# Reinforcement Learning

Prof. M. Elif Karslıgil

Yildiz Technical University

Computer Engineering Department

Intelligent Systems Laboratory

Adapted from slides by

# Reinforcement Learning

- Learning algorithms differ in the information available to learner
  - **Supervised**: correct outputs, e.g., class label
  - **Unsupervised**: no feedback, must construct measure of good output
  - **Reinforcement learning**: Reward (or cost)

# Reinforcement Learning

- In supervised learning, the problem is to predict an output $t$ given an input $x$.
- But often the ultimate goal is not to predict, but to make decisions, i.e., take actions.
- And we need to take a sequence of actions.
- The actions have long-term consequences.

An agent

observes the world

takes an action and its states changes

with the goal of achieving long-term rewards.

Reinforcement Learning Problem: An agent continually interacts with the environment. How should it choose its actions so that its long-term rewards are maximized?

# Reinforcement Learning

- Classification and regression are one-time tasks. That is, a classification/regression model is given an input and returns an output. Then, the next input it is given it entirely independent from the first.

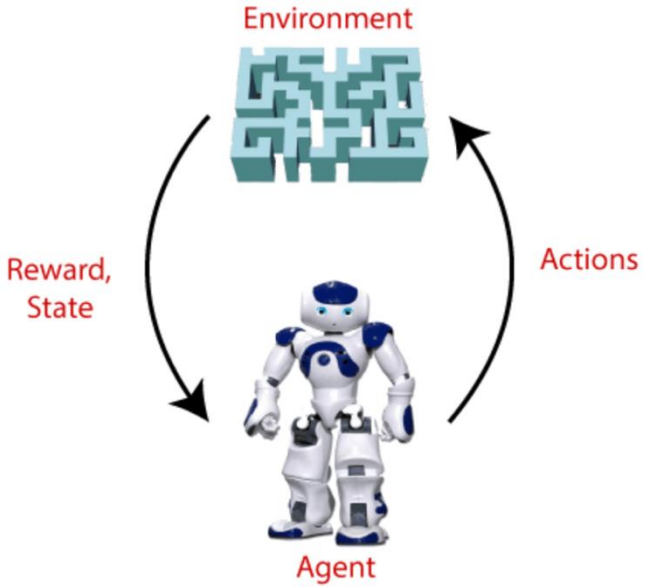
- In sequential decision making problems, we need to make a series of decisions over time, in which each decision influences the possible future.

# What is Markov Decision Process

- A Markov Decision Process (MDP) is a fully observable, probabilistic state model.
- A MDP is a framework for describing sequential decision-making problems.
- If there is only one outcome for each action (with probability 1), then the problem is deterministic. MDP assumes that each action could have multiple outcomes, with each outcome associated with a probability.
- MDPs consider stochastic non-determinism; that is, where there is a probability distribution over outcomes.

# Reinforcement Learning

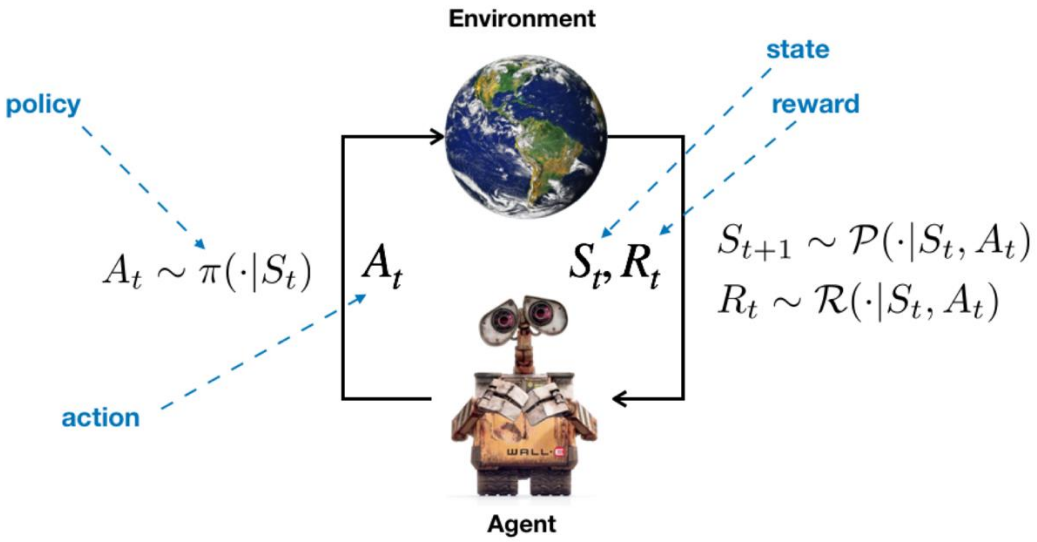
- Markov Decision Process DP has five components that work together:
  - **Agent:** It's the model eg. the robot.
  - **Environment:** the real-world environment which the agent interacts as part of its operation. eg. the terrain that the robot has to navigate, its surroundings, factors such as wind, friction, lighting, temperature etc.
  - **State:** represents the current 'state of the world' at any point. eg. it could capture the position of the robot relative to its terrain, the position of objects around it, and perhaps the direction and speed of the wind.
  - **Action:** these are the actions that the agent takes to interact with the environment. eg. The robot can turn right, left etc.
  - **Reward:** is the positive or negative reinforcement that the agent receives from the environment as a result of its actions



# Elements of Reinforcement Learning

- An **agent** interacts with an **environment** (e.g. game of Breakout)
- In each time step $t$,
  - the agent receives **observations** (e.g. pixels) which give it information about the **state** $\mathbf{s}_t$ (e.g. positions of the ball and paddle)
  - the agent picks an **action** $\mathbf{a}_t$ (e.g. keystrokes) which affects the state
- The agent periodically receives a **reward** $r(\mathbf{s}_t, \mathbf{a}_t)$, which depends on the state and action (e.g. points)
- The agent wants to learn a **policy** $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$
  - Distribution over actions depending on the current state and parameters $\theta$

# Elements of Reinforcement Learning




- policy $\pi : S \mapsto A$
  - A map from state space to action space.
  - May be stochastic.
- reward function *R(S)*
  - It maps each state (or, state-action pair) to a real number, called reward.
- value function
  - Value of a state (or, state-action pair) is the total expected reward, starting from that state (or, state-action pair).

# Reinforcement Learning vs. Supervised Learning

- Supervised learning

$$f : X \rightarrow Y$$

  - X: inputs
  - Y: outputs
  - The predicted outputs can be evaluated immediately by the teacher
  - *f* evaluated with loss function

- Reinforcement learning for learning a policy

$$\pi \ : \ S \mapsto A$$

  - S: states
  - A: actions
  - Take action A to affect state S
  - The predicted action A cannot be evaluated directly but can be given positive/negative rewards
  - Policy evaluated with value functions

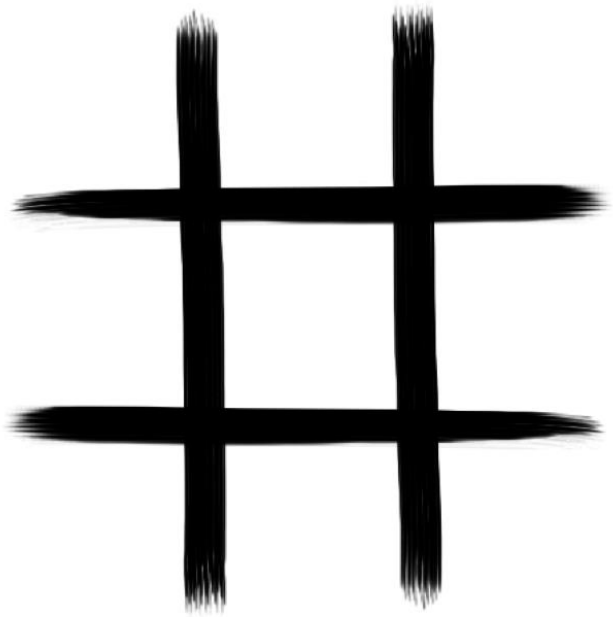
# Example of Reinforcement Learning

- How should a robot behave so as to optimize its “performance”? (Robotics)

- How to automate the motion of a helicopter? (Control Theory)

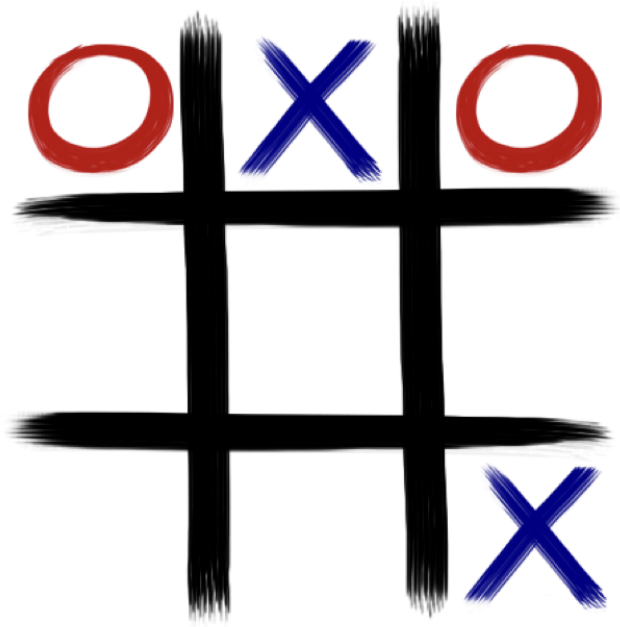
- How to make a good chess-playing program? (Artificial Intelligence)

# Example: Tic, Tac, Toe



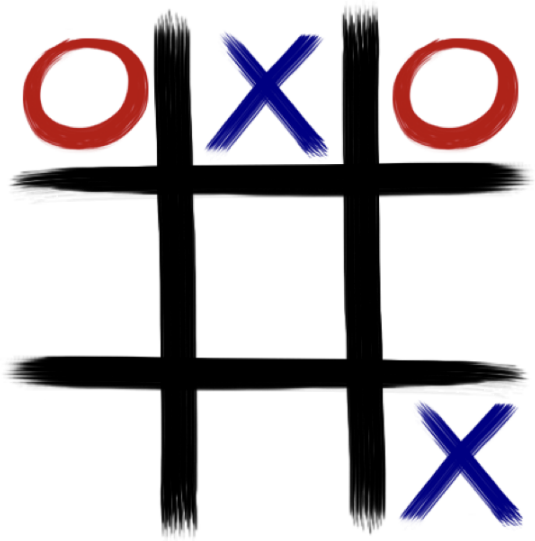
environment

# Example: Tic, Tac, Toe

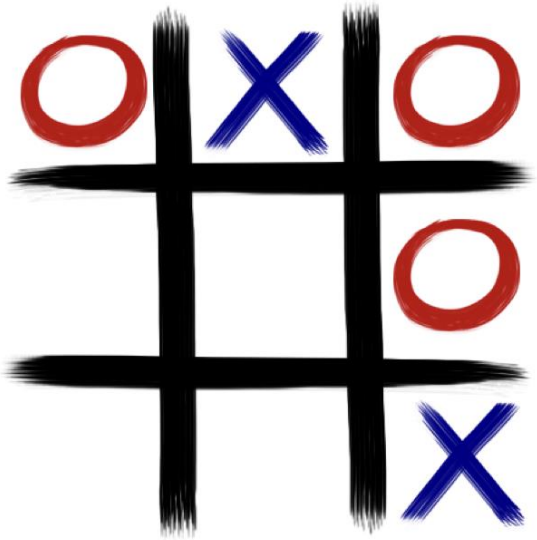


(current)
state

# Example: Tic, Tac, Toe

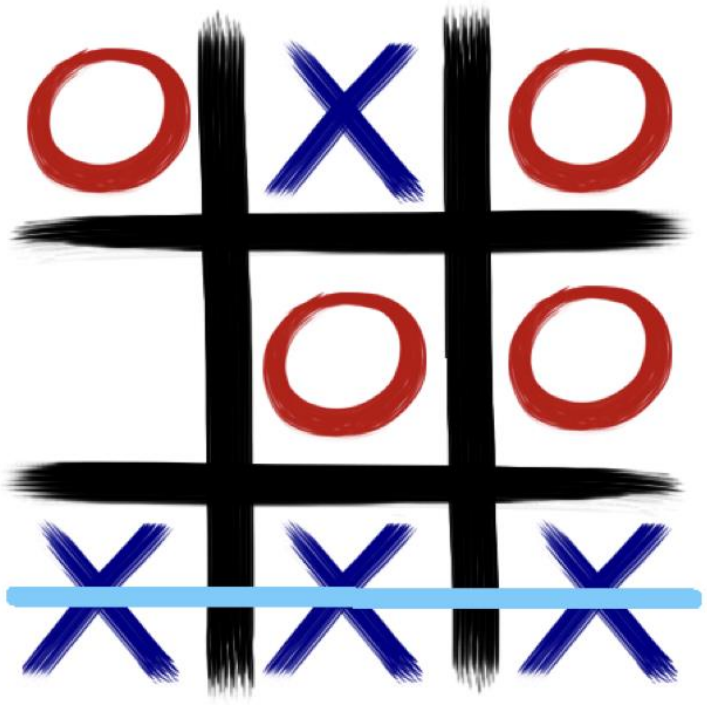


(current) state



action

# Example: Tic, Tac, Toe



reward

(here: -1)

# Example: Tic, Tac, Toe

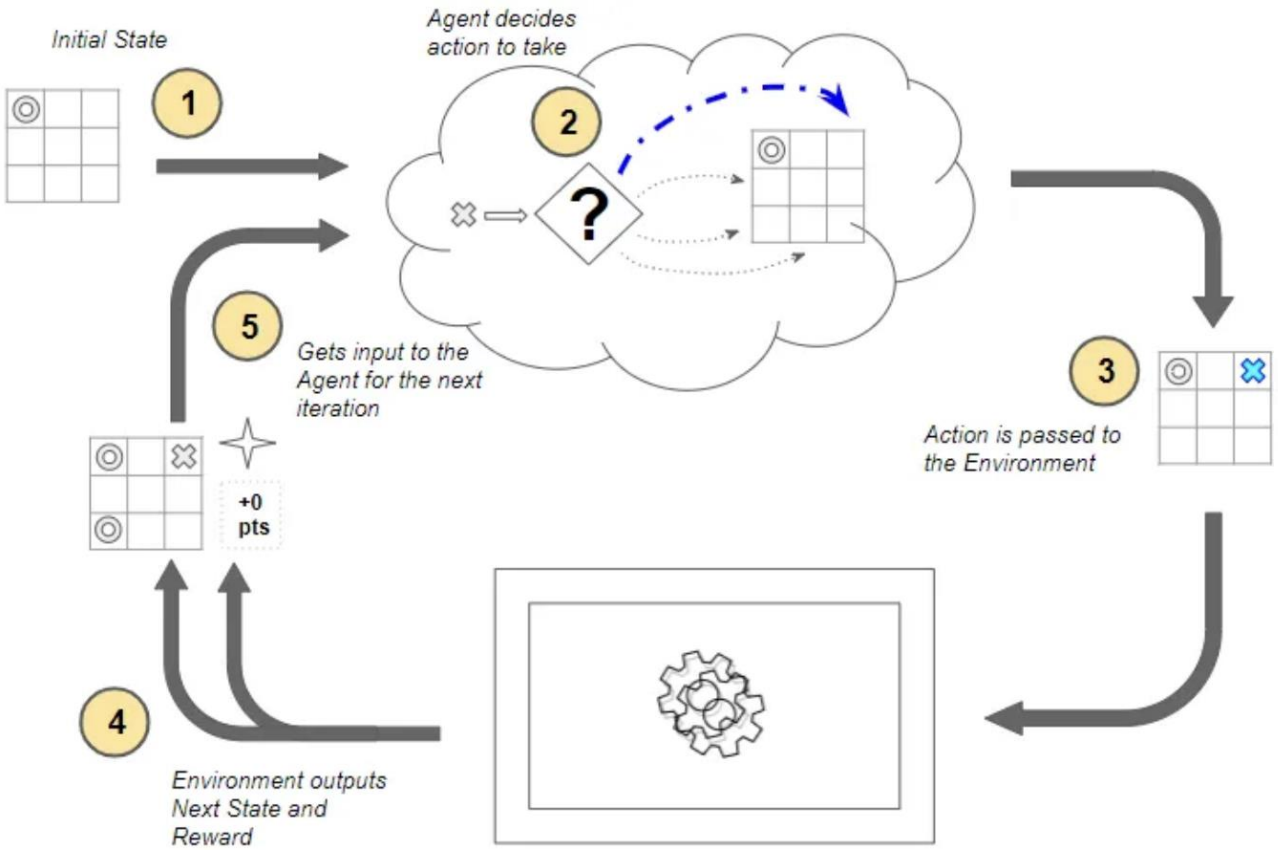
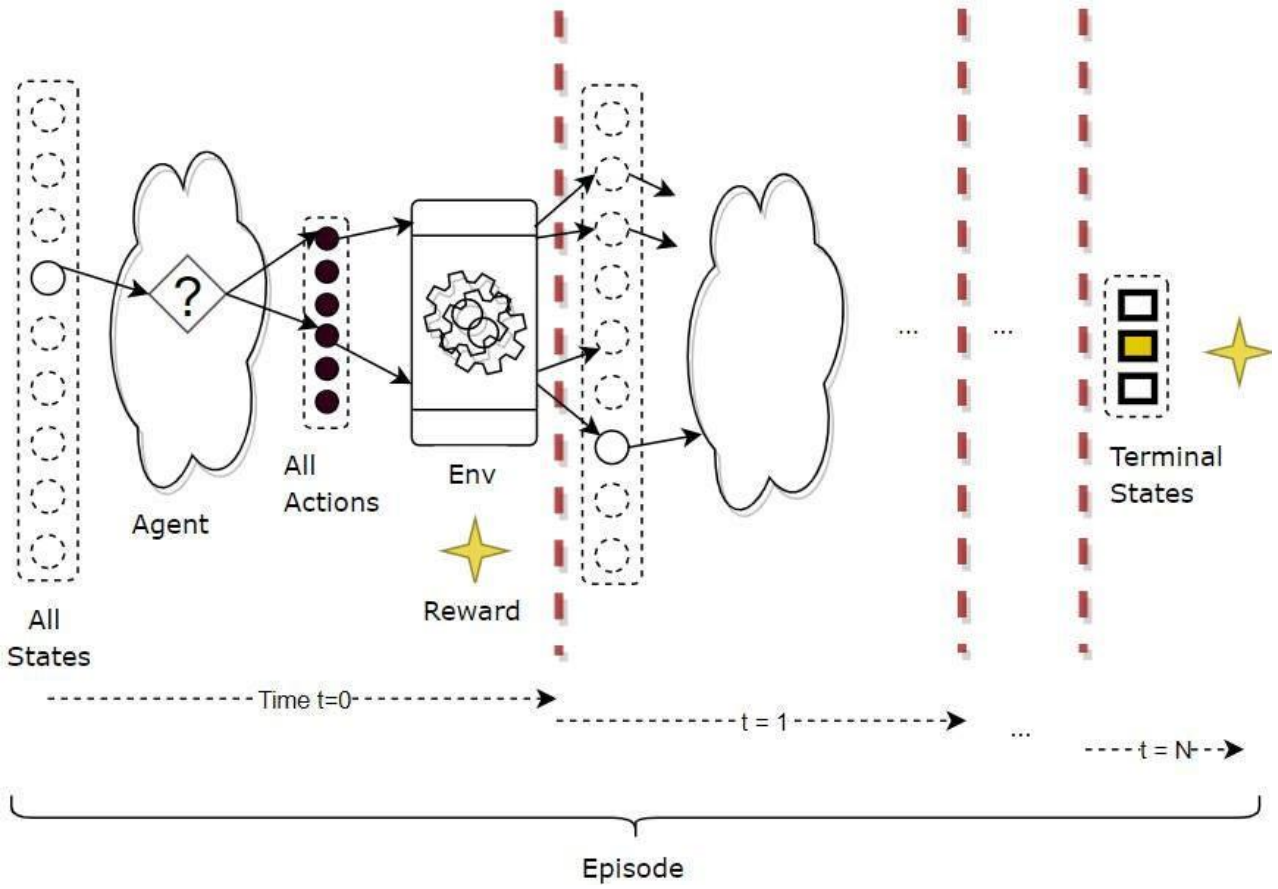
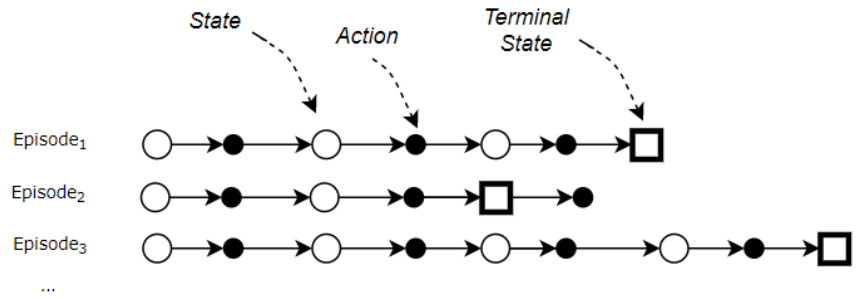



Image by Ketan Doshi

# MDP Time Steps




Each episode is independent of the next one.

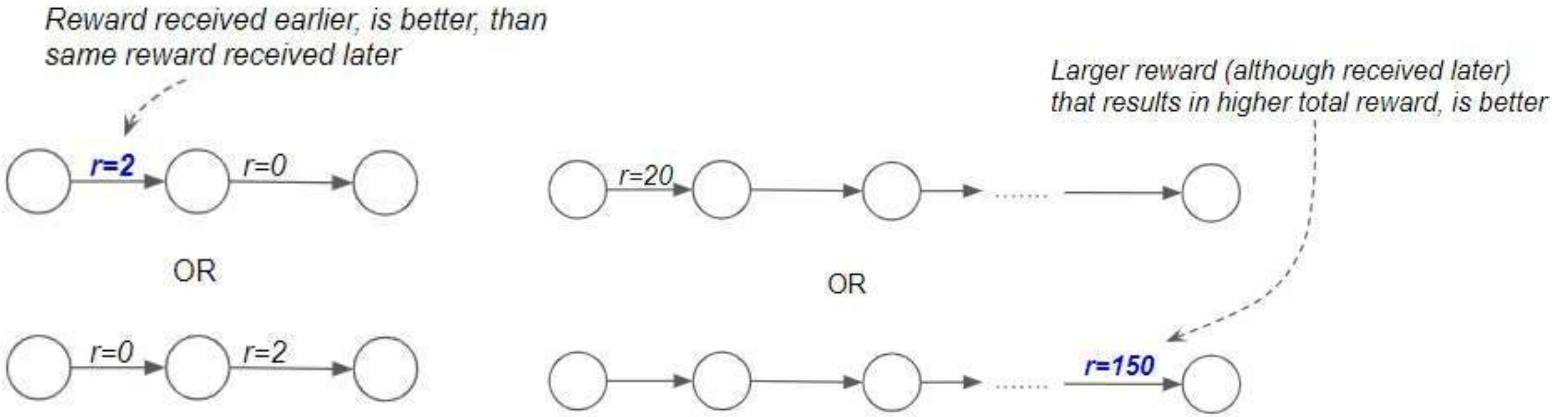
Image by Ketan Doshi

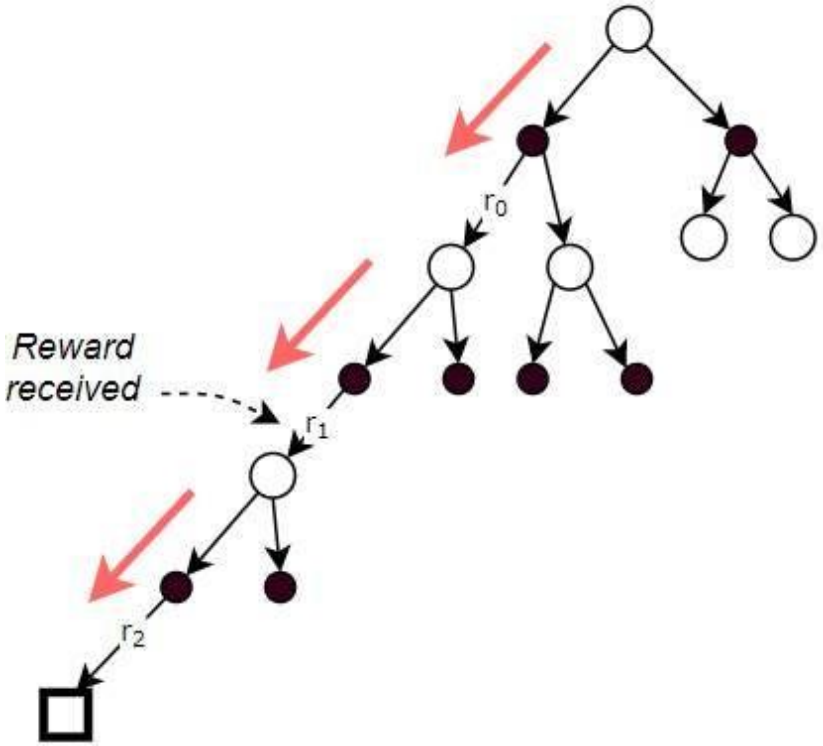
# Return : Total Reward

- The Return is the total of the rewards received at each time-step
- Immediate Reward is more valuable than Later Reward
- Rewards that give us the highest Total Returns are better

$$Return = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots + \gamma^n r_n$$

discount factor $\gamma$

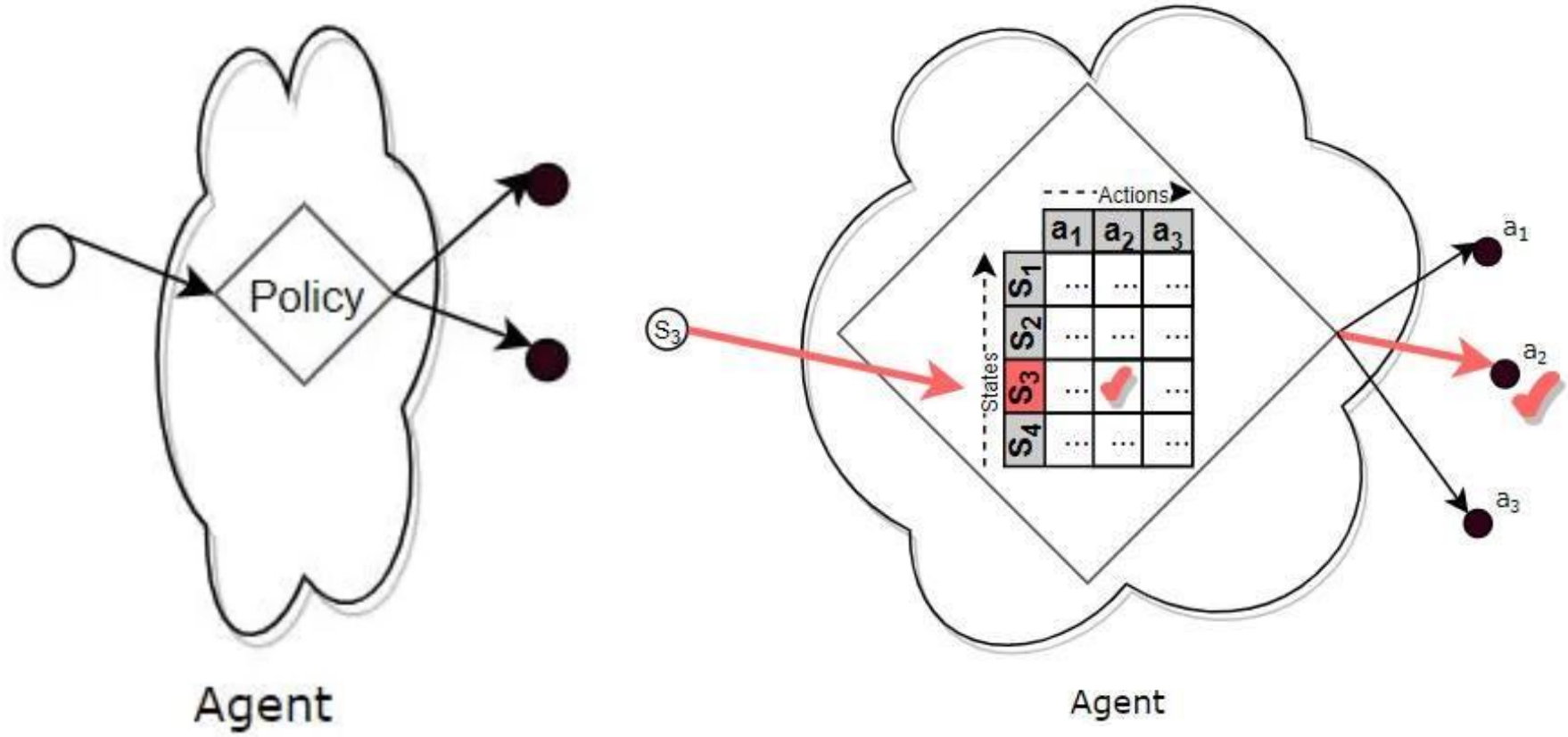



Images by Ketan Doshi

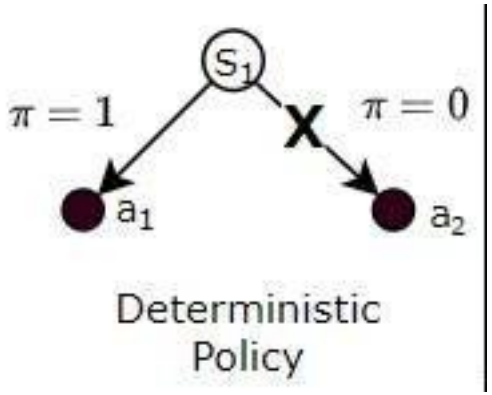
# Policy

The Policy tells the Agent which action to pick from any state

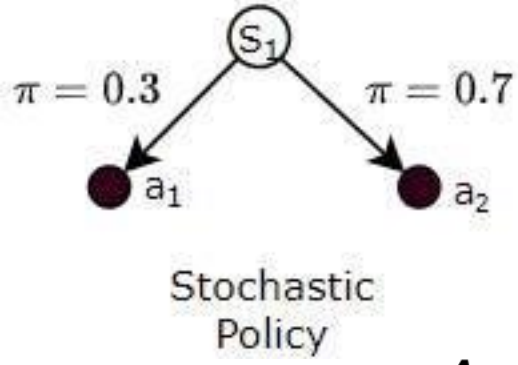


# Policy

- The policy tells the Agent which action to pick from any state : $a_t = \pi(s_t)$
- Deterministic Policy: the agent always chooses the same fixed action when it reaches a particular state
- Stochastic Policy: the agent varies the actions it chooses for a state, based on some probability for each action.



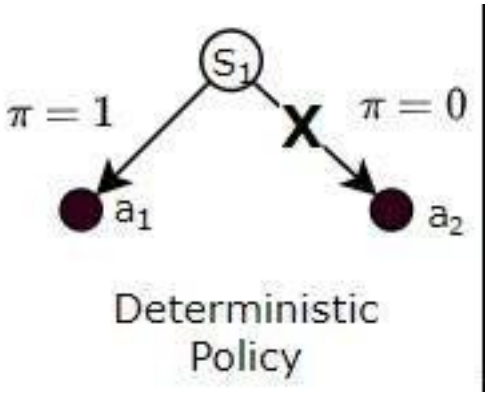

$a = \pi(s)$



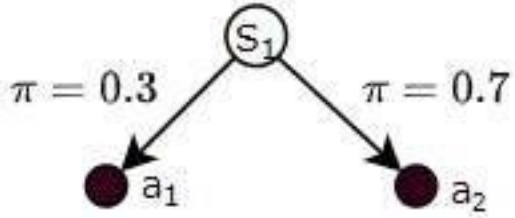

$A \sim \pi(\cdot|s)$

# Policy

- The policy tells the Agent which action to pick from any state
- Deterministic Policy: the agent always chooses the same fixed action when it reaches a particular state
- Stochastic Policy: the agent varies the actions it chooses for a state, based on some probability for each action.



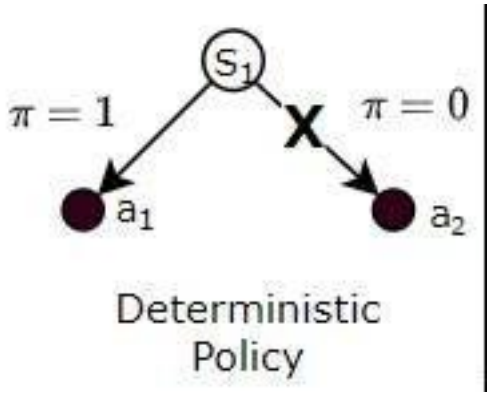
Deterministic Policy



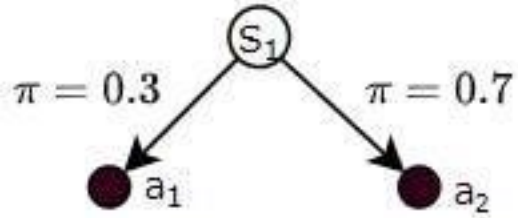
Stochastic Policy

# Value

- The Value is the expected future reward
- Deterministic Policy: the agent always chooses the same fixed action when it reaches a particular state
- Stochastic Policy: the agent varies the actions it chooses for a state, based on some probability for each action.



Deterministic Policy



Stochastic Policy

# Reinforcement Learning