✓ In this case, the methods can often find exact solutions (**optimal value function** and the **optimal policy**).

Tabular Solution Methods

- Fundamental classes of methods for solving finite Markov Decision Problems (MDPs):
  - ✓ **Dynamic programming**
    - **Methods** are well developed mathematically, but require a complete and accurate model of the environment.
  - ✓ **Monte Carlo methods**
    - Don't require a model and are conceptually simple, but are not well suited for step-by-step incremental computation.
  - ✓ **Temporal difference learning**
    - Require no model and are fully incremental, but are more complex to analyze. The methods also differ in several ways with respect to their efficiency and speed of convergence.

Each class of methods has its strengths and weaknesses

Summery

- ✓ We discussed what is the RL
- ✓ Presented State, Action, Value, Model
- ✓ Saw some application examples
- ✓ Formulated the fundamentals
- ✓ We introduced Two fundamental tasks of reinforcement learning
- ✓ Discussed tabular Solution Methods