

# Lecture 1: Introduction to Reinforcement Learning

**Idea:** An automated agent learning through experience/data to make good decisions under uncertainty.

---

## Reinforcement Learning (Generally) Involves:

- **Optimization:**
  - The goal is to find an optimal way to make decisions.
    - Yielding best outcomes or at least very good outcomes
  - Explicit notion of decision utility.
  - Example: finding minimum distance route between two cities given network of roads.
- **Delayed Consequences:**
  - Decisions now can impact things much later...
  - Introduces two challenges:
    - When planning: decisions involve reasoning about not just immediate benefit of a decision but also its longer term ramifications.
    - When learning: temporal credit assignment is hard (what caused later high or low rewards?)
- **Exploration:**
  - Learning about the world by making decisions.
    - Agent as scientist.
    - Learn to ride a bike by trying and failing.
  - Decisions impact what we learn about.
    - Only get a reward for decision made.
    - Don't know what would have happened for other decision.
    - If we choose to go to Stanford instead of MIT, we will have different later experiences...
- **Generalization:**

- Policy is **mapping from past experience to action** (i.e. the rule of how to act in certain circumstances).
- Why not just pre-program a policy?
  - The scenario space is HUGE → We can't write rules to address each of these cases individually.

### RL vs Other AI and Machine Learning:

	AI Planning	Supervised Learning	Unsupervised Learning	Reinforcement Learning	Imitation Learning
Optimization	X			X	X
Learns from experience		X	X	X	X
Generalization	X	X	X	X	X
Delayed Consequences	X			X	X
Exploration				X	

- Imitation learning typically assumes input demonstrations **of good policies**. ← **learn only from the experts**.
- Imitation Learning reduces Reinforcement Learning to Supervised Learning. For many good reasons, imitation learning is very popular.

### Two Problem Categories where RL is Particularly Powerful

1. **No examples of desired behavior.**
  - e.g., because **the goal is to go beyond human performance** or there is no existing data for a task.
2. **Enormous search or optimization problem with delayed outcomes.**
  - For problem with really really large search space, even there, we may not have great techniques for solving them. → It's sort of a reduction that people can think of taking a planning problem and trying to reduce it to a reinforcement learning problem to make it more tractable.

---

### Course Outline:

- Markov decision processes & planning.
- Model-free policy evaluation.

- Model-free control.
- Policy Search.
- Offline RL **including RL from Human Feedback and Direct Preference Optimization.**
- Exploration.
- Advanced Topics.

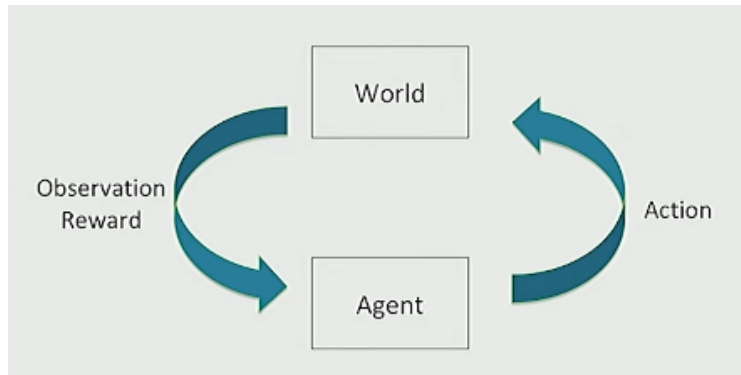
---

**Refresher Exercise: AI tutor as a Decision Process:**

- Student initially does not know either addition (easier) nor subtraction (harder).
- AI tutor agent can provide practice problems about addition or subtraction.
- AI agent gets rewarded +1 if student gets problem right, -1 if gets problem wrong.
- Model this as a Decision Process.
  - Given the easier problem first, if student gets it right → move to the harder problem next, else give another easy problem.
- Define state space, action space, and reward model.
  - State space: Student get the easy problem right or wrong, the hard problem right or wrong.
  - Action space: AI tutor controls between the easier or harder problems.
  - Reward model: +1 if student gets problem right, -1 if gets problem wrong.
- What does the dynamics model represent?
  - The learning progress of the student for the easy or hard problem types.
- What would a policy to optimize the expected discounted sum of rewards yields?
  - The cumulative reward of either the addition or subtraction problem to be as high as possible.

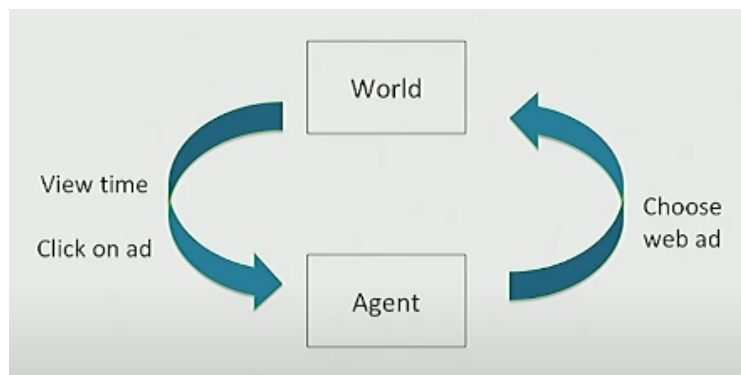
---

**Sequential Decision Making**

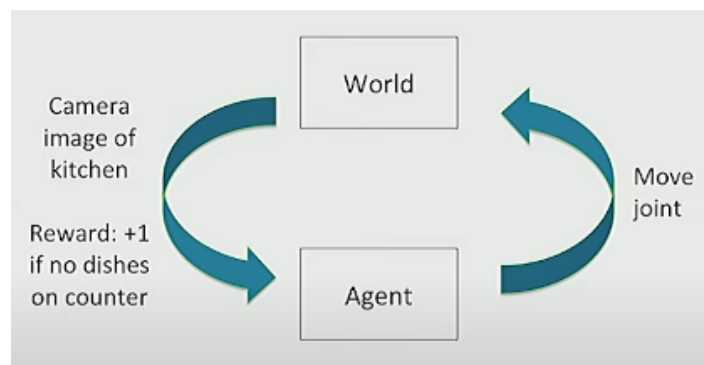


- Goal: Select actions to maximize total expected future reward.
- May require balancing immediate & long term rewards.

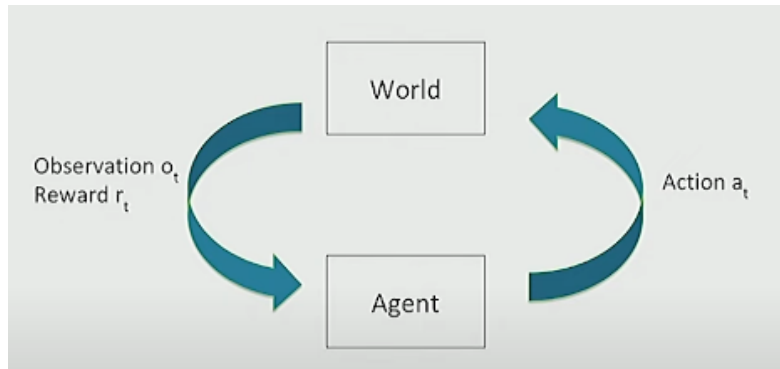
Example: Web Advertising



Example: Robot Unloading Dishwasher



### Sequential Decision Process: Agent & the World (Discrete Time)



Assume that we have a finite series of time steps.

- Each time step  $t$ :
  - Agent takes an action  $a_t$ .
  - World updates given action  $a_t$ , emits observation  $o_t$  and reward  $r_t$ .
  - Agent receives observation  $o_t$  and reward  $r_t$ .

### History: Sequence of Past Observations, Actions & Rewards

- History  $h_t = (a_1, o_1, r_1, \dots, a_t, o_t, r_t)$ .
- Agent chooses action based on history.
- State is information assumed to determine what happens next.
  - Function of history:  $s_t = (h_t)$

---

### Markov Assumption

- Information state: sufficient statistic of history.
- State  $s_t$  is Markov if and only if:
  - $p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t)$
  - $\rightarrow$  Sufficient if assume  $s_t == h_t$ .
- Future is independent of past given present.

**Note:** observation is independent from state .

Example: In atari games, the **state** of the agent is the history of the last 4 consecutive frames (to capture the temporal dependencies or momentum of the objects) while the **observation** is only the current frame.

### Why is Markov Assumption Popular?

- Simple and often can be satisfied if include some history as part of the state.
- In practice often assume most recent observation is sufficient statistic of history:  $s_t = o_t$ .
- State representation has big implications for:
  - Computational complexity.
  - Data required.
  - Resulting performance.

### Types of Sequential Decision Processes

- Is the state Markov? Is the world partially observed? (Partially Observed Markov Decision Process)
- Are dynamics deterministic or stochastic?
- Do actions influence only immediate reward (bandits) or reward and next state?

### MDP Model

- **Agent's representation** of how world changes given agent's action (not the actual model of the world).
- Transition / dynamics **model predicts next agent state**.
  - $p(s_{t+1} = s' | s_t = s, a_t = a)$  or  $p(s' | s, a)$
- Reward model predicts immediate reward (if I'm in this state, and I take this action, what will be my immediate reward?).
  - $r(s_t = s, a_t = a) = \mathbb{E}[r_t | s_t = s, a_t = a]$  or  $r(s, a) = \mathbb{E}[r_t | s, a]$

### Policy

- Policy  $\pi$  determines how the agent chooses actions in a specific state.
- $\pi : S \rightarrow A$ , mapping from states to actions.
- Two types of policy:
  - **Deterministic** policy (we only have one action correspond to a state):

- $\pi(s) = a$
- **Stochastic** policy:
  - $\pi(a|s) = Pr(a_t = a | s_t = s)$

**Note:** Our model might start with a particular policy. Over time, it will explore lots of different policies in trying to search for something that's good.

## Evaluation and Control

- Evaluation:
  - (Someone give you a fixed policy and you want to know how good it is)
  - Estimate/predict the expected rewards from following a given policy.
- Control:
  - Optimization: find the best policy.

---

## Markov Process or Markov chain

- **Memoryless random process.**
  - A Markov Process is a type of random process where the future state only depends on the current state and not on the sequence of states that came before. This is called the **Markov Property**.
  - Mathematically:
    - $P(s_{t+1} | s_t, s_{t-1}, \dots, s_0) = P(s_{t+1} | s_t)$
    - This means that the probability of moving to the next state  $s_{t+1}$  depends only on the current state  $s_t$ , not the entire history.
- **Definition of Markov Process:**
  - $S$  is a (finite) set of states ( $s \in S$ ).
  - $P$  is **dynamics / transition model** that specifies the probability of transitioning from one state to another, expressed as  $p(s_{t+1} = s' | s_t = s)$ , which means the probability of transitioning to state  $s'$  from state  $s$ .
- **Note: no rewards, no actions.**
  - A **Markov Process** is purely **focused** on **state transitions**. It does not involve:
    - **Reward:** Unlike Reinforcement Learning, where states and actions are associated with rewards.

- **Actions:** It doesn't model decisions that lead to different transitions - just the probabilistic behavior of the process.

- **Transition Matrix Representation ( $P$ ):**

- When a number of states  $N$  is finite, the transition probabilities between states can be represented as a matrix.

- $$P = \begin{bmatrix} P(s_1|s_1) & P(s_2|s_1) & \dots & P(s_N|s_1) \\ P(s_1|s_2) & P(s_2|s_2) & \dots & P(s_N|s_2) \\ \vdots & \vdots & \ddots & \vdots \\ P(s_1|s_N) & P(s_2|s_N) & \dots & P(s_N|s_N) \end{bmatrix}$$

- Each row represents the probabilities of transitioning from a particular state to all other states.
- Each row has the sum of 1.

- ▼ Example to illustrate:

- Imagine a weather system with 3 states:

- $s_1$ : Sunny.
- $s_2$ : Cloudy.
- $s_3$ : Rainy.

- The transition probabilities might look like this:

- $$P = \begin{bmatrix} 0.8 & 0.1 & 0.1 \\ 0.3 & 0.4 & 0.3 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$$

- Row 1: If it's sunny ( $s_1$ ), there's an 80% chance it will stay sunny ( $P(s_1|s_1)$ ), 10% chance it becomes cloudy ( $P(s_2|s_1)$ ), and 10% chance it will rain ( $P(s_3|s_1)$ ).
- Row 2: If it's cloudy ( $s_2$ ), there's an 30% chance it becomes sunny ( $P(s_1|s_2)$ ), 40% chance it stays cloudy ( $P(s_2|s_2)$ ), and 30% chance it will rain ( $P(s_3|s_2)$ ).
- Row 3: If it's rainy ( $s_3$ ), there's an 20% chance it will become sunny ( $P(s_1|s_3)$ ), 30% chance it becomes cloudy ( $P(s_2|s_3)$ ), and 50% chance it will rain ( $P(s_3|s_3)$ ).

---

## Markov Reward Process (MRP)

- Markov Reward Process is an extension of a Markov Process where rewards are added to the states.



- It combines:
  1. The state transitions from a **Markov Process**.
  2. A **Reward function** that assigns expected reward to states.
- Key components of an MRP:
  - $S$  is a finite set of states ( $s \in S$ ).
  - $P$  is dynamics/transition model that specifies  $P(s_{t+1} = s' | s_t = s)$ , which defines the probability of transitioning to state  $s'$  from state  $s$ .
  - $R$  is a reward function  $R(s_t = s) = \mathbb{E}[r_t | s_t = s]$  for being in state  $s$ .
  - Discount factor  $\gamma \in [0, 1]$  that determines how future rewards are valued relative to immediate rewards.
    - If  $\gamma = 0$ , only immediate rewards are considered.
    - If  $\gamma = 1$ , future rewards are valued equally as immediate rewards.
- Note: no actions.
  - Similar to Markov Process, an MRP doesn't involve actions. It models the system's behavior and rewards but doesn't account for decisions or strategies.
- If finite number ( $N$ ) of states, can express  $P$  as a matrix like in Markov Process, and  $R$  as a vector.
- Discounted Return:
  - The goal in an MRP is often to compute the **discounted return**, which is the total reward received over time, discounted by  $\gamma$ :
    - $G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$
    - This can be rewritten as:  $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$

---

## Return & Value Function

- Horizon ( $H$ ):
  - **Definition:** The horizon refers to the number of time steps in an episode, which can either be finite or infinite:
    - **Finite horizon:** The process ends after a certain number of steps ( $H$  is a finite number).
    - **Infinite horizon:** The process continues indefinitely ( $H = \infty$ ).
  - Otherwise called **finite** Markov reward process.
- Definition of Return,  $G_t$  (for a Markov Reward Process)

- **Definition:** The return at time  $t$  is the **discounted sum of rewards** from time  $t$  up to the horizon  $H$ .
  - $G_t = \sum_{k=0}^{H-1} \gamma^k r_{t+k}$
- Key points:
  - If  $\gamma = 1$ , no discounting occurs, and all rewards are treated equally.
  - If  $\gamma < 1$ , rewards further in the future are given less weight.
- Definition of State Value Function,  $V(s)$  (for a Markov Reward Process):
  - **Definition:** The state value function measures the **expected return** from starting from a particular state  $s$ . It accounts for all possible transitions and rewards from that state onward:
 
$$V(s) = \mathbb{E}[G_t | s_t = s]$$
    - $$= \mathbb{E}\left[\sum_{k=0}^{H-1} \gamma^k r_{t+k} | s_t = s\right]$$

$$= \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^{H-1} r_{t+H-1} | s_t = s]$$

---

## Discount Factor

- Mathematically convenient (avoid infinite returns and values)
- Humans often act as if there's a discount factor  $< 1$ .
- If episode lengths are always finite ( $H < \infty$ ), can use  $\gamma = 1$ .