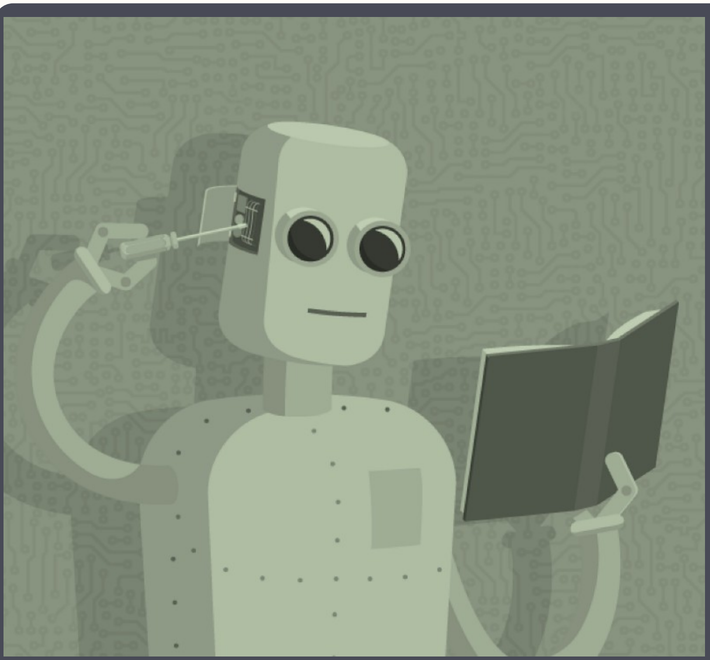
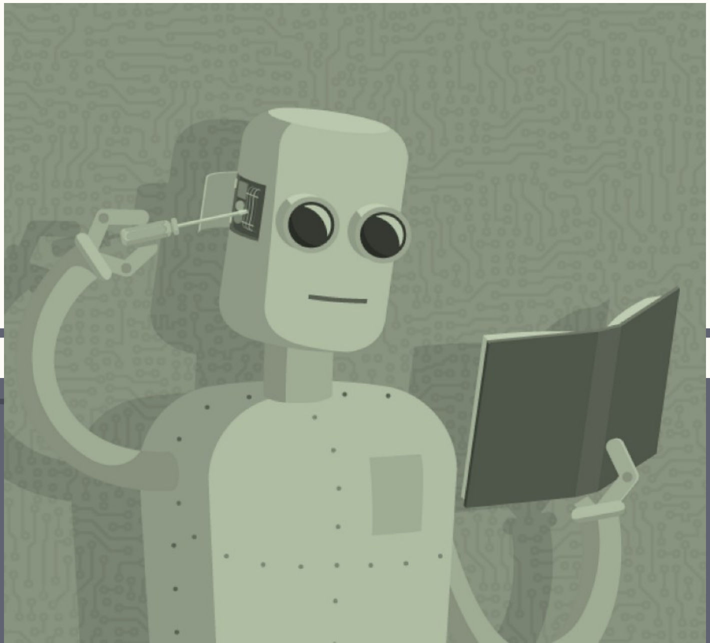


EE7209: Machine Learning (TE)

Dr. Prins

Department of Electrical & Information Engineering
Faculty of Engineering
University of Ruhuna

Reinforcement Learning (RL)



LMS Video Games Link – Group Activity

https://www.youtube.com/watch?v=NAf8uexaZ08

How would you learn to play the game?

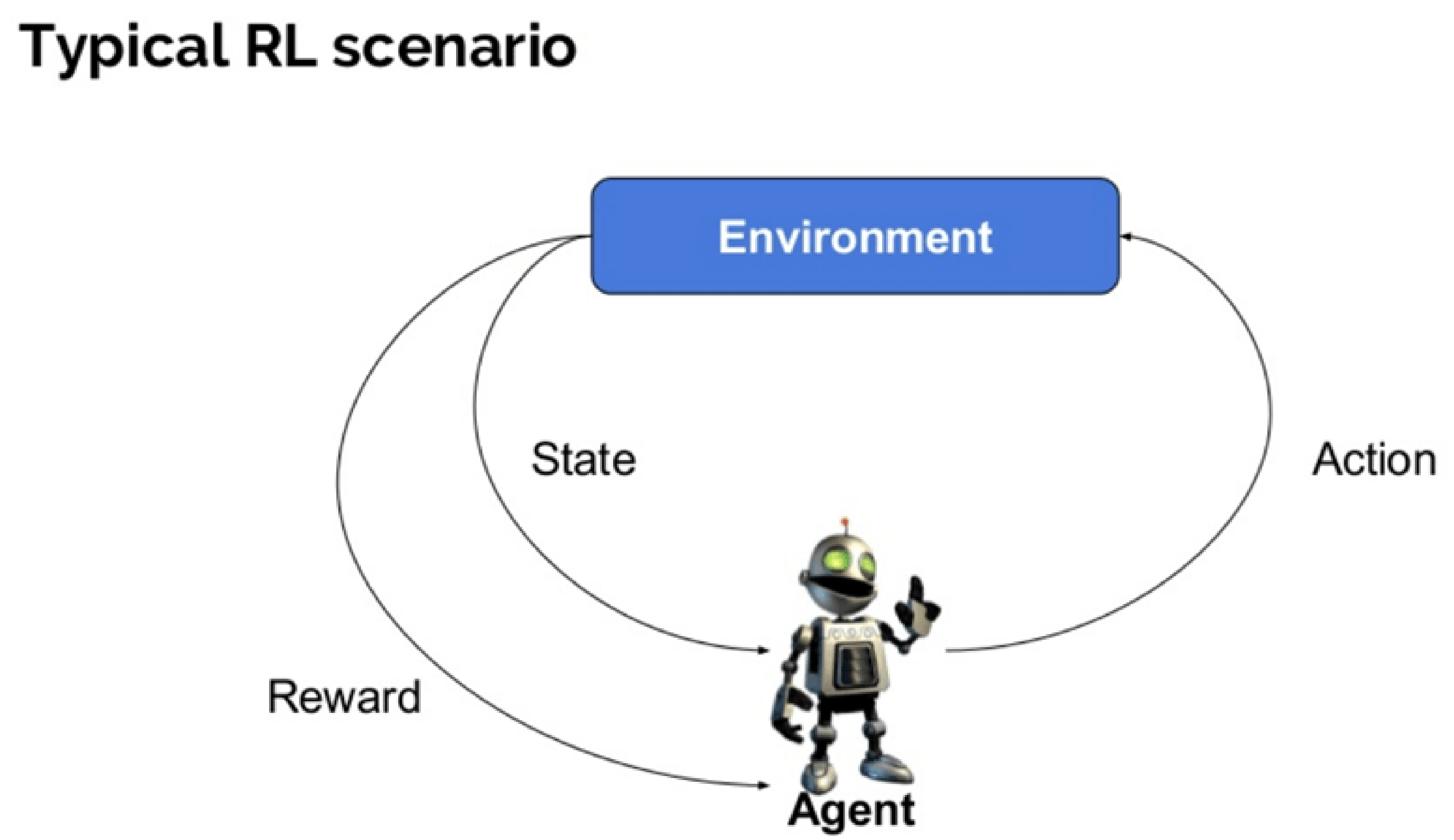
– Score
– Random key strokes
– Pleasant and unpleasant sounds
– Obstacles and bonus
– Trial and error
– More experience you get, the better, but longer it takes

How would you learn to play the game?

- Score *– value*
- Random key strokes *– action*
- Pleasant and unpleasant sounds *– reward*
- Obstacles and bonus *– policy/ strategy*
- Trial and error *– exploring*
- More experience you get, the better, but longer it takes *– interaction with environment*

prins@eie.ruh.ac.lk

How would you learn to play the game?



Terminology and key elements

- **Agent**: performs actions in an environment to gain some reward.
- Action (a): possible moves of the agent
- **Environment** (e): A scenario the agent has to face.
- State (s): Current situation of environment.
- **Reward** (R): An immediate return
- **Policy** (π): The strategy
- **Value** (V): The expected long-term return under π.
- Q-value or action-value (Q): long term value with the current action in mind.

Terminology

– **Agent: a hypothetical entity which performs actions in an environment to gain some reward.**

– **Action (a): All the possible moves that the agent can take.**

– **Environment (e): A scenario the agent has to face.**

– **State (s): Current situation returned by the environment.**

– **Reward (R): An immediate return sent back from the environment to evaluate the last action by the agent.**

Terminology

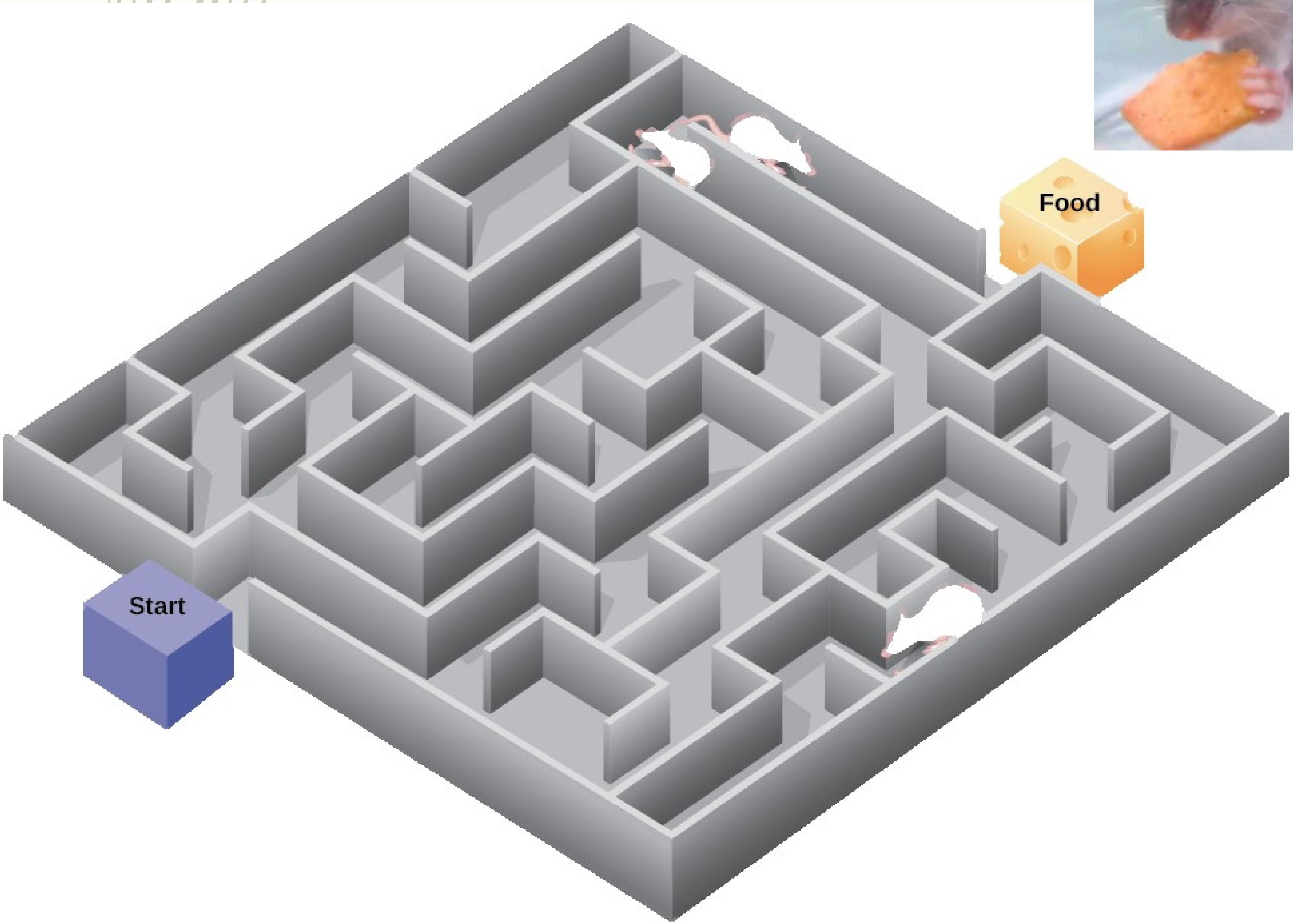
– Policy (π): The strategy that the agent employs to determine next action based on the current state.

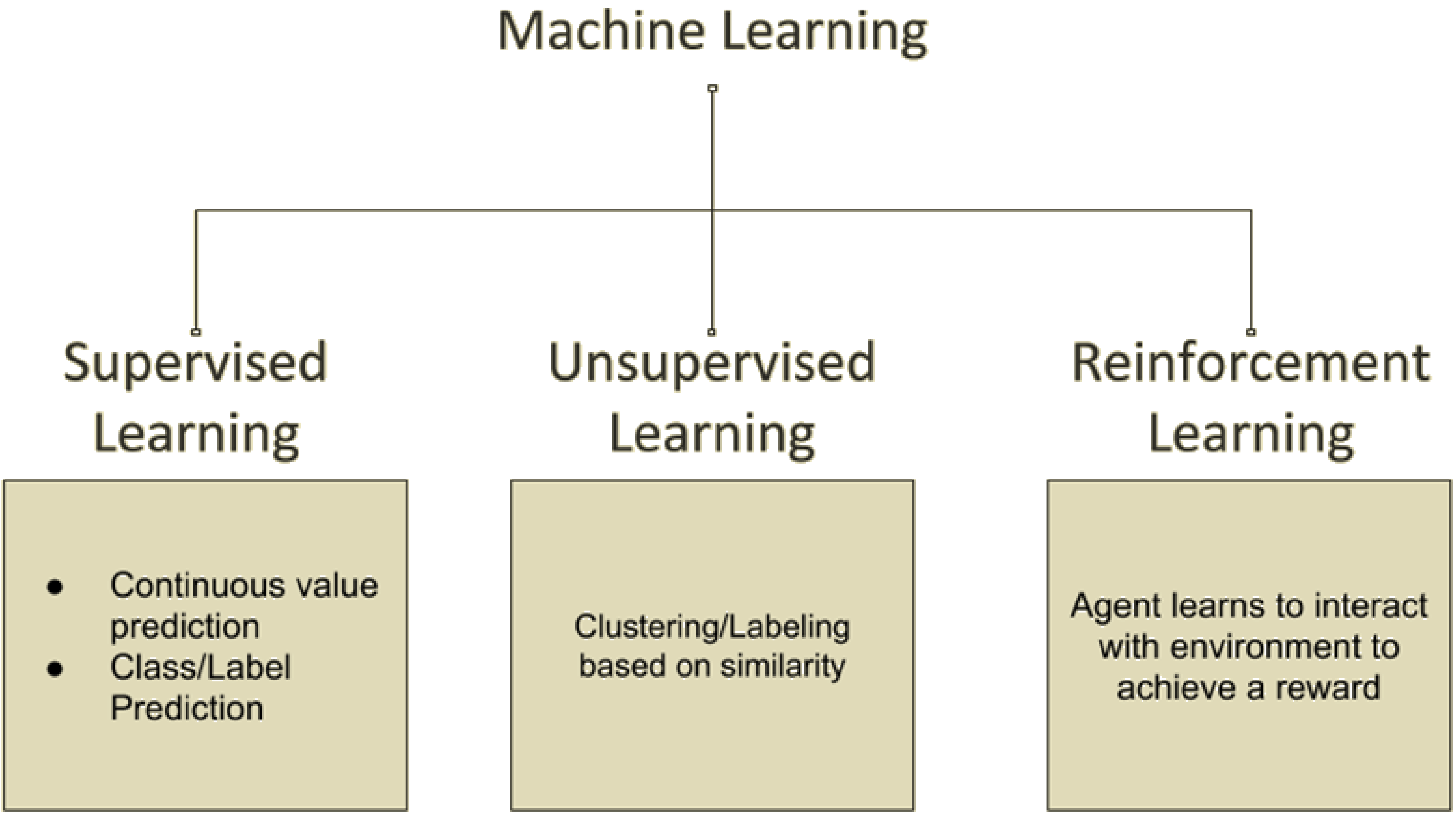
– Value (V): The expected long-term return with discount, as opposed to the short-term reward R. $V\pi(s)$, is defined as the expected long-term return of the current state s under policy π.

– Q-value or action-value (Q): Q-value is similar to Value, except that it takes an extra parameter, the current action a. $Q\pi(s, a)$ refers to the long-term return of the current state s, taking action a under policy π.

prins@eie.ruh.ac.lk

Reinforcement Learning





RL is defined by a learning problem, not an algorithm

prins@eie.ruh.ac.lk

RL decision making

1. Credit Assignment Problem
   - What did I do right/wrong in the process?
2. Exploration vs Exploitation
3. Learning Models
   - MDP - Markov Decision Processes (Sequential, discrete time steps)
   - Q-learning (value based)
4. DP (Dynamic Programming)
   - A collection of algorithms that can be used to compute optimal policies

Architecture

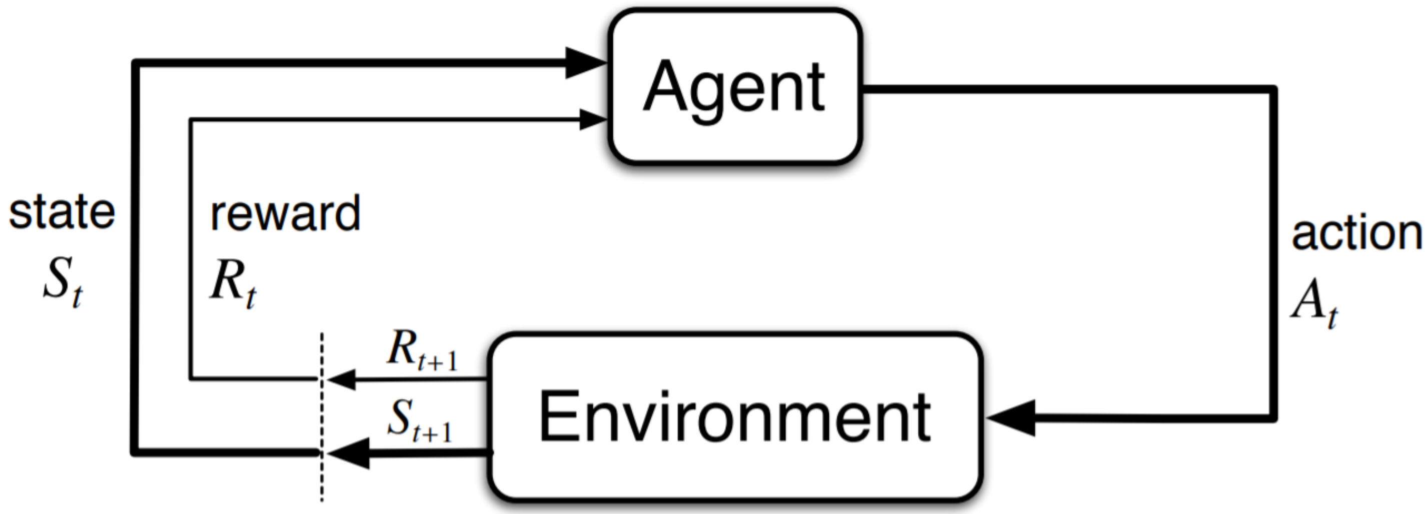


Figure 3.1: The agent–environment interaction in a Markov decision process.

–Sequential, discrete time steps

Bellman optimality equation

$$(Q)_t = (1-\alpha)(Q)_t + \alpha[(R)_t + \gamma \max(Q)_{t+1}]$$

Example

Room number 2 to 5

– Initial state = state 2

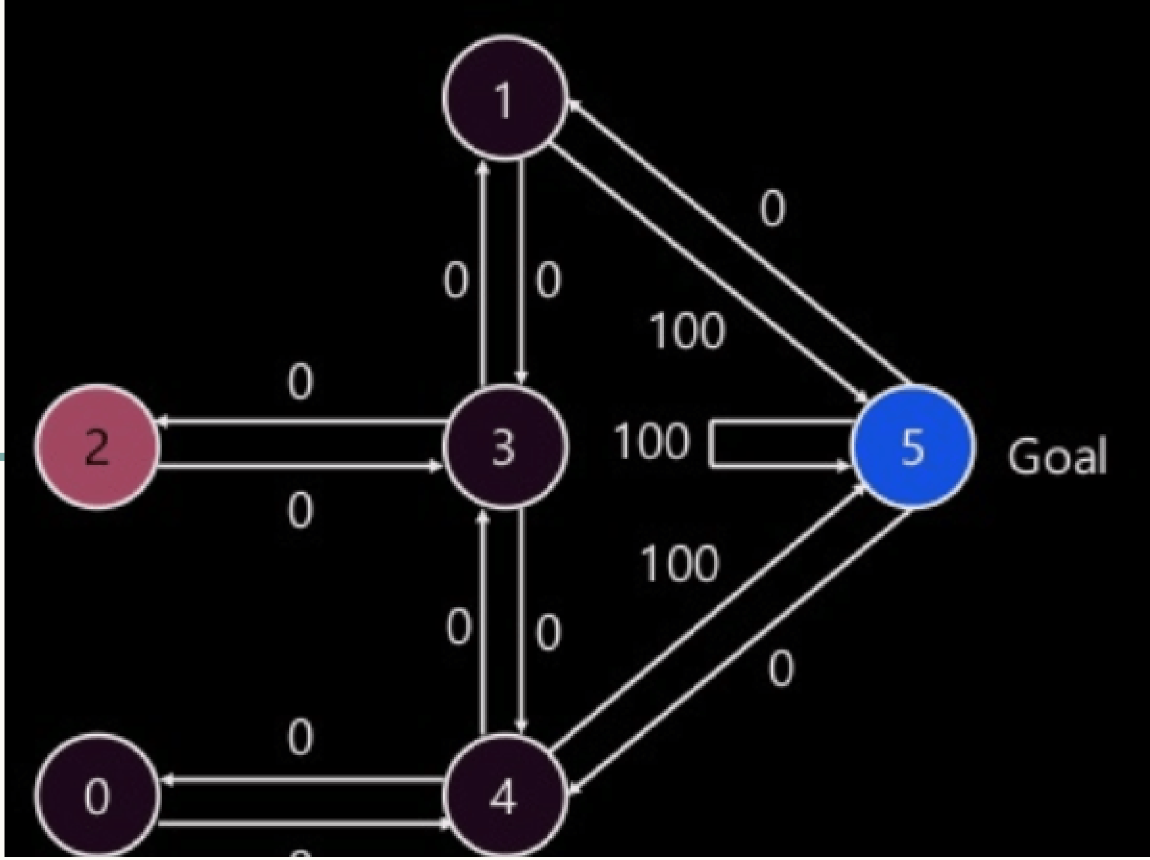
– State 2-> state 3

– State 3 -> state (2,1,4)

– State 4-> state (0,5,3)

– State 1-> state (5,3)

– State 0-> state 4



Approaches to implement RL

https://www.youtube.com/watch?v=9JZID-h6ZJ0

1. Value-based
   - Maximize the value function V(s) defined by long term return of the current state s under policy π
2. Policy-based
   - Come up with a policy
   - eg greedy policy – action with highest value
   - Policy π determines the next action a at any state s.
   - *Deterministic* – same s, same a with policy π
   - *Stochastic* – probability based
3. Model-based
   - Create a virtual model for each environment
   - Model differs for each environment

Value-based: "Where is the best place to be?"

Policy-based: "What should I do next?"

Model-based: "Let me predict the future before moving!"

prins@eie.ruh.ac.lk

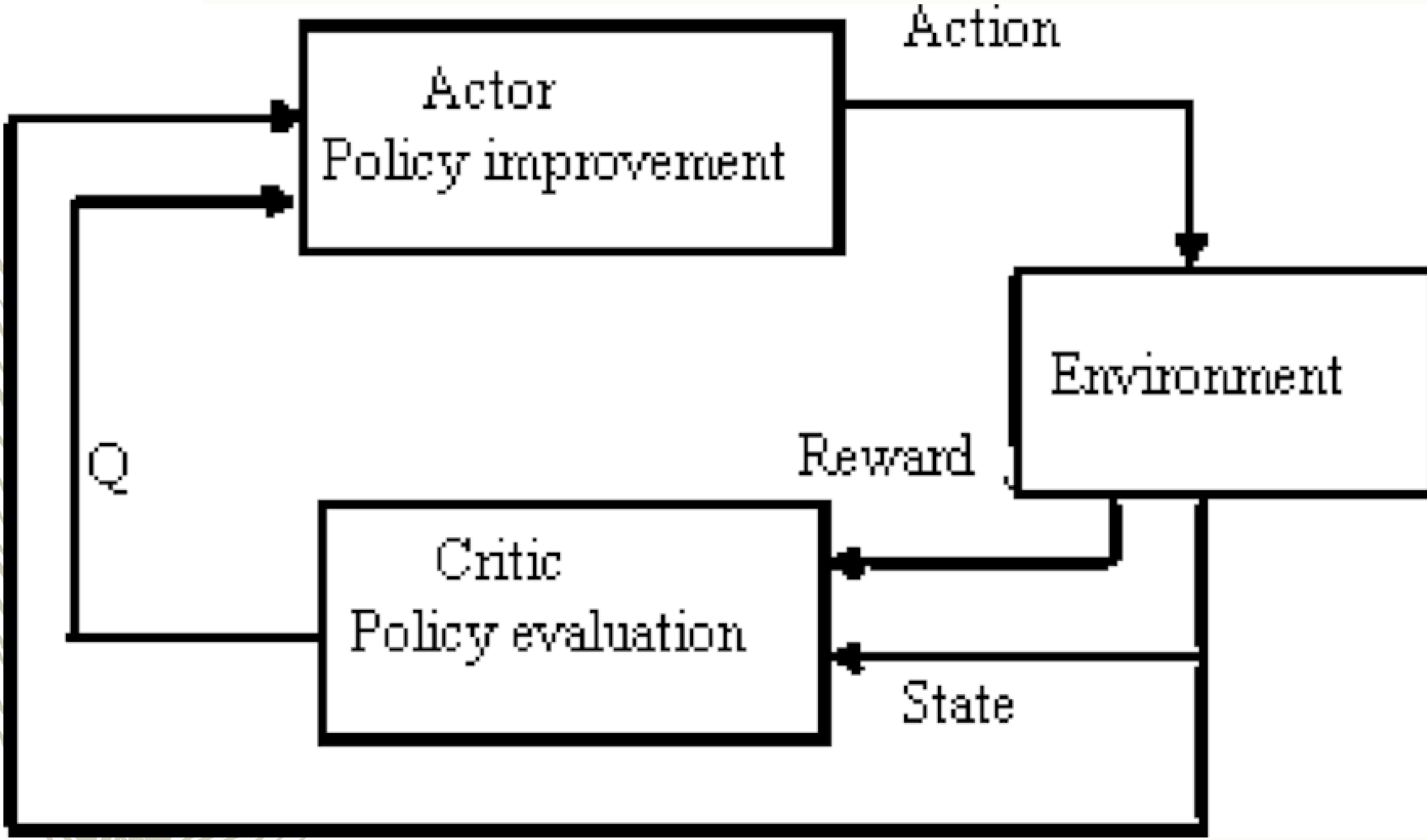
Reinforcement Learning

– Adaptive process

– Uses previous experience

– Improve the outcomes of future choices

-Critic RL Architecture

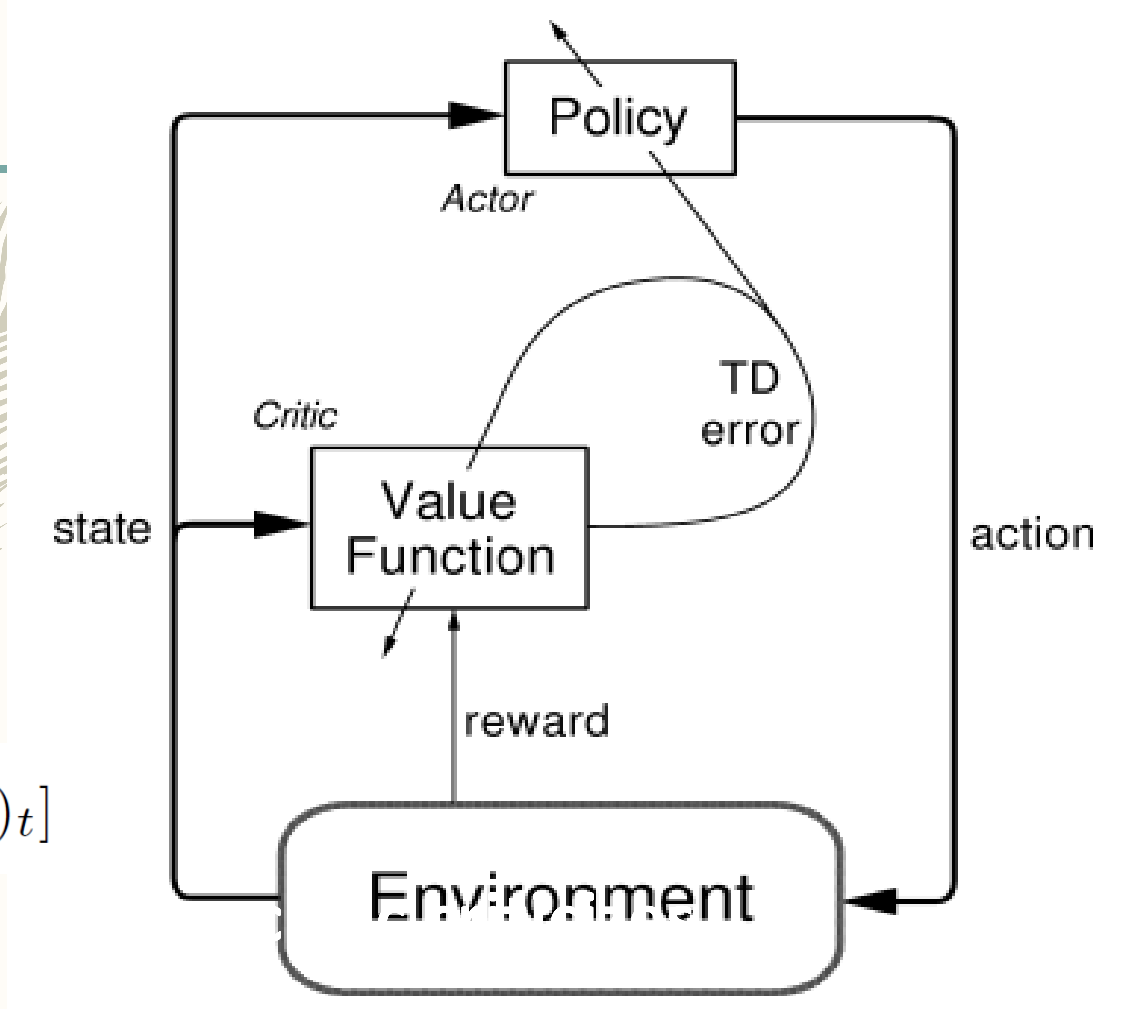


tor takes the Bellman expectation equation

$$(Q)_t = E[(R)_t + \gamma(Q)_{t+1}]$$

Temporal difference equation

$$(\delta)_t = E[(R)_t + \gamma(Q)_{t+1}] - (Q)_t$$

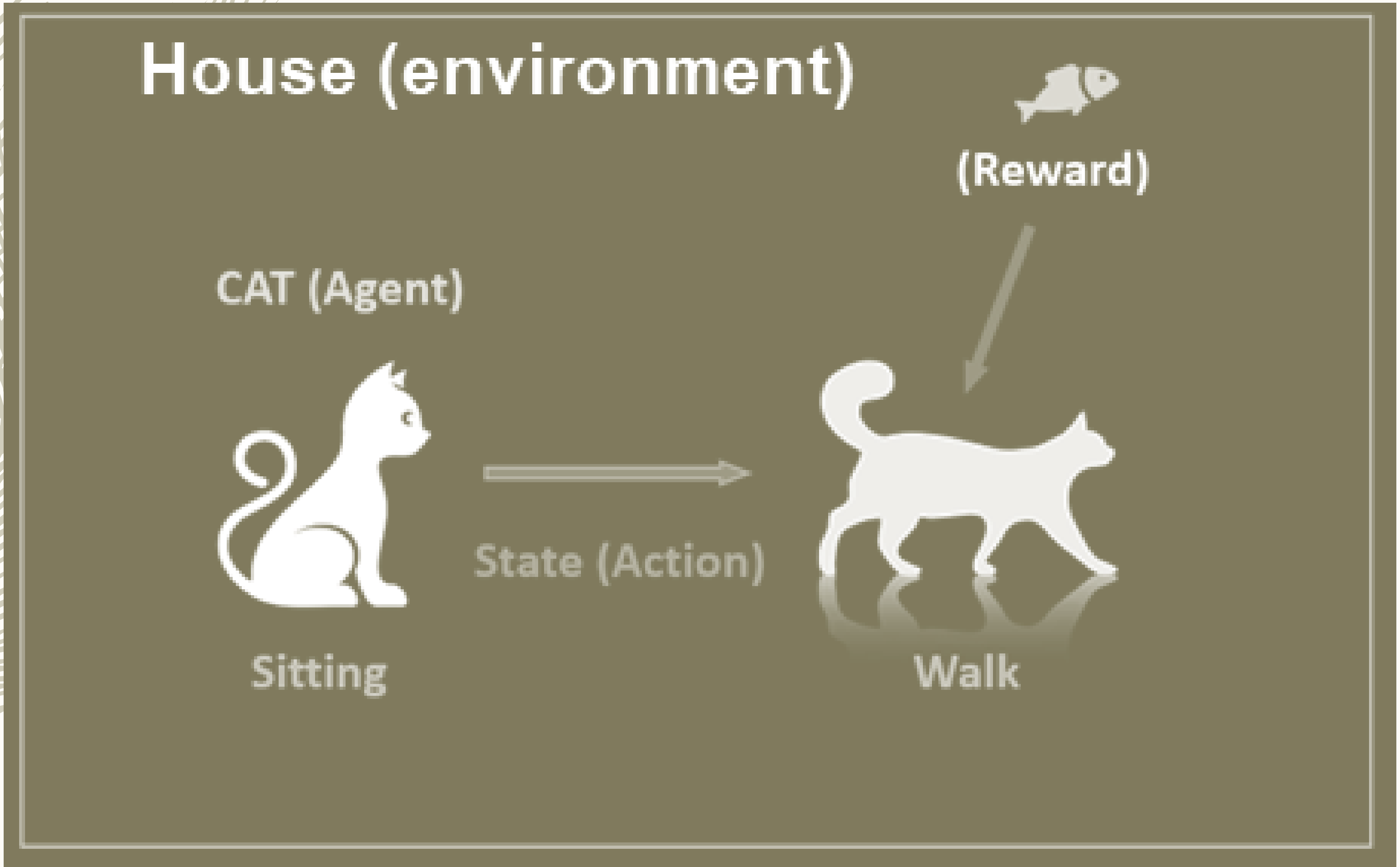


$$(Q)_t = (Q)_t + \alpha[(\delta)_t]$$

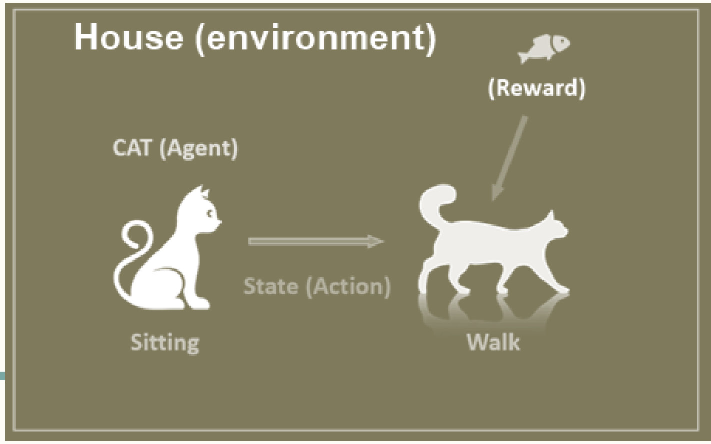
prins@eie.ruh.ac.lk

Activity

- Based on RL, teach your cat how to play a new trick
- Problem(s): Cat doesn't understand English/ Sinhala/ Tamil



Teaching a Cat new tricks



–RL

- Environment: Simulate different scenarios
- Cat responds in many ways – one of which will be rewarded a fish (positive reinforcement)
- Cat repeats the action that got him the fish
- Negative reinforcement to learn what not to do

Parameters	Reinforcement Learning	Supervised Learning
Decision style	reinforcement learning helps you to take your decisions sequentially.	In this method, a decision is made on the input given at the beginning.
Works on	Works on interacting with the environment.	Works on examples or given sample data.
Dependency on decision	In RL method learning decision is dependent. Therefore, you should give labels to all the dependent decisions.	Supervised learning the decisions which are independent of each other, so labels are given for every decision.
Best suited	Supports and work better in AI, where human interaction is prevalent.	It is mostly operated with an interactive software system or applications.
Example	Chess game	Object recognition

Applications of RL

- Autonomous flying/ driving
- Pole balancing
- Game Theory/ Multi-Agent Interaction
- AI, Robotics
- AlphaGo
  - Board games
  - RL, DL, Trees
- Industrial Logistics
- Business strategy planning

prins@eie.ruh.ac.lk

Why Reinforcement Learning?

https://www.youtube.com/watch?v=9JZID-h6ZJ0

- Which situation needs an action
- Which action yields the highest reward over the longer period.
- Provides the learning agent with a reward function.
- Allows it to figure out the best method for obtaining large rewards.

When Not to Use RL?

- When you have enough data to solve the problem with a supervised learning method
- RL is computing-heavy and time-consuming - in particular when the action space is large.

Challenges in RL

- Feature/reward design which should be very involved

You have to carefully design the rewards and features (inputs) for the system to learn properly

- Parameters may affect the speed of learning.

Parameters Can Slow Learning:
Things like learning rate, discount factor, etc., control how fast or slow the agent learns.

- Realistic environments can have partial observability.

Partial Observability:
In real life, you often don't have full information about the environment.

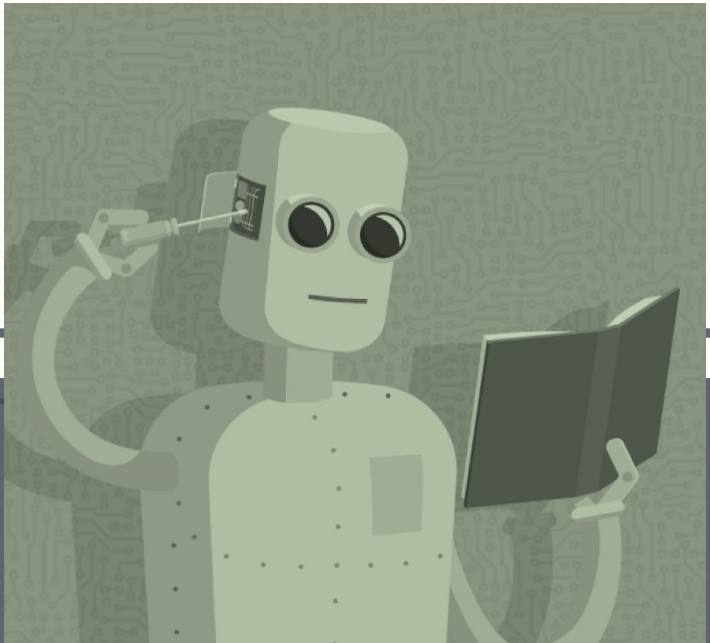
- Too much Reinforcement may lead to an overload of states which can diminish the results.

Too Much Reinforcement = Overload:
If you give feedback (rewards/punishments) too often, the agent may experience too many states and struggle to learn clear strategies

- Realistic environments can be non-stationary.

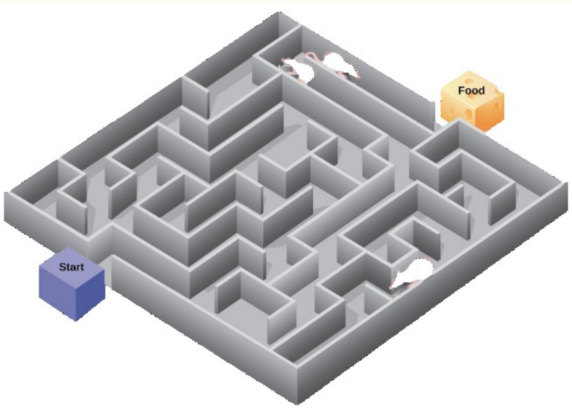
Realistic Environments are Non-Stationary:
In real-world problems, the "rules of the world" can change over time.

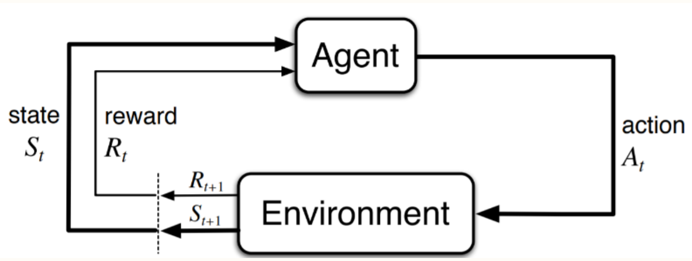
Summary



Summary

- What is RL
- Key elements
  1. Agent
  2. Environment (e)
  3. Reward (R)
  4. Policy (π)
  5. Value (V)
- Architecture
- Approaches
  1. Value
  2. Policy
  3. Model
- Differences from SL
- Applications
- Challenges
- When to choose and when not to choose





prins@eie.ruh.ac.lk

Books (available online)

- Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

- Szepesvári, Csaba. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning* 4, no. 1 (2010): 1-103.

Thank you!