UNIT 1 INTRODUCTION TO REINFORCEMENT LEARNING

Dr. Soni Sweta

Syllabus

Unit	Description	Duration
1.	**Introduction to RL** Introduction to RL terminology, Elements of RL, RL framework and applications, Immediate RL	03

Course Outcomes

After completion of the course, students will be able to –

1. Apply the basics of Reinforcement Learning (RL) to compare with traditional control design
2. Correlate how RL relates and fits into the broader umbrella of machine learning, deep learning
3. Recommend value functions and appropriate algorithms for optimal decision-making
4. Design a dynamic programming approach to an industrial control problem

References

Text Books

1. Laura Graesser and Wah Loon Keng, *Foundations of Deep Reinforcement Learning: Theory and Practice in Python*, 1st Edition, Pearson India/ Padmavati Publisher, 2022.
2. Richard Sutton and Andrew Barto, *Reinforcement Learning: An Introduction*, 2nd Edition, MIT Press, 2018.
3. Abhishek Nandy and Manisha Biswas, *Reinforcement Learning: With Open AI, TensorFlow and Keras Using Python*, 1st Edition, Apress Publisher, 2017.

Reference Books

1. Nimish Sanghi, *Deep Reinforcement Learning with Python: With PyTorch, TensorFlow and OpenAI*, 2nd edition, Apress Publisher, 2021.
2. Alexander Zai and Brandon Brown, *Deep Reinforcement Learning in Action*, 1st Edition, Manning Publisher, 2020.
3. Csaba Szepesvari, *Algorithms for Reinforcement Learning*, 3rd Edition, Morgan & Claypool Publisher, 2019.

- Richard Sutton, Andrew Barto, “Reinforcement Learning: An Introduction”, 2nd edition, MIT Press, 2018
- Abhishek Nandy, Manisha Biswas, “Reinforcement Learning with OpenAI, TensorFlow and Keras Using Python
- Csaba Szepesvari , “Algorithms for RL”, Morgan and Claypool Publisher, 2010
- http://ndl.iitkgp.ac.in/he_document/nptel/courses_106_106_1061 06143_video_lec5
- https://www.youtube.com/watch?v=IRm6ma9NIhQ&list=PLZ_sI4f41TGvthD8dA7daahlbLV0yDW0w&index=1

Overall Concepts in RL

- Introduction, Applications
- Multi-arm Bandit problem
- Markov Decision Process
- Value Iteration
- Policy Iteration
- Monte Carlo Algorithms
- Temporal Difference
- Q Learning
- SARSA
- Policy Gradient methods
- Actor-Critic method

How do we (the humans) actually learn?

- Is it
  - Unsupervised
  - Supervised
  - Reinforcement
  - All of the above

Can you identify type of learning

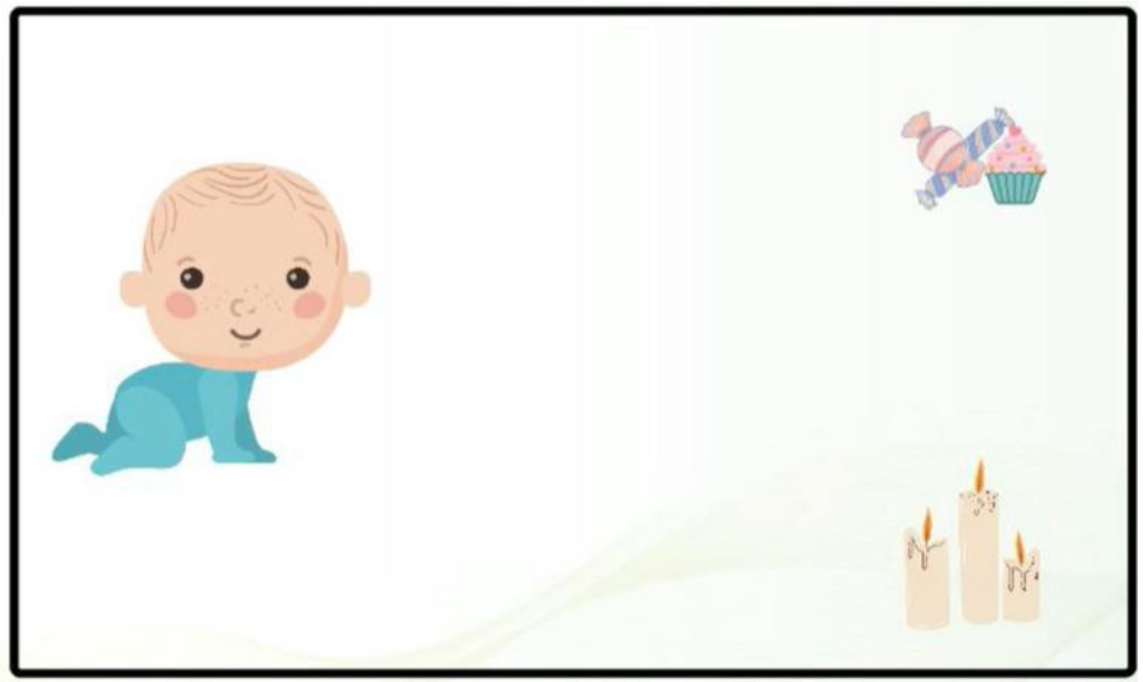
- Scenario
  - Child points to an object , say apple
  - Parents calls it as apple

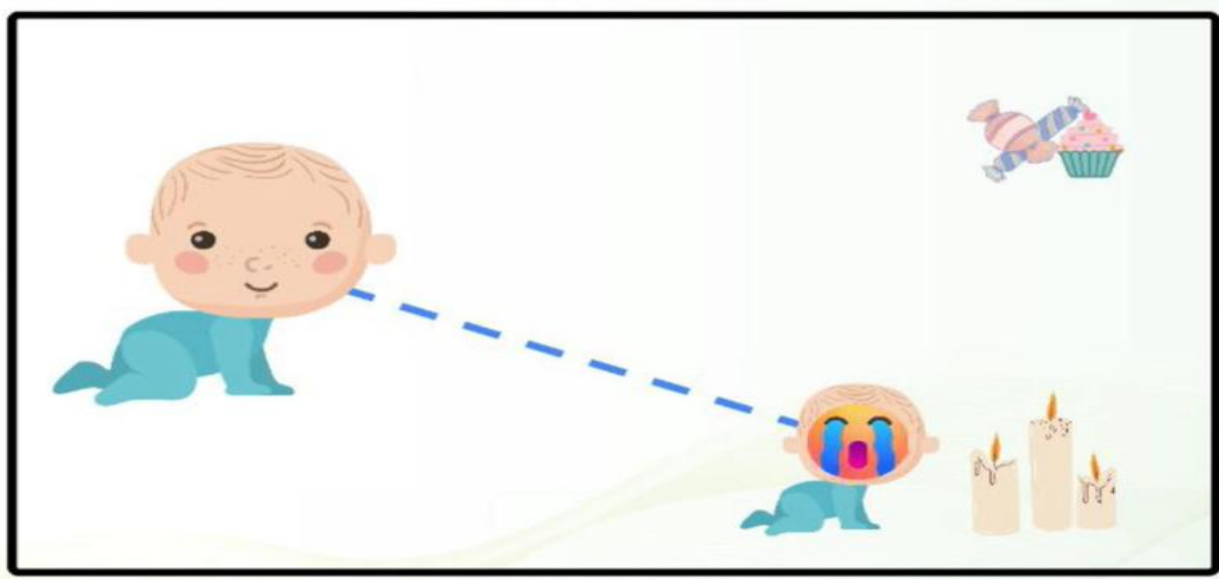
- **Supervised**

Can you identify type of learning

- Scenario
  - Child listens to conversation happening around it
  - Child listens and tries to analyze the recurring pattern, sentence flow, context, tone etc
  - Child then picks up the language and attempts to make its own sentences

- **Unsupervised**





- Familiar models of machine learning
  - Supervised: Classification, Regression, etc.
  - Unsupervised: Clustering, Frequent patterns, etc.

- How did you learn to cycle?

- A trial-and-error learning paradigm
- Learn about a system through interaction
- Inspired by behavioural psychology!
  - Pavlov's dog

CONDITIONING

Pavlov's Dog Experiment

BEFORE CONDITIONING

Unconditioned stimulus

Unconditioned response

Neutral stimulus

No response

DURING CONDITIONING

Food + Bell

Unconditioned response

AFTER CONDITIONING

Conditioned stimulus

Conditioned response

- The Pavlov dog experiment is a series of experiments conducted by Russian physiologist Ivan Pavlov in the 1890s that demonstrated classical conditioning:

  - **Discovery**

    Pavlov noticed that his dogs would salivate when he entered the room, even when he wasn't bringing them food.

  - **Experimentation**

    Pavlov paired a neutral stimulus, like a metronome, with a positive stimulus, like food. After repeated pairings, the dogs learned to associate the metronome with food and would salivate when they heard it.

  - **Theory**

    Pavlov's experiments demonstrated that behaviors can be learned by associating two unrelated stimuli. This is known as classical conditioning, and the response to the neutral stimulus is called a conditioned response.

  - **Significance**

    Pavlov's work was the first systematic study of learning and conditioning. His findings inspired other scientists to investigate similar behaviors in humans.

- Pavlov Dog experiment :Predicting the output of bell which is food and the dog reacting appropriately like salivate. In RL agent always try to predict the outcomes like if we do this move we will get reward. Outcome here is amount of reward and punishment.

Example of RL

Teach dog new tricks

How Reinforcement Learning works in a broader sense:

- Your dog is an "agent" that is exposed to the **environment**. The environment could in your house, with you.
- The situations they encounter are analogous to a **state**. An example of a state could be your dog standing and you use a specific word in a certain tone in your living room
- Our agents react by performing an **action** to transition from one "state" to another "state," your dog goes from standing to sitting, for example.
- After the transition, they may receive a **reward** or **penalty** in return. You give them a treat! Or a "No" as a penalty.
- The **policy** is the strategy of choosing an action given a state in expectation of better outcomes.

Can you identify type of learning

- Scenario
  - Toddler tries to crawl, tries to stand, then take few steps
  - If it manages to stand for few moments, it feels good about itself and gets applaud from parents
  - If it falls, it might feel discouraged
  - Eventually it learns to walk without any explicit instructions
- **Reinforcement**

History of reinforcement learning

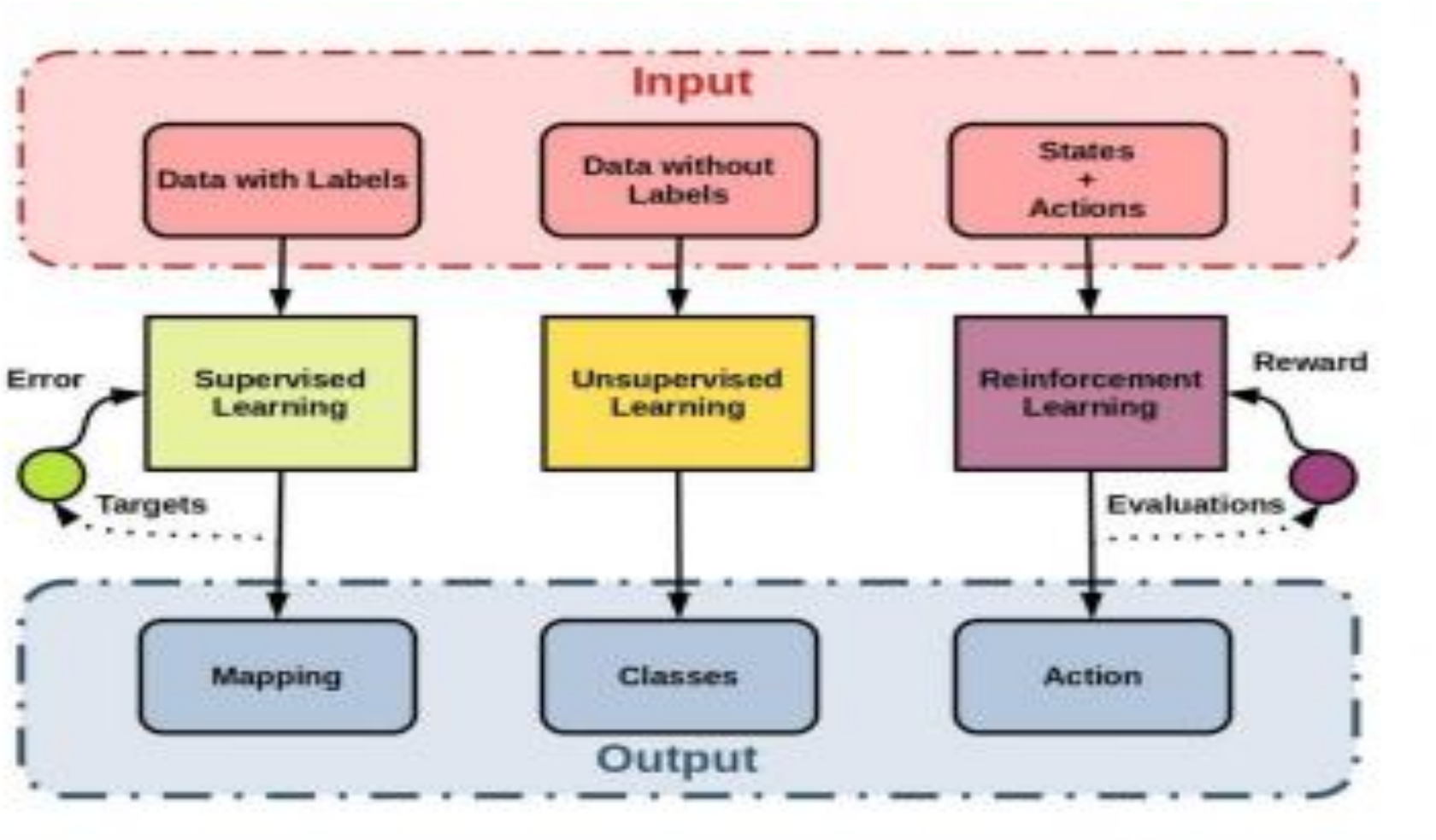
Three concurrent threads

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

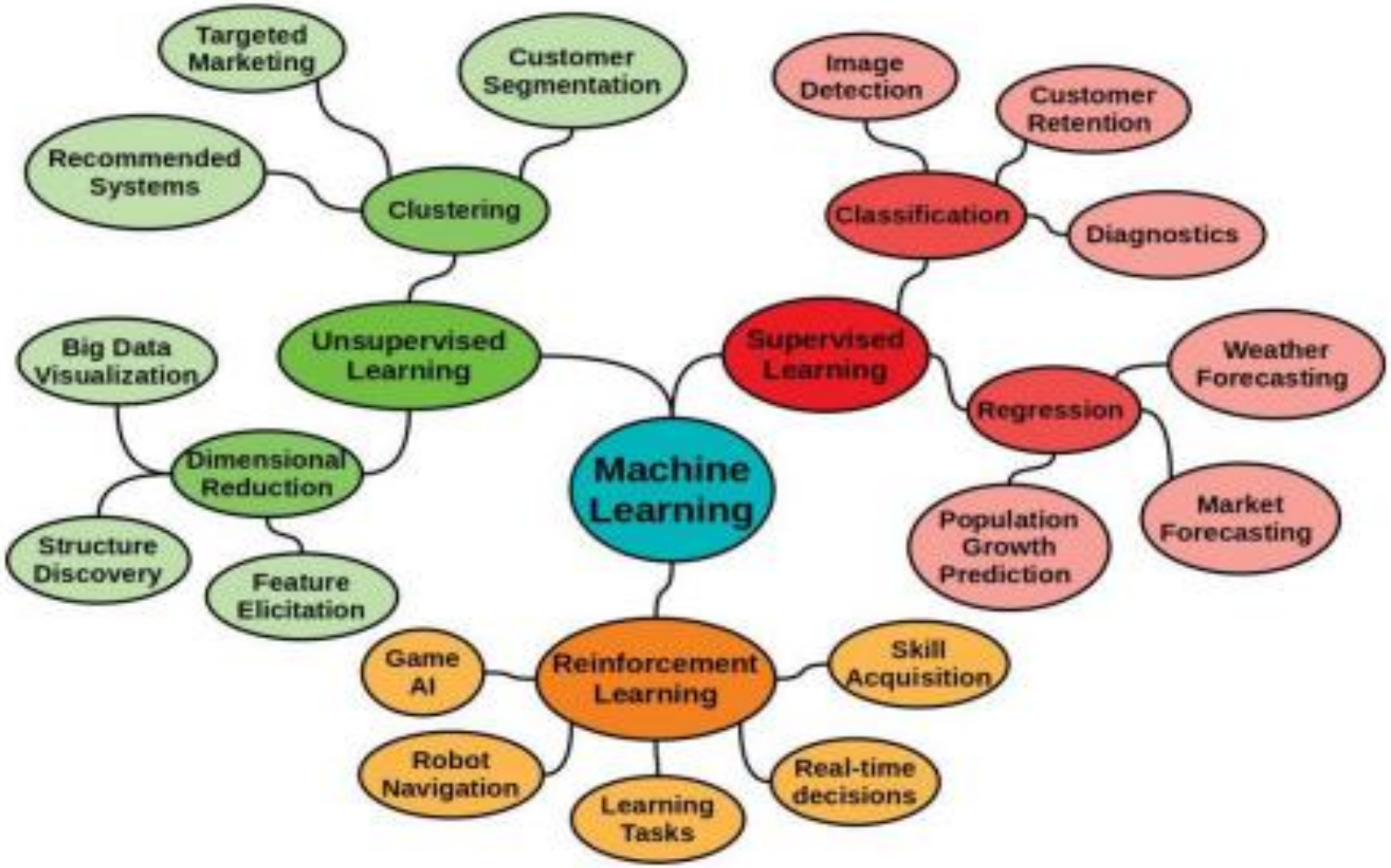
Supervised, Unsupervised and Reinforcement Machine Learning



Ref: https://starship-knowledge.com/supervised-vs-unsupervised-vs-reinforcement

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

Applications of RL, Supervised ML and Unsupervised ML



Ref: https://starship-knowledge.com/supervised-vs-unsupervised-vs-reinforcement

Reinforcement Learning(RL) Problem

- It is learning
  - what to do
  - how to map situations to actions
  - so as to maximize a numerical reward signal
- 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 situation and all subsequent rewards
- Distinguishing features
  - **Trial and Error**
  - **Immediate or Delayed reward**
- RL is an art and science of decision making
- It works for all the problems where sequence of decisions are important
  - Example-Chess, Maze, Industrial Robot Arm, Path Planning, Sweeper Robot

Introduction To Reinforcement Learning

Roots of Reinforcement Learning

- Reinforcement learning (RL) is inspired by how humans and animals learn through interactions.
- Reward and Punishment: Just like we learn to repeat actions that give us rewards (like getting a treat) and avoid actions that lead to punishment (like getting scolded).
- Trial and Error: Similar to how we try different methods to solve a puzzle until we find the right one.
- Learning over Time: Just as we get better at a game by playing it repeatedly.
- Rewards from the sequence of actions

Reinforcement Learning Definition

- **Reinforcement learning (RL)** is a branch of machine learning where the learning of the Agent occurs via interacting with an Environment.
- It is a feedback based learning.

Source: Google Images

Introduction- Reinforcement Learning

- **Reinforcement Learning (RL) is the science of decision making. It is about learning the optimal behavior in an environment to obtain maximum reward.**

- In RL, the data is accumulated from machine learning systems that use a trial-and-error method

➢ **How does RL differ from supervised learning?**

Reinforcement learning differs from supervised learning in a way that-

Supervised-

❑In supervised learning, the training data has the answer key with it so the model is trained with the correct answer itself.

❑In supervised learning **the decisions are independent of each other** so labels are given to each decision.

Reinforcement-

○ In reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

❑In Reinforcement **learning decision is dependent**, So we give labels to sequences of dependent decisions

- Reinforcement learning is an autonomous, self-teaching system that essentially learns by trial and error. It performs actions with the aim of maximizing rewards, or in other words, **it is learning by doing in order to achieve the best outcomes.**

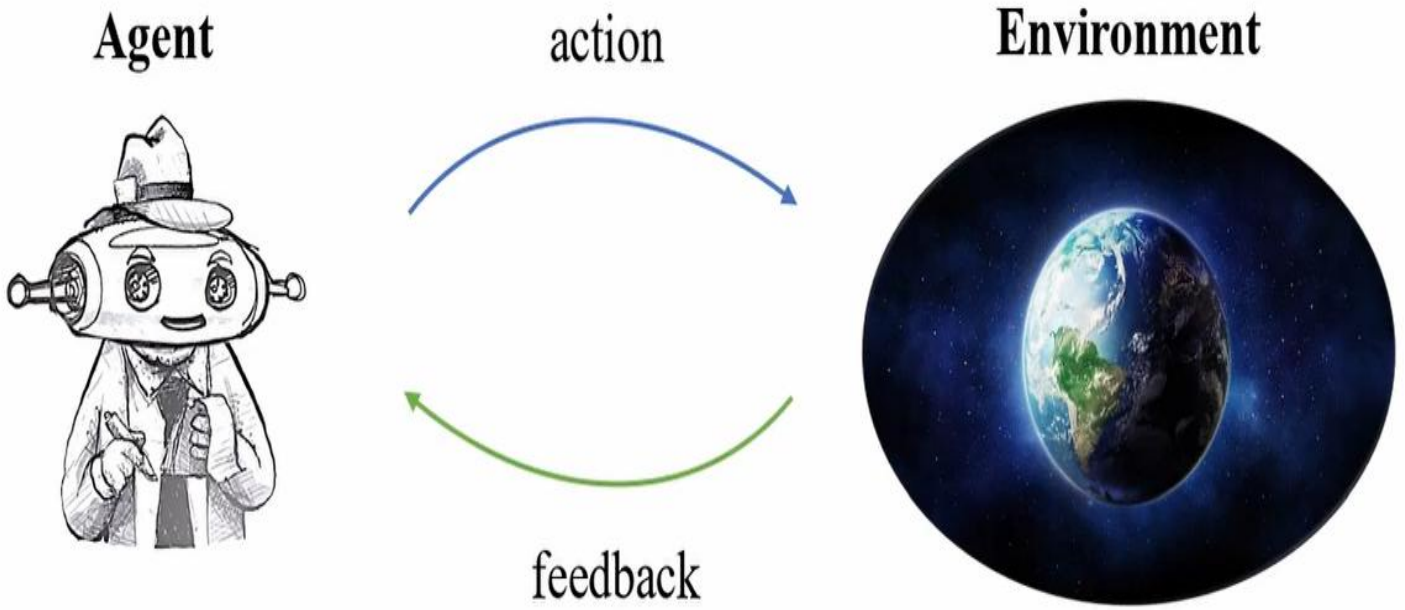
Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty

1.1 Reinforcement Learning

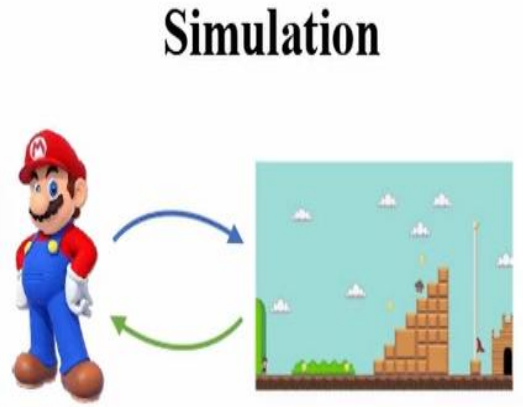
Reinforcement learning is learning what to do—how to map situations to actions—so as to maximize a numerical reward signal. The learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics—trial-and-error search and delayed reward—are the two most important distinguishing features of reinforcement learning.

Agent???

An Agent is anything that can control its surroundings through sensors, understand and act in the environment. eg-A small programming computer , Robot, Human etc



Environment can be physical or simulation eg game simulation environment



- The agent learns from how to move from initial state to goal state (target state)
- Agent always ask question to himself what is the best action in each independent state

Advantages of Reinforcement Learning

Some of the advantages of reinforcement learning are –

•Reinforcement learning doesn't require pre-defined instructions and human intervention.

•Reinforcement learning model can adapt to wide range of environments including static and dynamic.

•Reinforcement learning can be used to solve wide range of problems, including decision making, prediction and optimization.

•Reinforcement learning model gets better as it gains experience and fine-tunes.

Disadvantages of Reinforcement Learning

Some of the disadvantages of reinforcement learning are –

•Reinforcement learning depends on the quality of the reward function, if it is poorly designed, the model can never get better with its performance.

•The designing and tuning of reinforcement learning can be complex and requires expertise.

Elements of Reinforcement Learning

Reinforcement learning elements are as follows:

- Policy
- Reward function
- Value function
- Model of the environment

History of reinforcement learning

Three concurrent threads

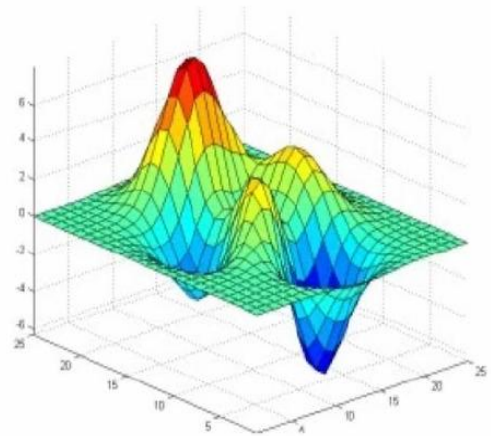
1) **Learning by trial and error**:
   - ✓ Originated in the psychology of animal learning
   - ✓ Earliest work in AI resulted revealing of RL (early 1980s).

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

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



When to apply Reinforcement Learning?

Reinforcement Learning is most suitable when:

- The problem environment is complex and uncertain, and traditional programming methods ineffective
- Feedback is sparse, delayed, and dependent on multiple decisions
- Decision-making (actions) follow a feedback loop

Why Is Reinforcement Learning Difficult?

The toughest parts of Reinforcement Learning are:

- To Map the Environment.
- To include All Possible Actions.

Reinforcement Learning

- It is Goal-oriented Learning.
- The Agent learns from the Consequence of its Actions.
- It is one of the most Active areas of Research in **Artificial Intelligence** (**AI**).

RLAlgorithm Steps

1. First, the agent observ the environment and interacts with the environment by performing an action.
2. The agent performs an action and moves from one state to another.
3. And then the agent will receive a reward based on the action it performed.
4. Based on the reward, the agent will learn whether the action was good or bad.
5. The agent will try to maximize the reward.

Learning and Planning

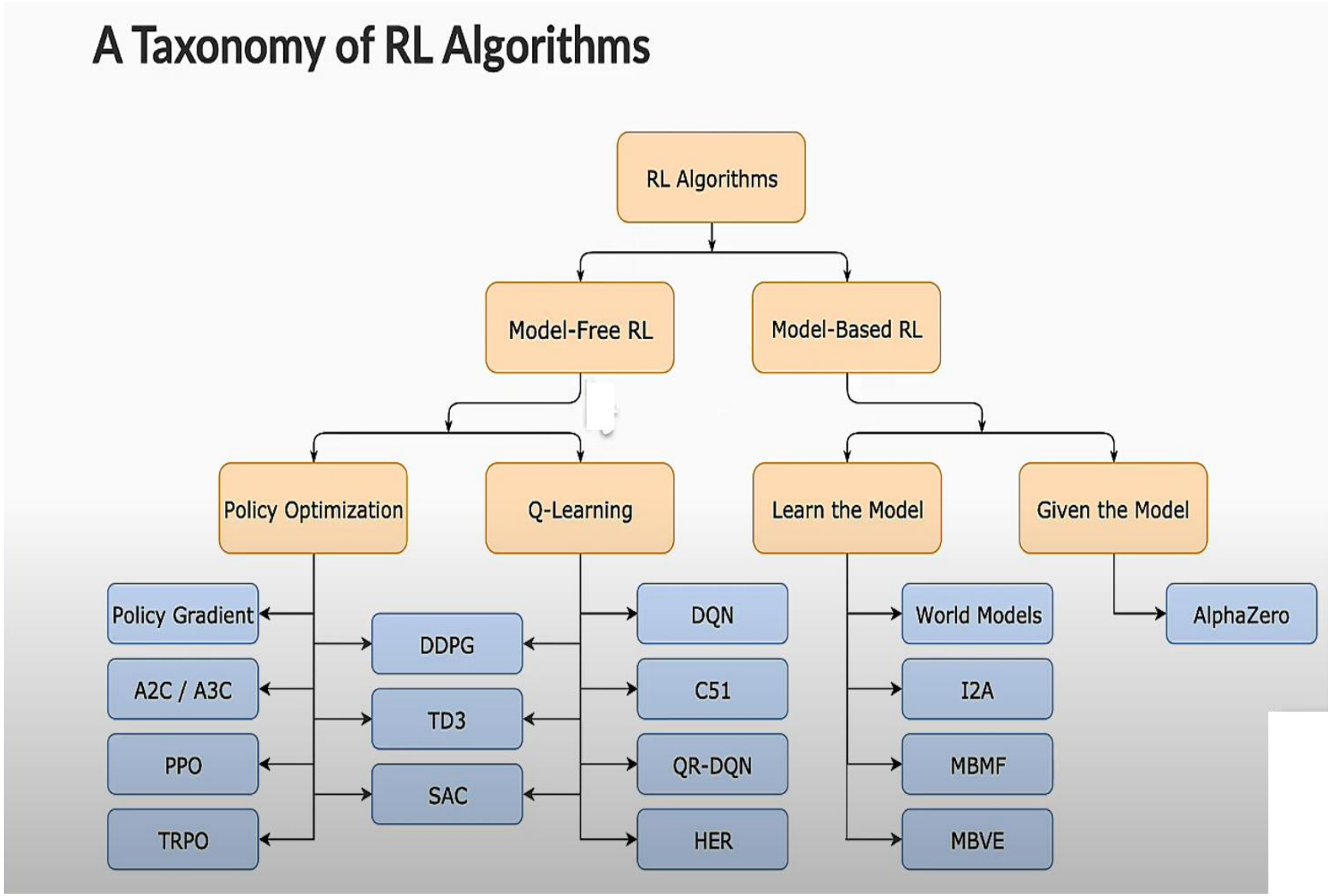
Two fundamental problems in sequential decision making

- Reinforcement Learning:
  - The environment is initially unknown
  - The agent interacts with the environment
  - The agent improves its policy
- Planning:
  - A model of the environment is known
  - The agent performs computations with its model (without any external interaction)
  - The agent improves its policy
  - a.k.a. deliberation, reasoning, introspection, pondering, thought, search

Model of the Environment:

- Model mimics the behavior of the environment. With the help of the model, one can make inferences about how the environment will behave. Such as, if a state and an action are given, then a model can predict the next state and reward.
- The model is used for planning, which means it provides a way to take a course of action by considering all future situations before actually experiencing those situations.
- The approaches for solving the RL problems **with the help of the model** are termed as the **model-based approach**.
- Comparatively, an approach **without using a model** is called a **model-free approach**.

RL algorithm is divided into two parts



- **Model-based Reinforcement Learning:**
  - This category of reinforcement learning algorithms involves creating a model of the environment's dynamics to make decisions and improve performance.
  - This model is ideal when the environment is static, and well-defined, where real-world environment testing is difficult.

Key Differences in between Model-free and Model-based Reinforcement Learning

Feature	Model-Free RL	Model-Based RL
Learning Approach	Direct learning from environment	Indirect learning through model building
Efficiency	Requires more real-world interactions	More sample-efficient
Complexity	Simpler implementation	More complex due to model learning
Environment Utilization	No internal model	Builds and uses a model
Adaptability	Slower to adapt to changes	Faster adaptation with accurate model
Computational Requirements	Less intensive	More computational resources needed
Examples	Q-Learning, SARSA, DQN, PPO	Dyna-Q, Model-Based Value Iteration

Simulated Environment

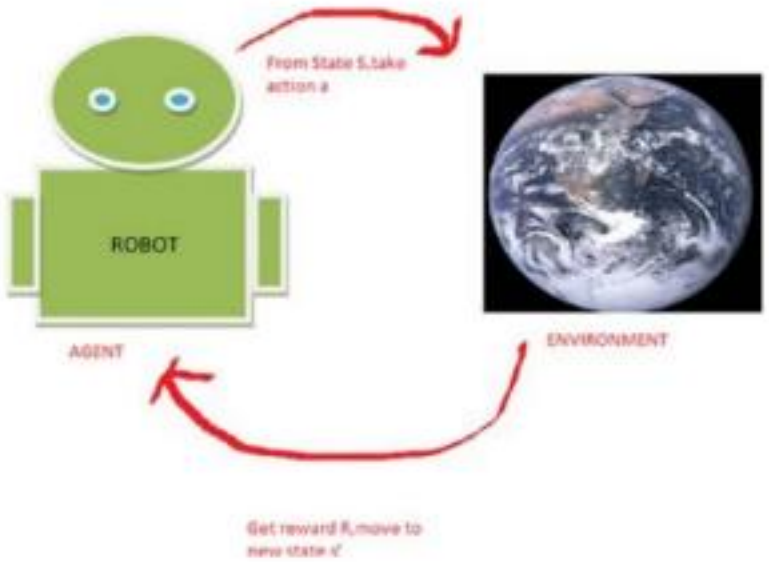
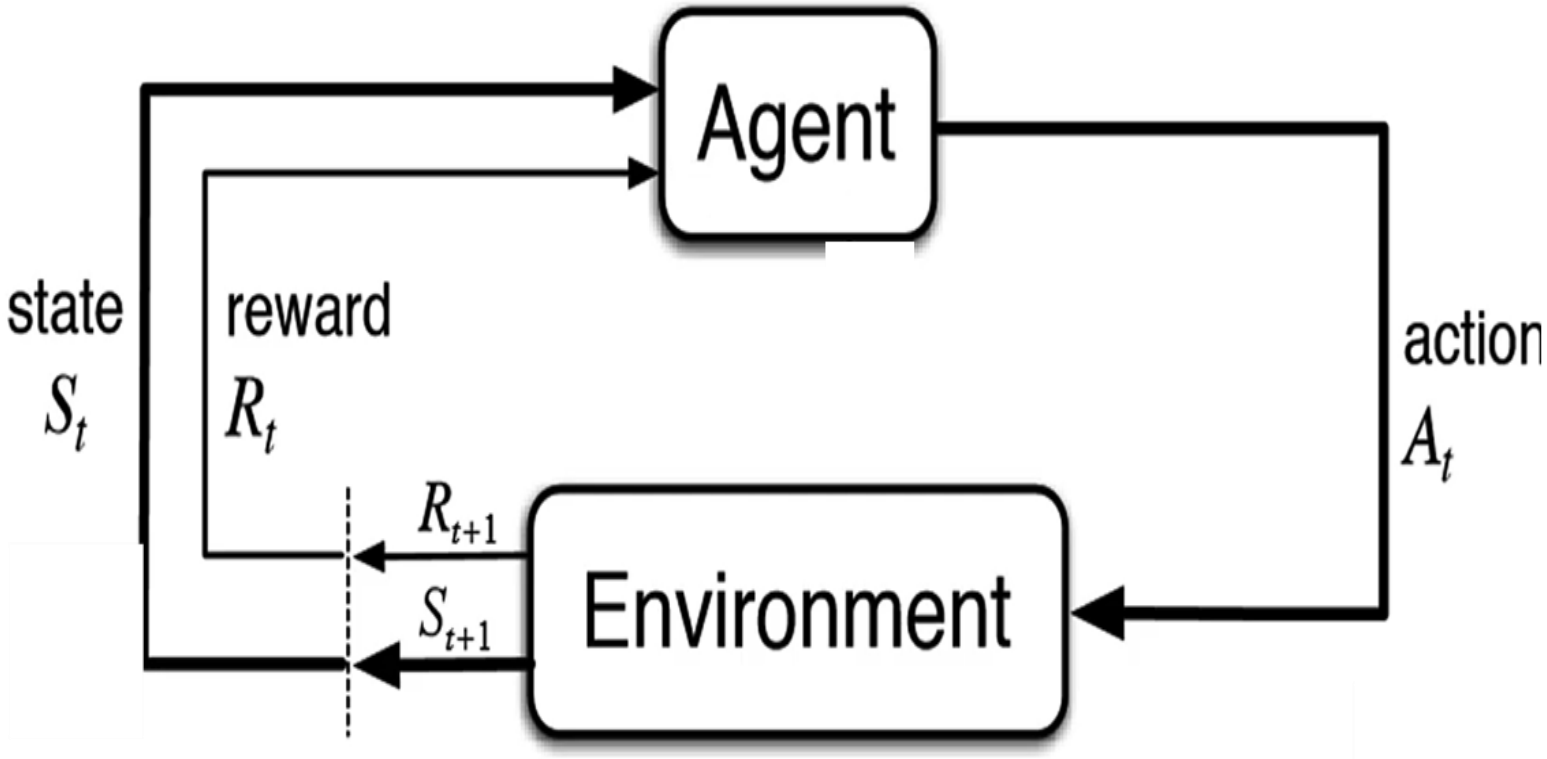


Figure 1-1. *Reinforcement Learning cycle*

Agent gain knowledge from the environment through sensors.
Agent performs action in the environment through actuators

Algorithm	Description	Policy	Action space	State space	Operator
Monte Carlo	Every visit to Monte Carlo	Either	Discrete	Discrete	Sample-means
Q-learning	State–action–reward–state	Off-policy	Discrete	Discrete	Q-value
SARSA	State–action–reward–state–action	On-policy	Discrete	Discrete	Q-value
Q-learning - Lambda	State–action–reward–state with eligibility traces	Off-policy	Discrete	Discrete	Q-value
SARSA - Lambda	State–action–reward–state–action with eligibility traces	On-policy	Discrete	Discrete	Q-value
DQN	Deep Q Network	Off-policy	Discrete	Continuous	Q-value
DDPG	Deep Deterministic Policy Gradient	Off-policy	Continuous	Continuous	Q-value
A3C	Asynchronous Advantage Actor-Critic Algorithm	On-policy	Continuous	Continuous	Advantage
NAF	Q-Learning with Normalized Advantage Functions	Off-policy	Continuous	Continuous	Advantage
TRPO	Trust Region Policy Optimization	On-policy	Continuous or Discrete	Continuous	Advantage

RL Framework



Main Characteristics of RL

- No supervisor while training
- Environment generally stochastic for real world applications
  - Model of the environment can be incomplete
- Feedback (Negative/Positive Reward) can be delayed or partial
- Agent uses experience form past to improve its performance over time
  - Actions which have fetched more rewards are preferred
- Agent tries various actions and prefers those actions that are best or have fetched more rewards
- RL uses Markov Decision Process Framework to define interaction between a learning agent and its environment

Reinforcement Learning(RL) Problem

- Challenges in RL
  - **Trade off between Exploration and Exploitation**
- To obtain a lot of reward, a RL agent must prefer actions that it has tried in the past and found to be effective in producing reward- **Exploit**
- But to discover such actions, it has to try actions that it has not selected before- **Explore**
- Note: **Neither exploration nor exploitation can be pursued exclusively without failing at the task**

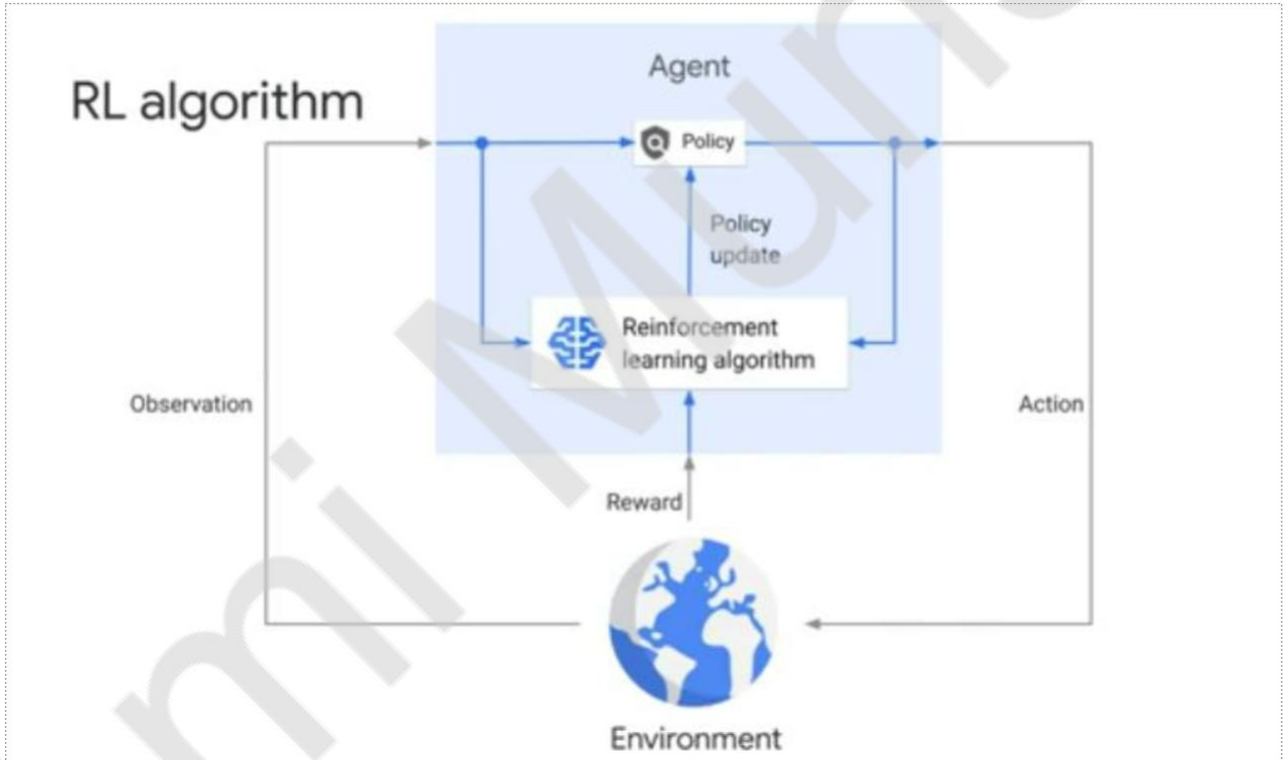
Ref: Richard Sutton, Andrew Barto, “Reinforcement Learning: An Introduction”, 2nd edition, MIT Press, 2018

Some applications of RL

- Resource Allocation
- Trading and Finance
- Healthcare
- Robotics
- Robotics

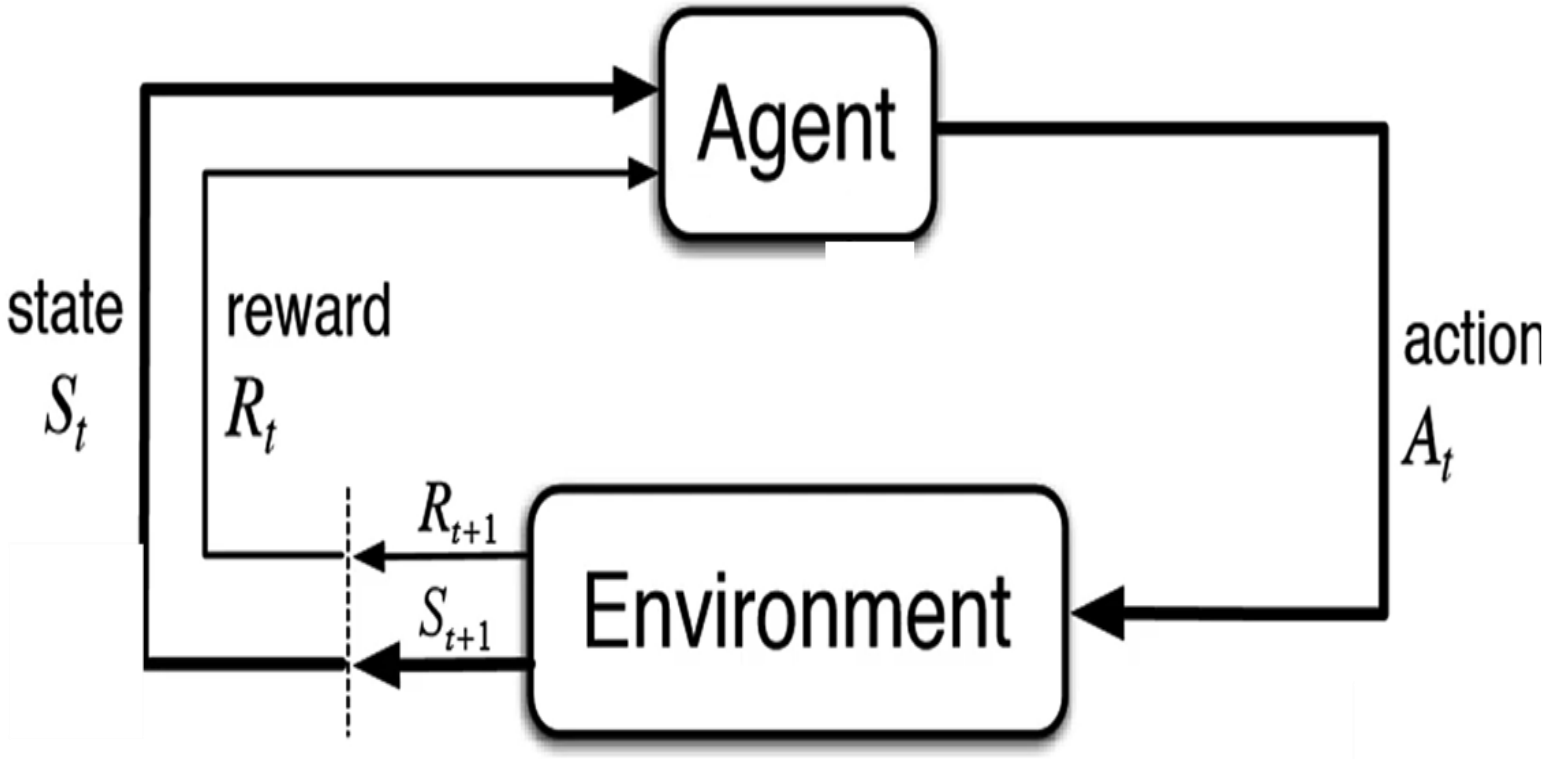
- Refer to some examples of RL in chapter 1 of Richard Sutton, Andrew Barto, “Reinforcement Learning: An Introduction”, 2nd edition, MIT Press, 2018

RL Algorithm



Ref: https://www.cloudskillsboost.google/course_templates/39/video/325107?locale=id

The RL Process: a loop of state, action, reward and next state



Ref: Richard Sutton, Andrew Barto, "Reinforcement Learning: An Introduction", 2nd edition, MIT Press, 2020
https://huggingface.co/learn/deep-rl-course/en/unit1/rl-framework

Elements of RL

- Reward signal indicates what is good in an immediate sense, a value function specifies what is good in the long run
- Roughly speaking, the value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state
- Rewards determine the immediate, intrinsic desirability of environmental states, values indicate the long-term desirability of states after taking into account the states that are likely to follow and the rewards available in those states
- For example, a state might always yield a low immediate reward but still have a high value because it is regularly followed by other states that yield high rewards
- Or the reverse could be true
- To make a human analogy
  - Rewards are somewhat like pleasure (if high) and pain (if low)
- Whereas values correspond to a more refined and farsighted judgment of how pleased or displeased we are that our environm

Elements of RL

- Final element of some reinforcement learning systems is a model of the environment
- This is something that mimics the behavior of the environment, or more generally, that allows inferences to be made about how the environment will behave
- For example, given a state and action, the model might predict the resultant next state and next reward

- Models are used for planning, by which we mean any way of deciding on a course of action by considering possible future situations before they are actually experienced

- Methods for solving reinforcement learning problems that use models and planning are called model-based methods, as opposed to simpler model-free methods that are explicitly trial-and-error learners—viewed as almost the opposite of planning

Recap

Reinforcement learning

Main sub elements of a RL

1) A policy

2) Reward signal

3) Value function

4) Model of the environment

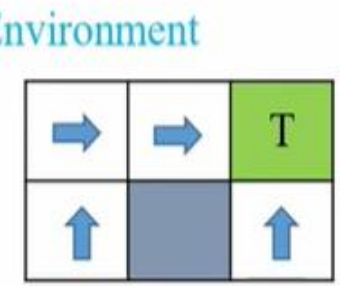
Reinforcement learning

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



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**

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.

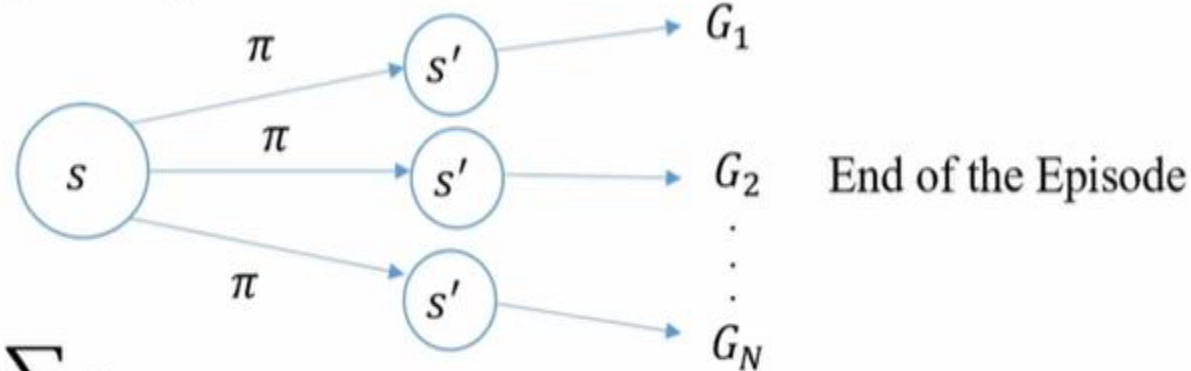
3 Value Function

✓ 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 ⇒ $V_\pi(s) = \frac{1}{N}\sum G_i$

Learning and Planning

Two fundamental problems in sequential decision making

- Reinforcement Learning:
  - The environment is initially unknown
  - The agent interacts with the environment
  - The agent improves its policy
- Planning:
  - A model of the environment is known
  - The agent performs computations with its model (without any external interaction)
  - The agent improves its policy
  - a.k.a. deliberation, reasoning, introspection, pondering, thought, search

Model of the Environment:

- Model mimics the behavior of the environment. With the help of the model, one can make inferences about how the environment will behave. Such as, if a state and an action are given, then a model can predict the next state and reward.
- The model is used for planning, which means it provides a way to take a course of action by considering all future situations before actually experiencing those situations.
- The approaches for solving the RL problems **with the help of the model** are termed as the **model-based approach**.
- Comparatively, an approach **without using a model** is called a **model-free approach**.

There are three main types of Reinforcement Learning

- Value-Bases
- Policy Based
- Model-Based

Each approach has its own strength and weakness, the choice of algorithm will depend on the specific problem you are trying to solve.

- **Value-Bases Reinforcement Learning :**
  - In this approach agent learns to estimate the value of each state or action based on the rewards it received.
  - This value is known as Q values.
  - The agent then select the actions with the highest Q-value in each state to maximize its long- term reward.
  - The most common used algorithm used for value-based reinforcement algorithm is **Q-learning.**
- **Policy Based Reinforcement Learning :**
  - In this approach the agent learns a optimal policy which is mapping from states to actions,without calculating value function
  - The policy is updated based on the rewards received by the agent, with the goal of maximizing the expected reward over time.
  - The most common algorithm used for policy-based reinforcement learning is **REINFORCE algorithm**.
- **Model Based Reinforcement Learning**:
  - The agent learns the model of the environment, which it can use to simulate the different scenarios and plan its action accordingly.
  - The model can learn through supervised learning or unsupervised learning and the agent can use it to predict the outcome of its actions before taking them.
  - The most common model-based reinforcement algorithm is **Dyna algorithm**.

Types of Reinforcement Learning Algorithms (on the basis of model based)

- There are various algorithms used in reinforcement learning such as Q-learning, policy gradient methods, Monte Carlo method and many more.
- All these algorithms can be classified into two broad categories –
- **Model-free Reinforcement Learning** :
  - It is a category of reinforcement learning algorithms that learns to make decisions by interacting with the environment directly, without creating a model of the environment's dynamics.
  - The agent performs different actions multiple times to learn the outcomes and creates a strategy (policy) that optimizes its reward points. This is ideal for changing, large or complex environments.
  - Not applicable for some scenario like self driving car

- **Model-based Reinforcement Learning:**
  - This category of reinforcement learning algorithms involves creating a model of the environment's dynamics to make decisions and improve performance.
  - This model is ideal when the environment is static, and well-defined, where real-world environment testing is difficult.

Key Differences in between Model-free and Model-based Reinforcement Learning

Feature	Model-Free RL	Model-Based RL
Learning Approach	Direct learning from environment	Indirect learning through model building
Efficiency	Requires more real-world interactions	More sample-efficient
Complexity	Simpler implementation	More complex due to model learning
Environment Utilization	No internal model	Builds and uses a model
Adaptability	Slower to adapt to changes	Faster adaptation with accurate model
Computational Requirements	Less intensive	More computational resources needed
Examples	Q-Learning, SARSA, DQN, PPO	Dyna-Q, Model-Based Value Iteration

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**.
- ✓ 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

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**

$$a = \pi(s)$$

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

Formal presentation of Fundamentals

✓ What is the **Return** (G_t)?

- It is the total future rewards at state s_t

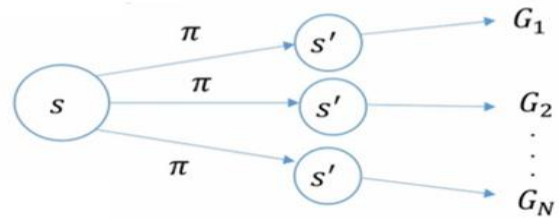
T is the end of the episode

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

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

- ✓ 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

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

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

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 Processes
  - ✓ **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

Immediate Reinforcement Learning

- Immediate reinforcement learning (Immediate RL) and general reinforcement learning (RL) share **core principles** but differ mainly in the timing and manner of policy updates and learning processes.

Immediate Reinforcement Learning (Immediate RL)

- ***Policy Update Frequency:***
  - ❖**Immediate RL** updates the policy or value function after every action taken by the agent. This means that the agent learns and adapts its strategy in real-time as it interacts with the environment.
- ***Learning Approach:***
  - ❖**Online Learning**: Immediate RL typically follows an **online learning approach**, where updates are made continuously and incrementally based on each new piece of data received.

Immediate RL Vs Full RL

- In Immediate RL
  - Agent receive rewards immediately after each action
  - Hence decision making becomes quicker
- In Full RL
  - Full reinforcement learning considers delayed rewards, requiring agents to strategize for long-term goals
  - This requires agents to strategize for long term goals
  - Profound understanding of environment is needed in this case

Immediate RL

Agent takes action Atat time $t$to get an immediate reward Rt

- Agent needs to explore all the available actions to identify the near optimal action which may maximize the cumulative reward
- Once after enough exploration, near optimal action is identified, then the agent can explore that action
- Here we encounter explore/exploit dilemma or conflict
- How much to explore and when to start exploiting
- Example of Immediate RL is Bandit Problem

Examples of Immediate Reinforcement Based in real life scenarios

- Giving Treats for Homework Completion
- Earning Points in a Game
- Receiving Applause After a Performance
- Receiving Praise for a Task
- Getting Paid Directly After Work
- Eating immediately after Feeling Hungry
- Social Media Notifications

Ref: https://fuskal.com/immediate-reinforcement/#Immediate_Reinforcement_and_Operant_Conditioning

Examples of Delayed Reinforcement-Based in real life scenarios

- Saving Money for Future Goals
- Completing a Degree for Career Advancement
- Physical Fitness and Exercise
- Learning a Musical Instrument
- Learning a New Language

Ref: https://fuskal.com/immediate-reinforcement/#Immediate_Reinforcement_and_Operant_Conditioning

- ***Suitability:***

❖**Real-Time Applications:** Ideal for scenarios where decisions need to be made **quickly and continuously**, such as in robotics, real-time game playing, or live trading systems.

Examples:

❖**Tic-Tac-Toe:**

- An agent updates its strategy after each move based on the immediate reward received.

❖**Self-Driving Cars:**

- The car's control system updates its driving policy in real-time as it gets feedback from the environment.

General Reinforcement Learning (RL)

- ***Policy Update Frequency:***

❖ RL does not necessarily update the policy after every action. It can use batch learning, where updates are made after accumulating a batch of experiences or updates at the end of an episode or after a predefined period.

- ***Learning Approach:***

❖**Offline and Online Learning:**

- General RL can follow both online and offline learning approaches.
- In offline learning, the agent gathers experiences and updates its policy after the interaction phase is over, possibly using more computational resources to perform these updates.

- Agent: The learner or decision-maker.
- Environment: Everything the agent interacts with.
- Action: All the possible moves the agent can make.
- State: A situation the agent finds itself in within the environment.
- Reward: A feedback signal. After each action, the agent receives a reward (or penalty), which is a numerical score, to learn what is good and bad.

- **Policy:** A strategy that the agent employs to determine its next action based on the current state.

- **Value Function:** It estimates what the total amount of reward an agent can expect to accumulate over the future, starting from a specific state.

State-action value function (Q function)

A state-action value function is also called the *Q* function. It specifies how good it is for an agent to perform a particular action in a state with a policy π. The *Q* function is denoted by *Q(s)*. It denotes the value of taking an action in a state following a policy π.

Temporal Difference

- Simple rule to explain complex behaviors
- Intuition: Prediction of outcome at time $t+1$ is better than the prediction at time t. Hence use the later prediction to adjust the earlier prediction.
- Has had profound impact in behavioral psychology and neuroscience!

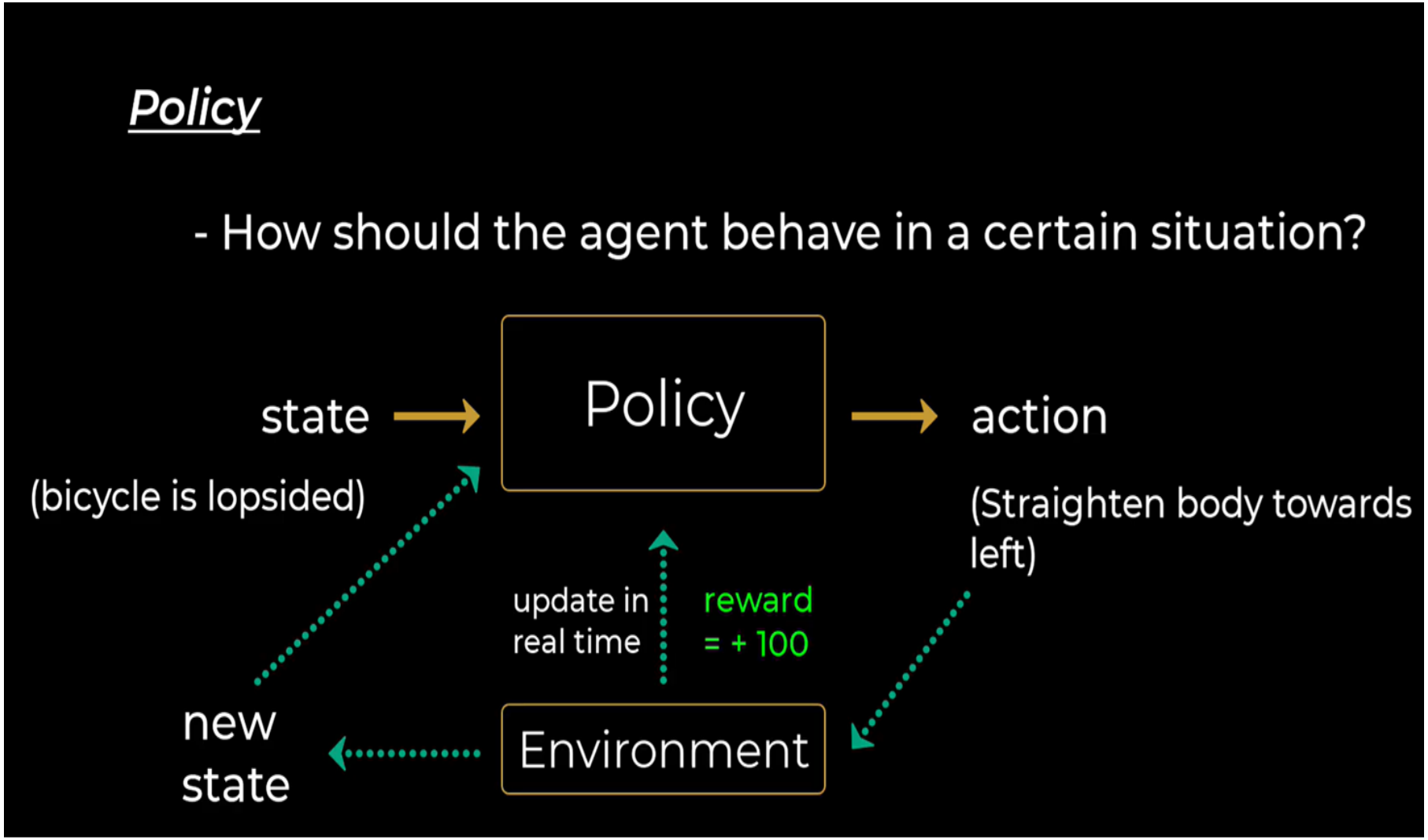
Optimal Control

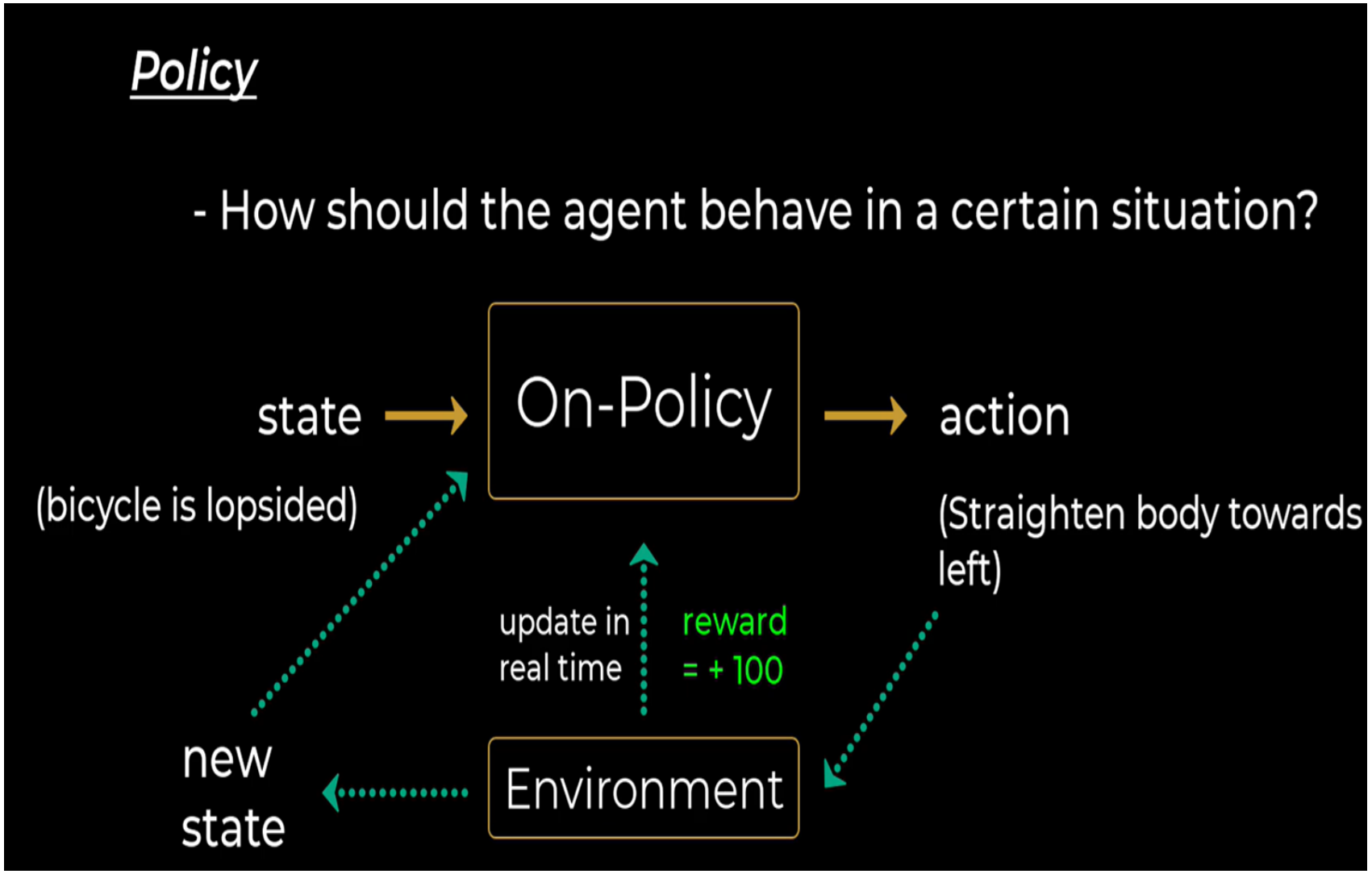
- Branch of mathematical optimization
- Designing a controller to maximize or minimize the objective function
- Optimal control involves finding a control policy that optimizes a certain objective, typically the cumulative reward or cost over time
- It deals with dynamical systems and aims to determine the best sequence of actions to take from any given state to achieve the optimal outcome.

Dynamic Programming

- Dynamic programming is a mathematical approach used to solve optimization problems by breaking them down into simpler subproblems
- In the context of Markov Decision Processes (MDPs), DP methods are used to find optimal policies by solving the Bellman equations
- Two primary DP methods are:
  - **Policy Iteration**: Alternates between evaluating a policy and improving it
  - **Value Iteration**: Iteratively updates the value function directly to find the optimal policy

RL strategies can be either On Policy and Off Policy : In the world of Reinforcement Learning (RL), two primary approaches dictate how an agent (like a robot or a software program) learns from its environment: On-policy methods and Off-policy methods.





Off Policy

- Agent learn how to ride bike by observing someone's else policy.

Return:

- Cumilative reward over time
- Agent should maximize return

Value Function

- Expected value of Return

Elements of Reinforcement Learning

Policy:

- Policy defines the learning agent behavior for given time period. It is a mapping from **perceived states of the environment to actions to be taken when in those states.**

- A policy is the agent's behaviour
- It is a map from state to action, e.g.
- Deterministic policy: $a = \pi(s)$
- Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$

Reward function:

- Reward function is used to define a goal in a reinforcement learning problem. A reward function is a function that provides a numerical score based on the state of the environment.

Elements of Reinforcement Learning

Value function: OR State- Value function for the policy π

- Value functions specify what is good in the long run.
- The value of a state is the total amount of reward an agent can expect to accumulate over the future, starting from that state.

- Value function is a prediction of future reward
- Used to evaluate the goodness/badness of states
- And therefore to select between actions, e.g.

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s \right]$$

ACTION - VALUES

Or State-Action Value Function

The value is the Expected Reward

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a] \quad \forall a \in \{1, \ldots, k\}$$

$\doteq$ → Is defined as

$\forall$ → For all

$\mathbb{E}$ → Expectation

$\in$ → Belongs to

- It specifies how good it is for an agent to perform a **particular action** in a state with policy pi.
- The Q function is denoted by Q(s).
- It denotes the value of taking an action in a state following a policy pi

ACTION - VALUES

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a] \quad \forall a \in \{1, \ldots, k\}$$

$$= \sum_r p(r|a)\, r$$

The Goal is to **Maximize** the **expected reward**

$$\underset{a}{\text{argmax}} \; q_*(a)$$

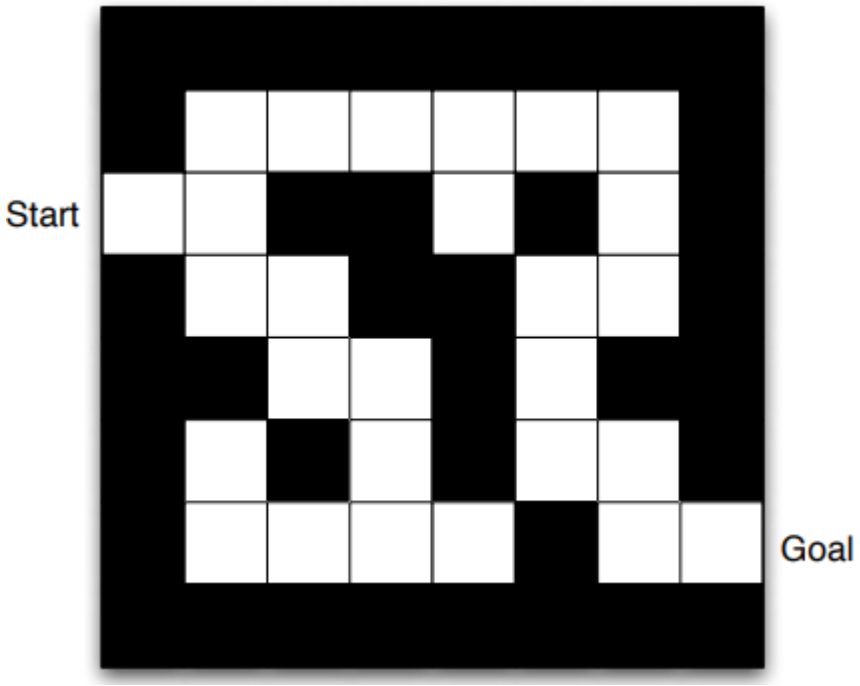
argmax means argument which maximize our function q*a

Model of the environment

- A model predicts what the environment will do next
- $\mathcal{P}$ predicts the next state
- $\mathcal{R}$ predicts the next (immediate) reward, e.g.

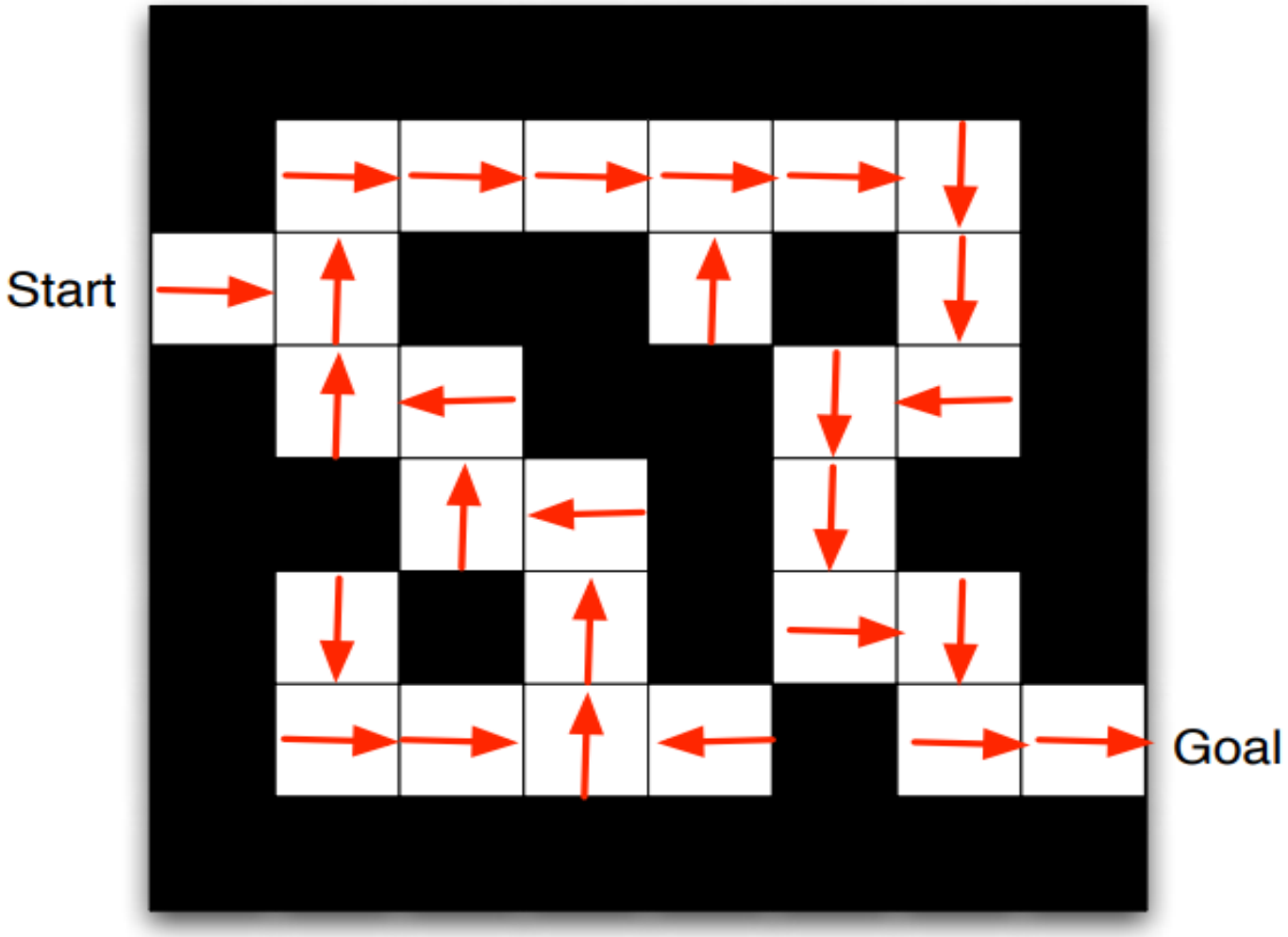
$$\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$$
$$\mathcal{R}^a_s = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$$

Maze Problem



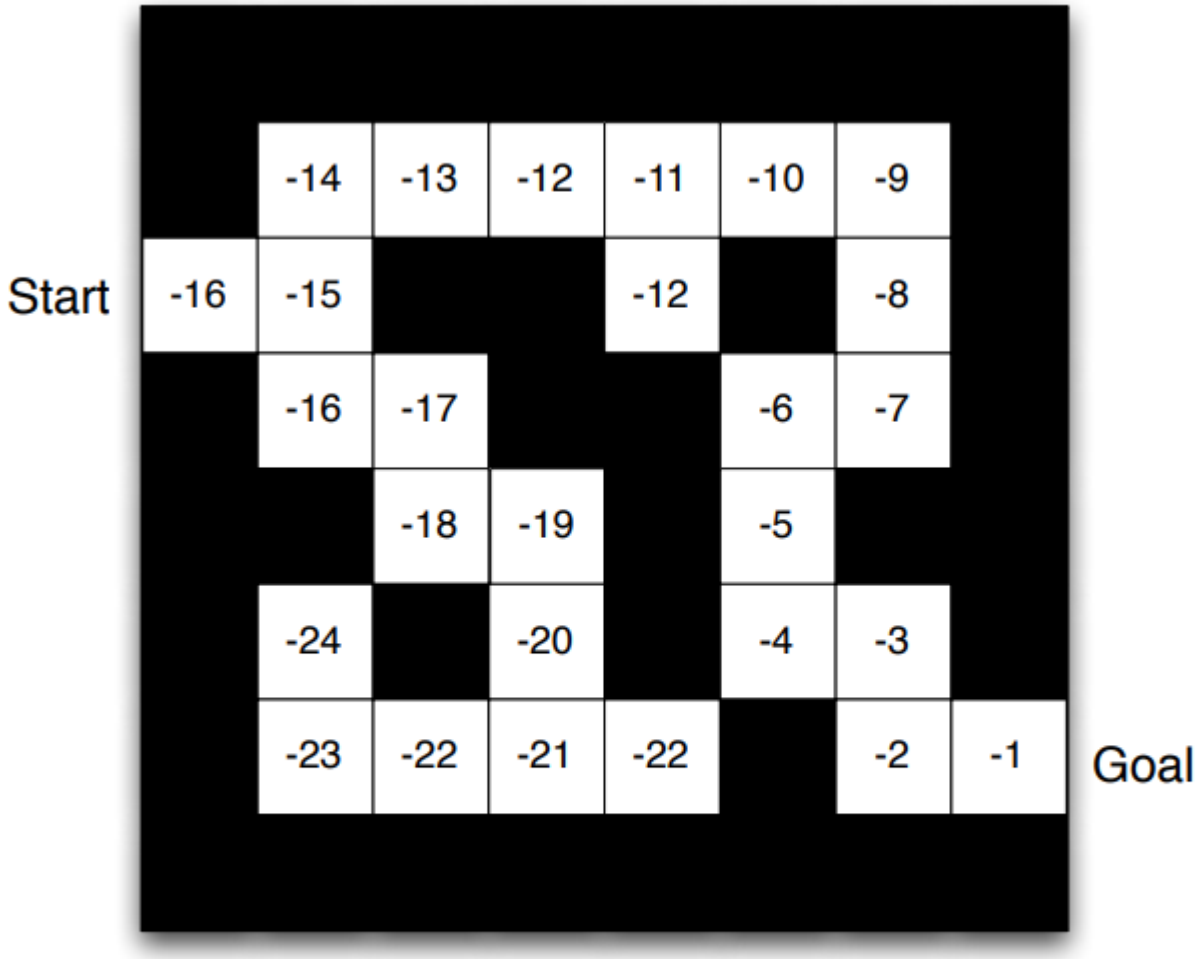
- Rewards: -1 per time-step
- Actions: N, E, S, W
- States: Agent's location

Maze Example: Policy



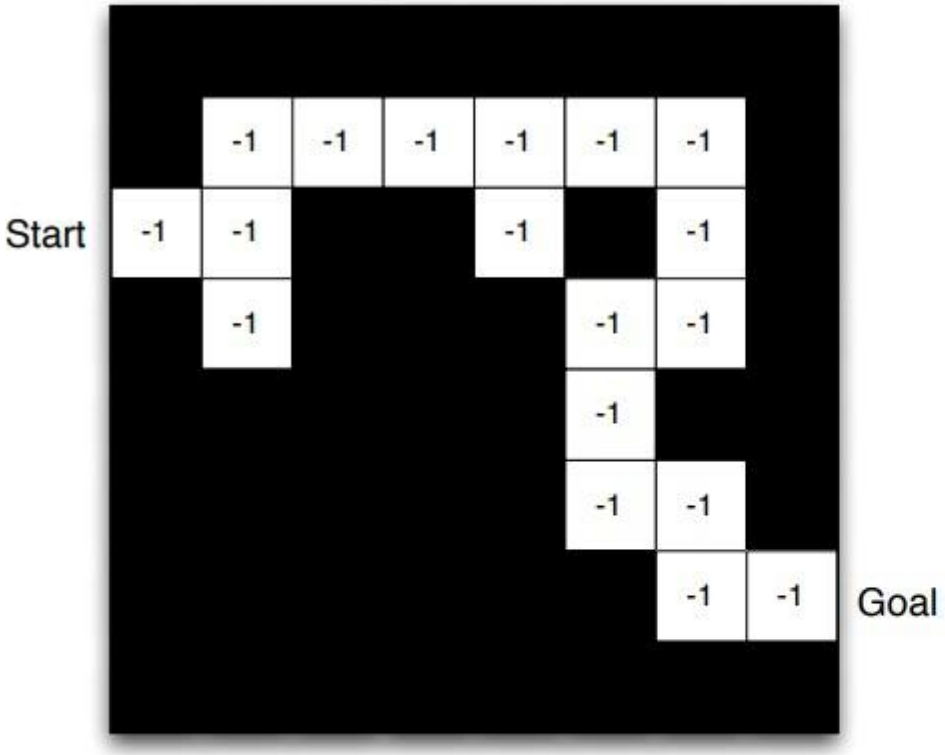
- Arrows represent policy $\pi(s)$ for each state s

Maze Example: **Value Function**



- Numbers represent value $v_\pi(s)$ of each state s

Maze Example: Model



- Agent may have an internal model of the environment
- Dynamics: how actions change the state
- Rewards: how much reward from each state
- The model may be imperfect

- Grid layout represents transition model $\mathcal{P}^a_{ss'}$
- Numbers represent immediate reward $\mathcal{R}^a_s$ from each state s (same for all a)