Lecture 1.0

Reinforcement Learning: An Introduction

Alexey Gruzdev
alexey.s.gruzdev@gmail.com
HSE, Winter 2019

RL Literature

- **An Introduction to Reinforcement Learning, Sutton and Barto, 1998**

  Available free online!

  last update: March 21, 2018

  http://incompleteideas.net/book/bookdraft2018mar21.pdf

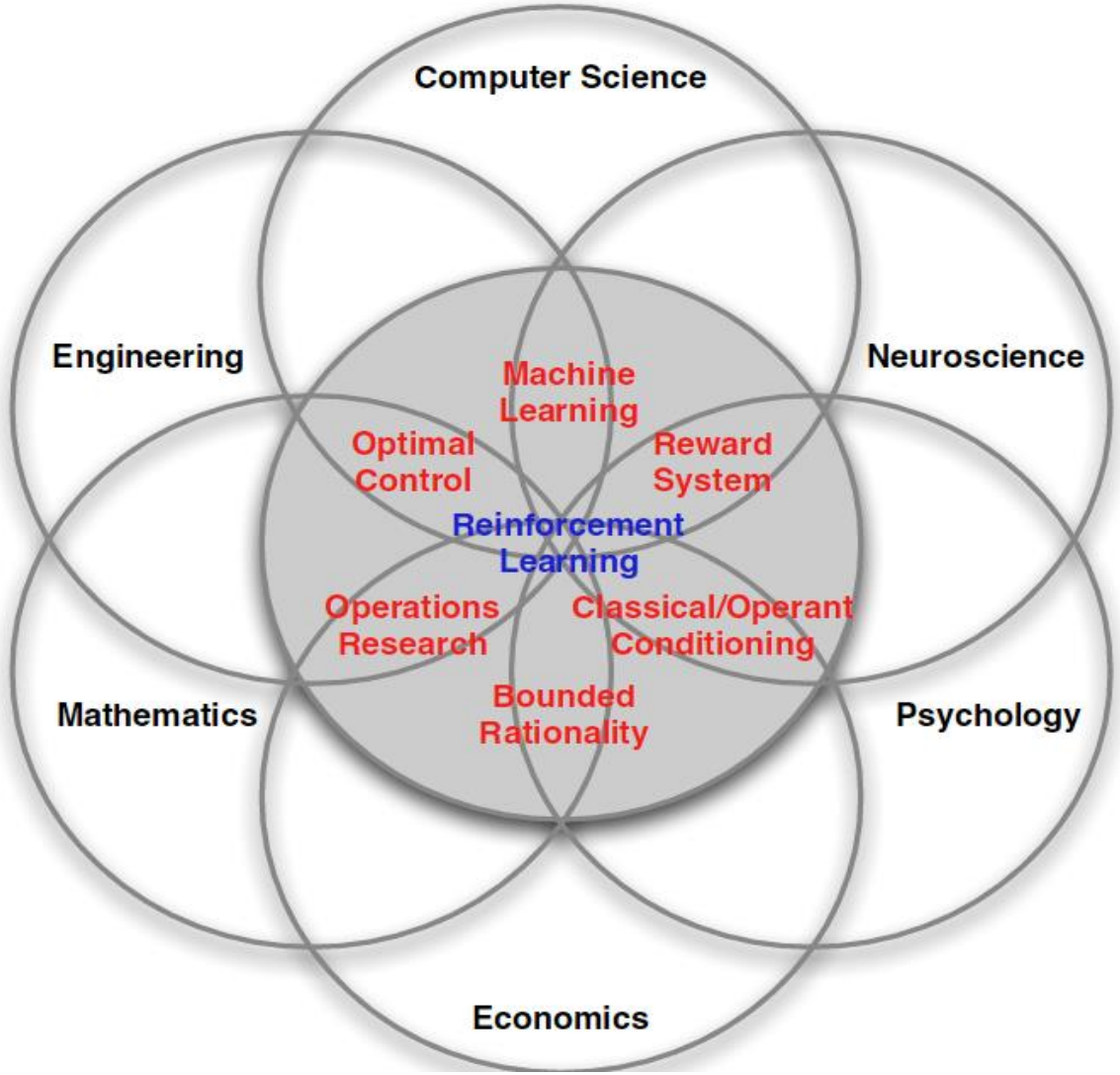
- **Algorithms for Reinforcement Learning, Szepesvari, Morgan and Claypool, 2010**

  Available free online!

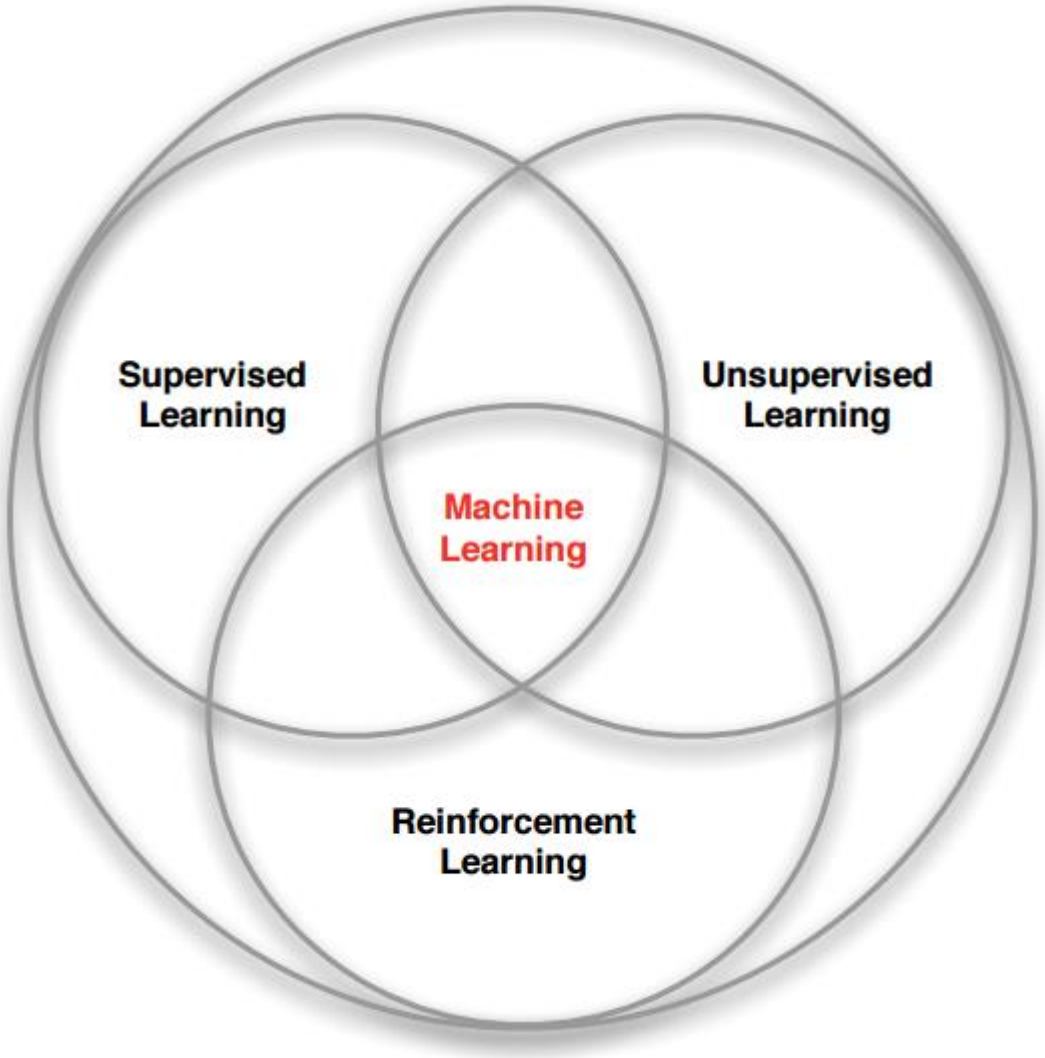
  last update: July 8, 2017

  https://sites.ualberta.ca/~szepesva/papers/RLAlgsInMDPs.pdf

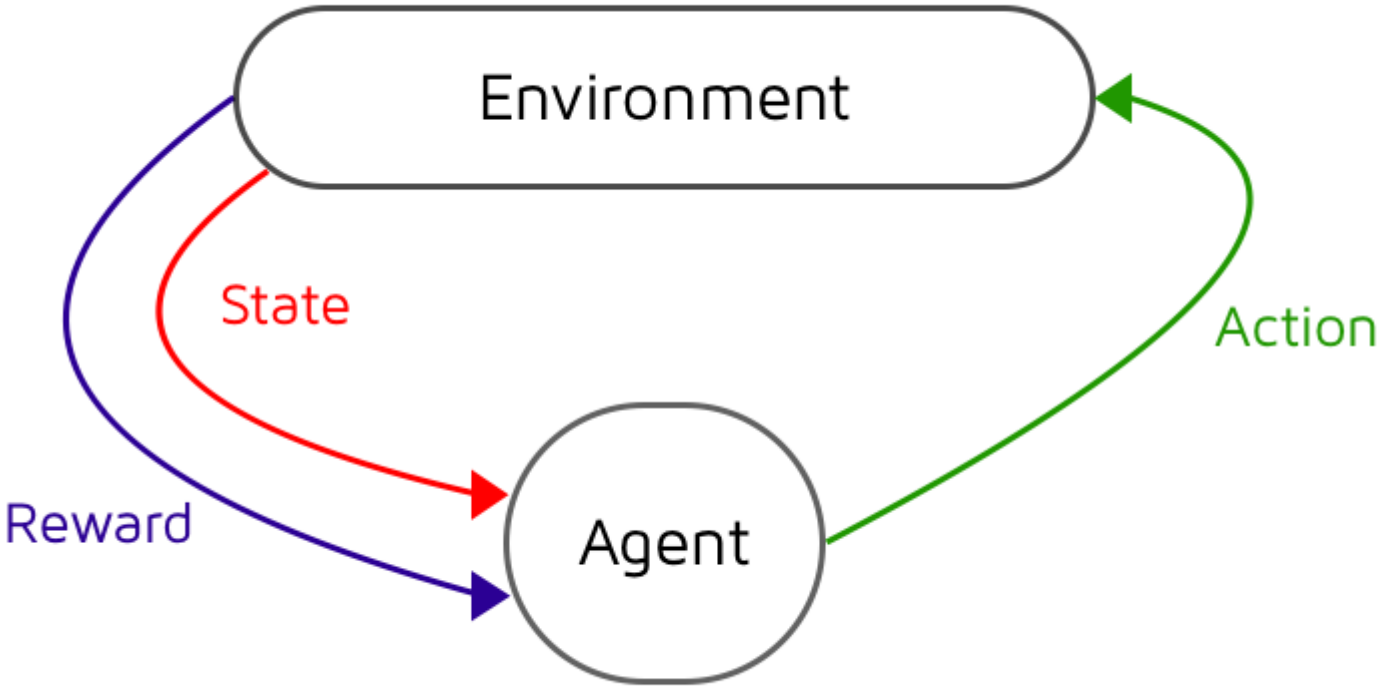
Many faces of Reinforcement Learning



Branches of Machine Learning



RL Mechanics



Reinforcement Learning Peculiarities

- What makes RL domain different from other machine learning approaches?
  - There is **no** external supervisor, only *reward* signal
  - Feedback can be delayed – not right now!
  - Time matters – sequential, not i.i.d data
  - Agent's actions affect future data it receives

Examples of Reinforcement Learning

- Fly stunt maneuvers in a quad copter / helicopter
- Derive 'optimal' neural network architecture
- Defeat the world's champion in Go
- Manage an investment portfolio
- Make a Humanoid robot walk
- Control a power station
- Play Atari games

Helicopter Maneuvers

- https://www.youtube.com/watch?v=VCdxqnofcnE

Playing Atari games

- https://www.youtube.com/watch?v=TmPfTpjtdgg
- https://www.youtube.com/watch?v=W2CAghUiofY

Rewards

- A reward R_t is a scalar feedback signal
- Indicates how well agent is doing at step *t*
- The agent's job is to maximize cumulative reward

Reinforcement Learning is based on the reward hypothesis.

<u>The reward hypothesis:</u>

All goals can be described by the maximization of expected cumulative reward .

Your turn - Is that correct or not?

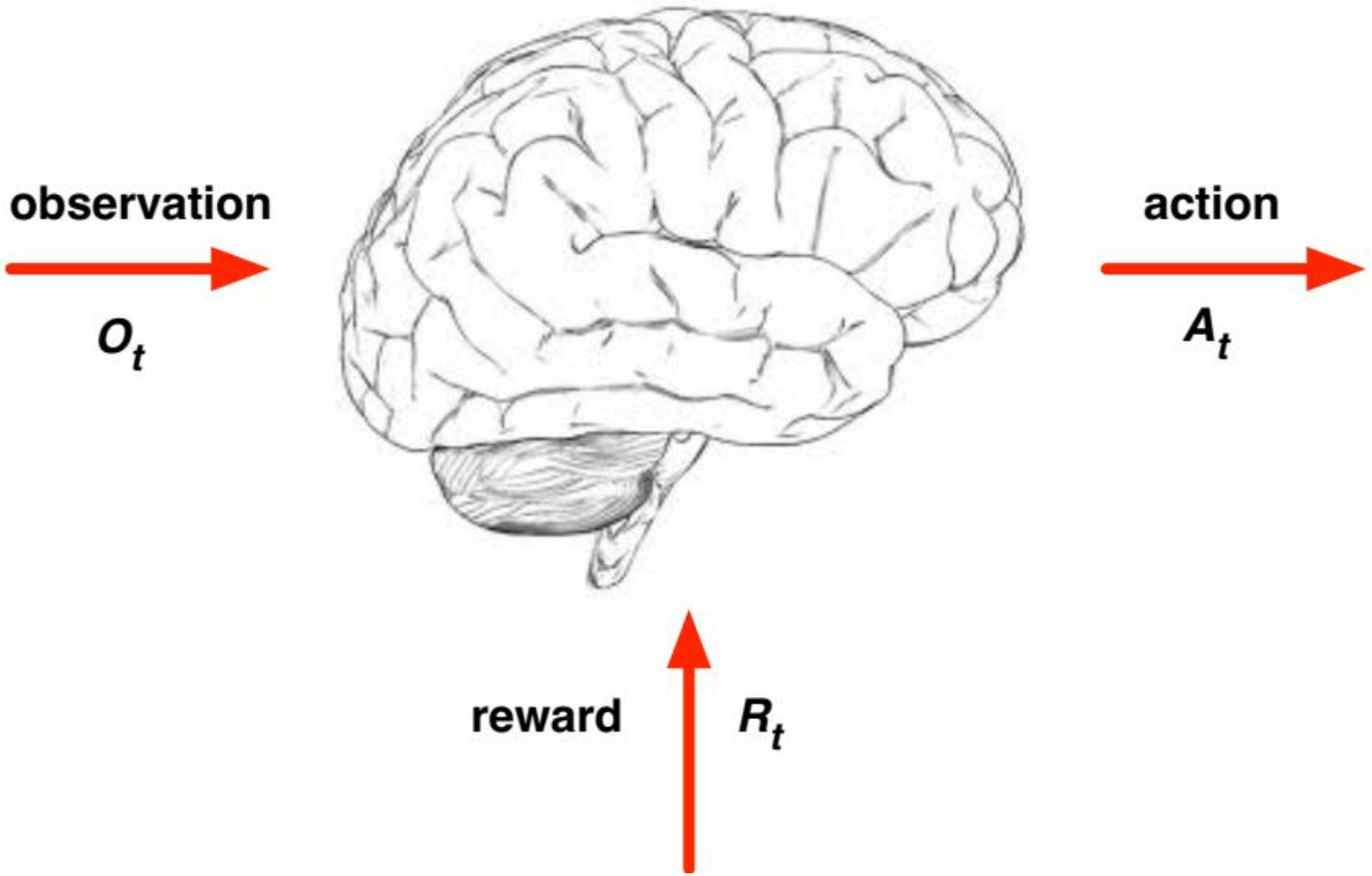
Examples of Rewards

- Fly stunt maneuvers in a quad copter / helicopter
  + reward for following desired trajectory
  - reward for crashing
- Derive 'optimal' neural network architecture
  + reward for obtaining small & accurate neural network
- Defeat the world champion in Go
  +/- reward for winning/losing a game
- Manage an investment portfolio
  + reward for each $ in bank
- Make a humanoid robot walk
  + reward for forward motion
  - reward for falling over
- Control a power station
  + reward for producing power
  - reward for exceeding safety thresholds
- Play many different Atari games better than humans
  +/- reward for increasing/decreasing game's score

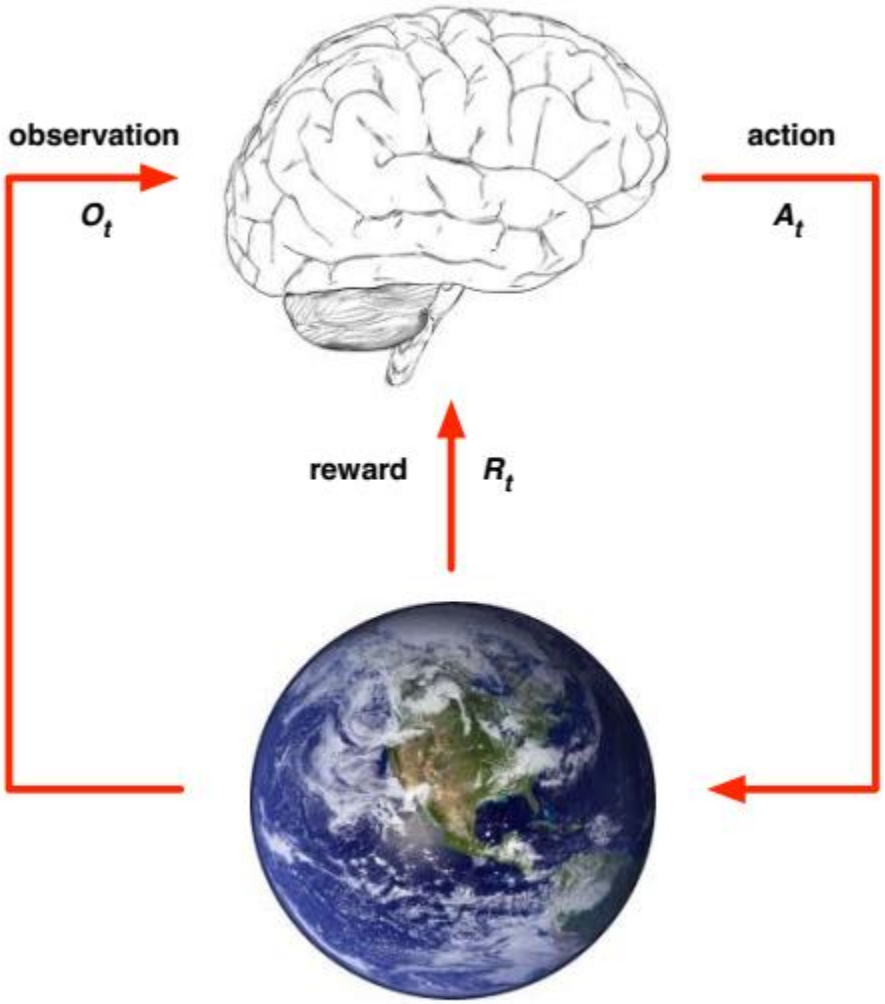
Sequential Decision Making

- Goal: *select actions to maximize total future reward*
- Actions may have long term consequences
- Reward may be delayed
- It may be better to sacrifice immediate reward to gain more long-term reward
  - A financial investment (may take months to mature)
  - Refueling a helicopter (might prevent a crash in several hours)
  - Blocking opponent moves (might help winning chances many moves from now)

Agent & Environment



Agent & Environment



- At each step t the agent:
  - Executes action A_t
  - Receives observation O_t
  - Receives scalar reward R_t
- The environment:
  - Receives action A_t
  - Emits observation O_{t+1}
  - Emits scalar reward R_{t+1}
- t increments at env. step

History and State

- The history is the sequence of observations, actions, rewards

$$H_t = O_1, R_1, A_1, \ldots, A_{t-1}, O_t, R_t$$

- i.e. all observable variables up to time t
- i.e. the sensorimotor stream of a robot or embodied agent

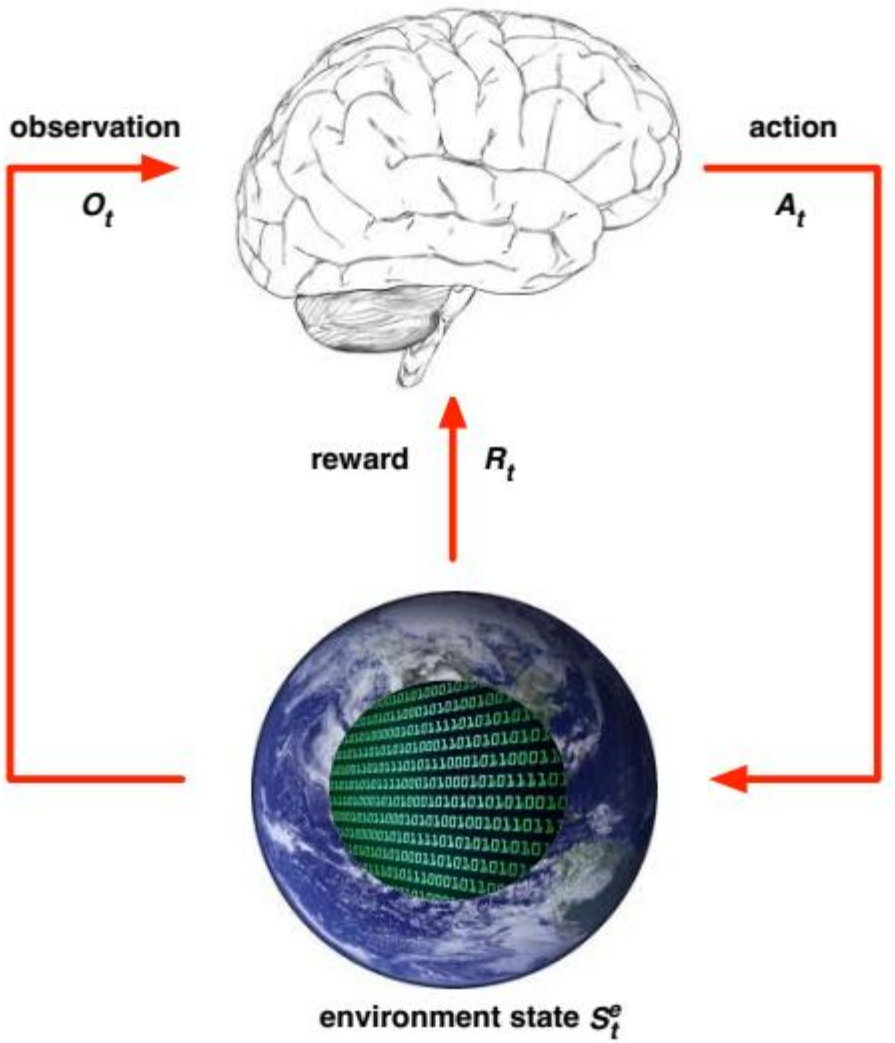
- What happens next depends on the history:
  - The agent selects actions
  - The environment selects observations/rewards

- State is the information used to determine what happens next

- Formally, state is a function of the history:

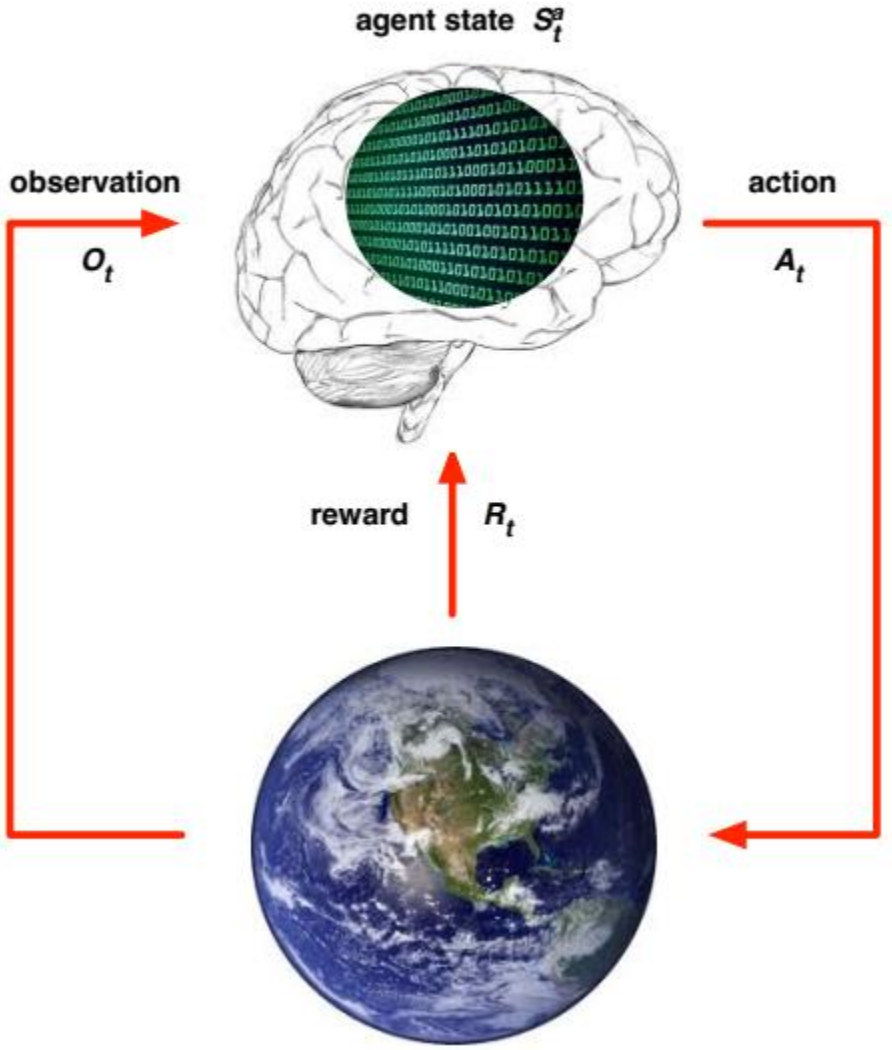
$$S_t = f(H_t)$$

Environment State



- The environment state S_t^e is the environment's private representation
- i.e. whatever data the environment uses to pick the next observation/reward
- The environment state is not usually visible to the agent
- Even if S_t^e is visible, it may contain irrelevant information

Agent State



- The agent state S_t^a is the agent's internal representation
- i.e. whatever information the agent uses to pick the next action
- i.e. it is the information used by reinforcement learning algorithms
- It can be any function of history:

$$S_a = f(H_t)$$

Markov State

- An information state (a.k.a. Markov state) contains all useful information from the history.

- Definition: a state S_t is Markov if and only if

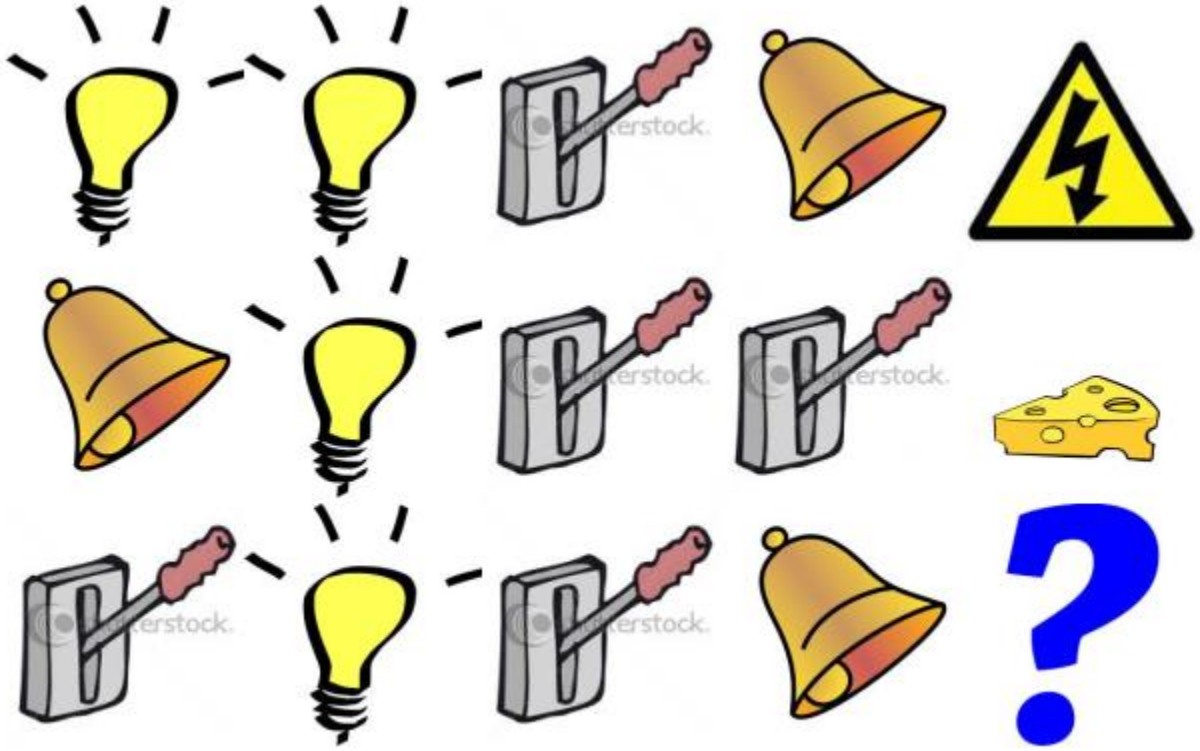
$$\mathbf{P}[S_{t+1} \mid S_t] = \mathbf{P}[S_{t+1} \mid S_1, \ldots, S_t]$$

- "The future is independent of the past given the present"

$$H_{1:t} \rightarrow S_t \rightarrow H_{t+1:inf}$$

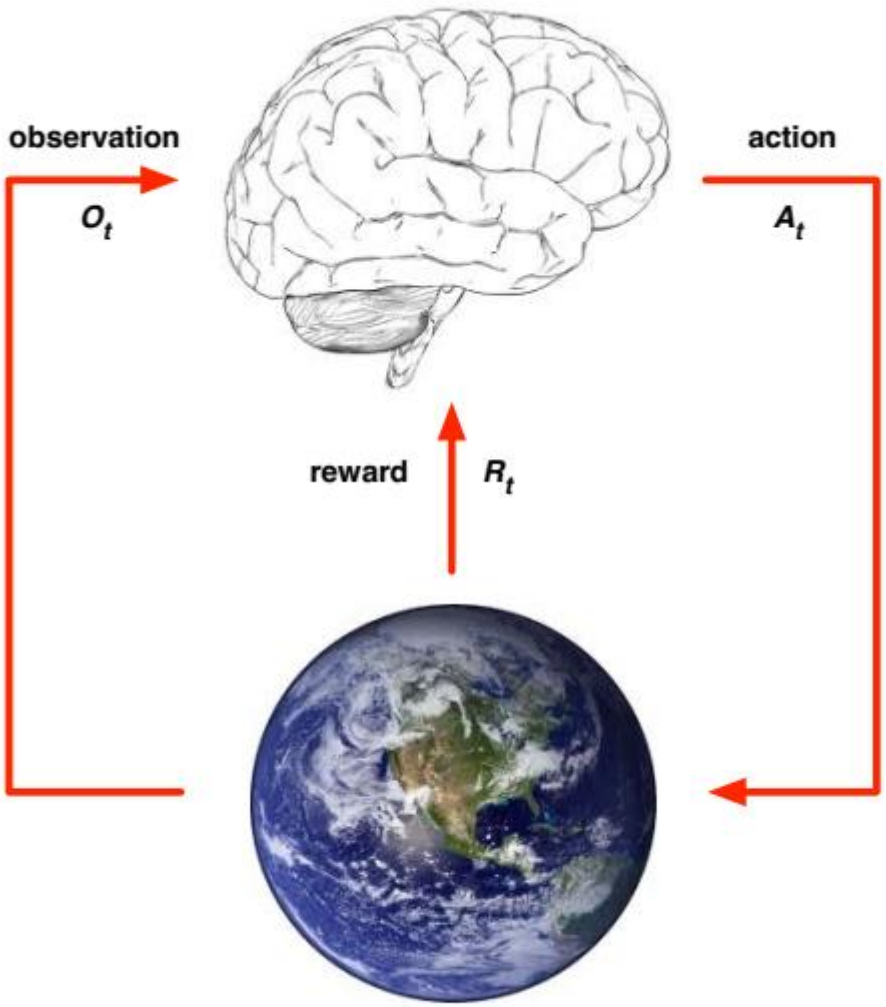
- Once the state is known, the history may be thrown away
- i.e. The state is a sufficient statistic of the future
- The environment state S_t^e is Markov
- The history H_t is Markov

Rat Example



- What if agent state = last 3 items in sequence?
- What if agent state = counts for lights, bells ?
- What if agent state = complete sequence?

Fully Observable Environments



- Full observability: agent directly observes the environment state

$$O_t = S_t^a = S_t^e$$

- Agent state = environment state = information state
- Formally, this is a Markov decision process (MDP)

Partially Observable Environments

- Partial observability: agent indirectly observes environment:
  - A robot with camera vision isn't told its absolute location
  - A trading agent only observes current prices
  - A poker playing agent only observes public cards
- Now agent state != environment state
- Formally this is a partially observable Markov decision process (POMDP)
- Agent must construct its own state representation S_t^a:
  - Complete history: $S_t^a = H_t$
  - Beliefs of environment state: $S_t^a = (P[S_t^e = s_1], \ldots, P[S_t^e = s_n])$
  - Recurrent neural network: $S_t^a = \sigma(S_{t-1}^a W_s + O_t W_o)$

Major Components of an RL Agent

- An RL agent may include one or more of these components:
  - Policy: agent's behavior function
  - Value function: how good is each state and/or action
  - Model: agent's representation of the environment

Policy

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

Value Function

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

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

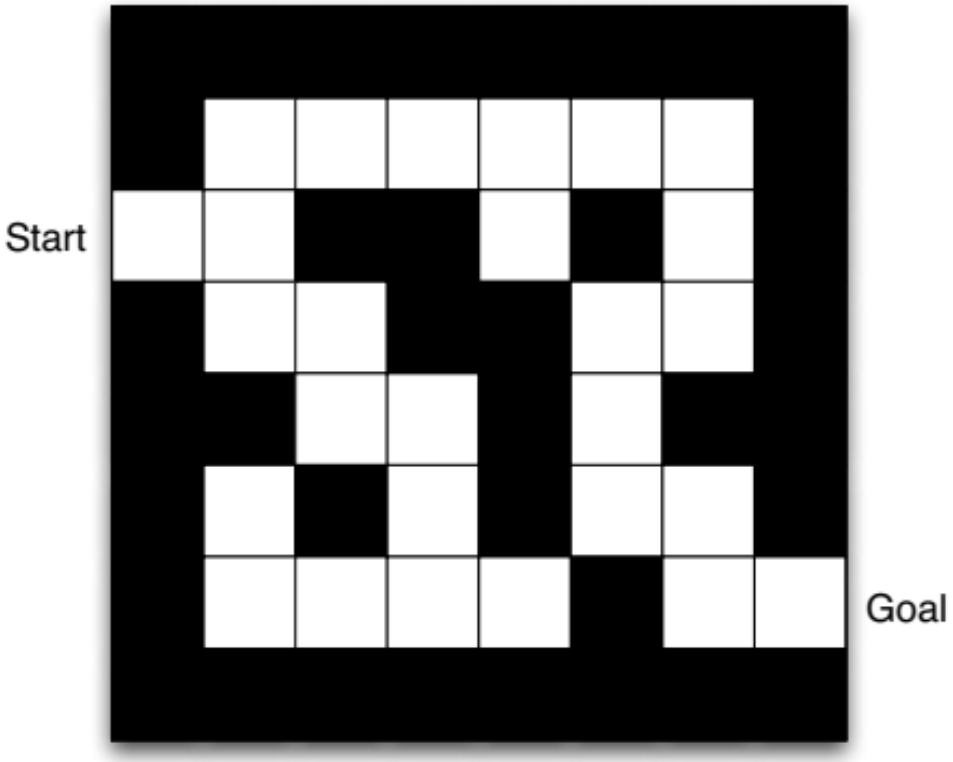
Model

- A model predicts what the environment will do next

- ***P*** predicts the next state
- ***R*** predicts the next (immediate) reward, e.g.

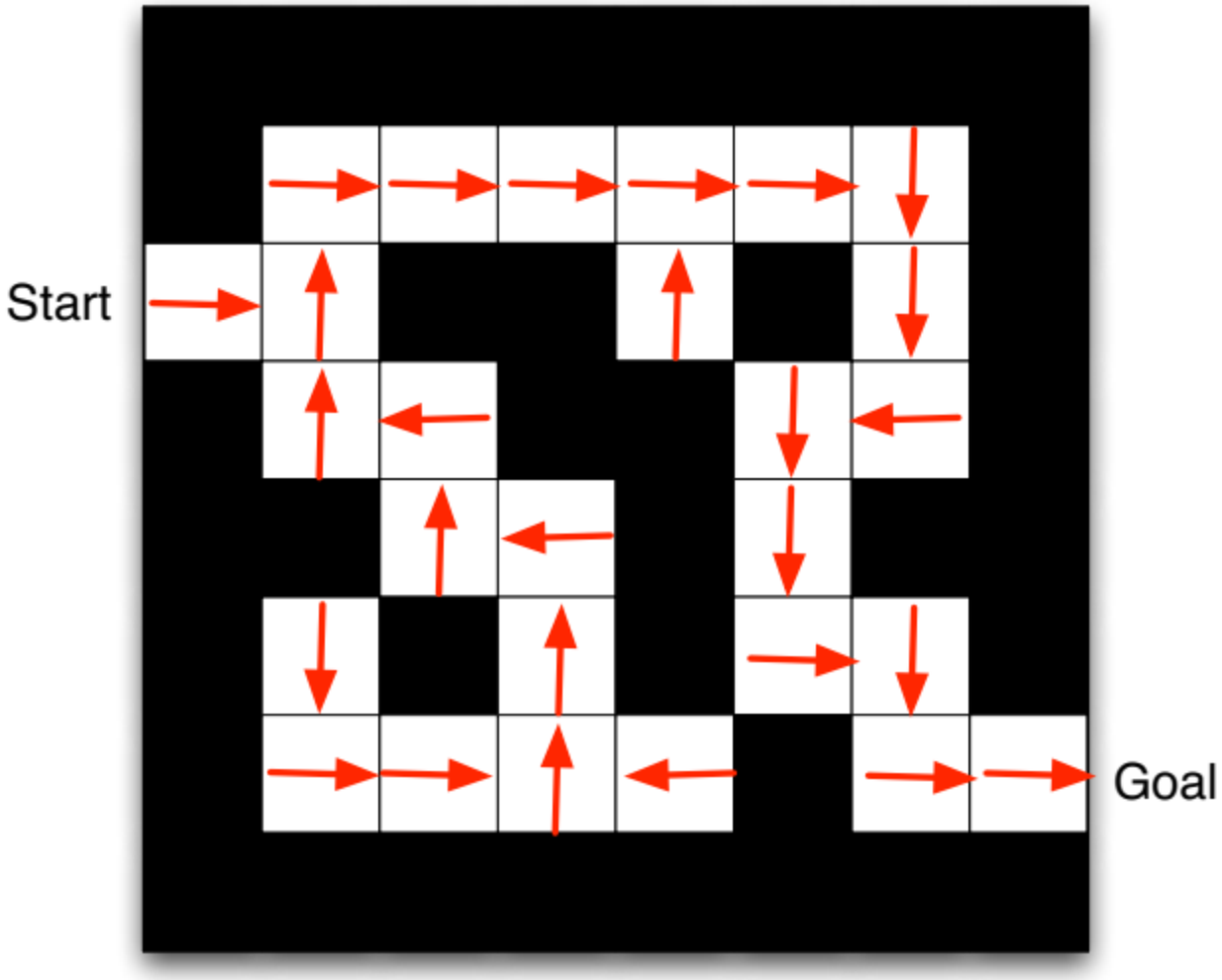
$$\boldsymbol{P}_{ss'}^{a} = \mathrm{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$$
$$\boldsymbol{R}_{s}^{a} = \mathrm{E}[R_{t+1} \mid S_t = s, A_t = a]$$

Maze Example

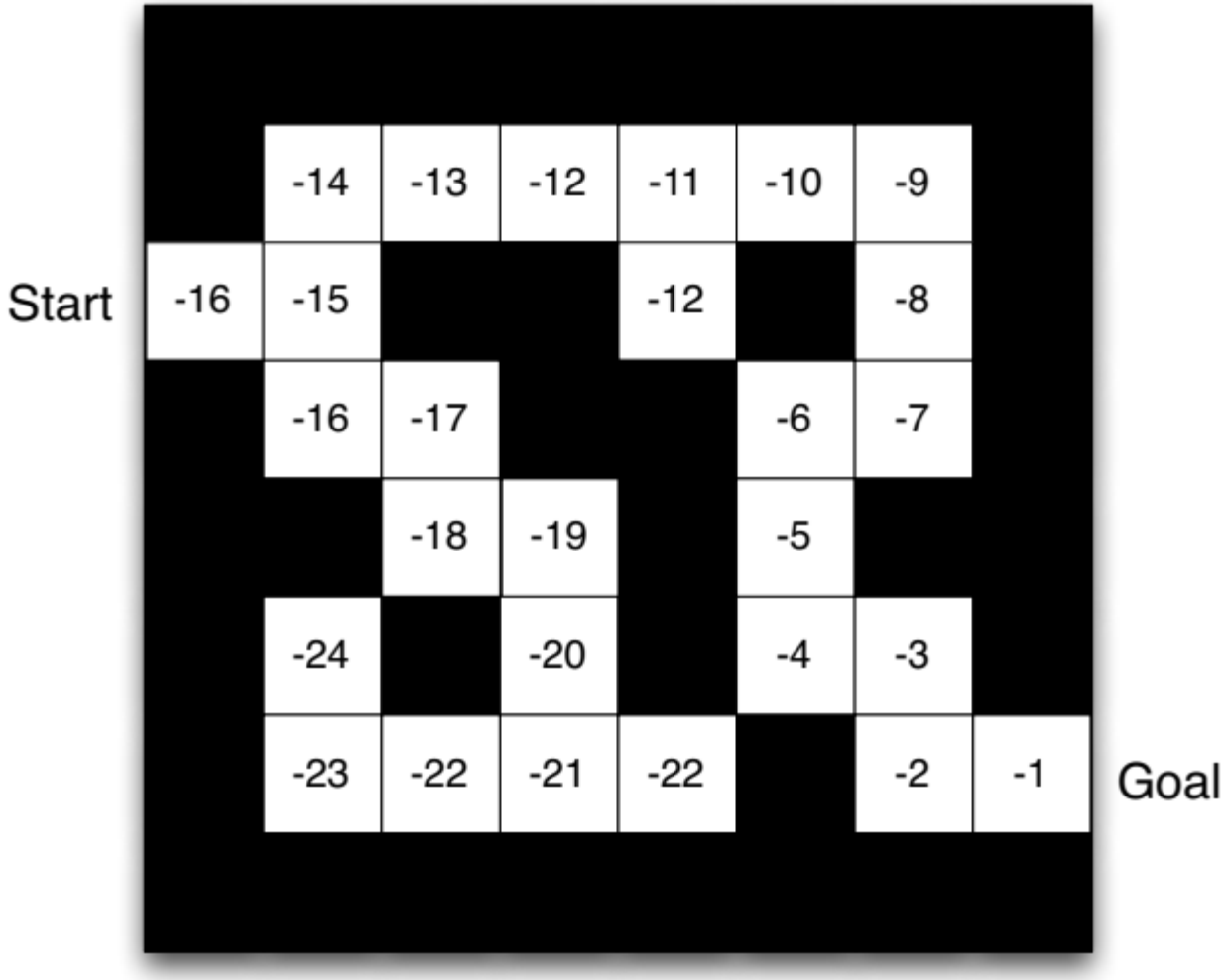


- Rewards: -1 per time-step
- Actions: N, E, S, W
- States: Agent's location

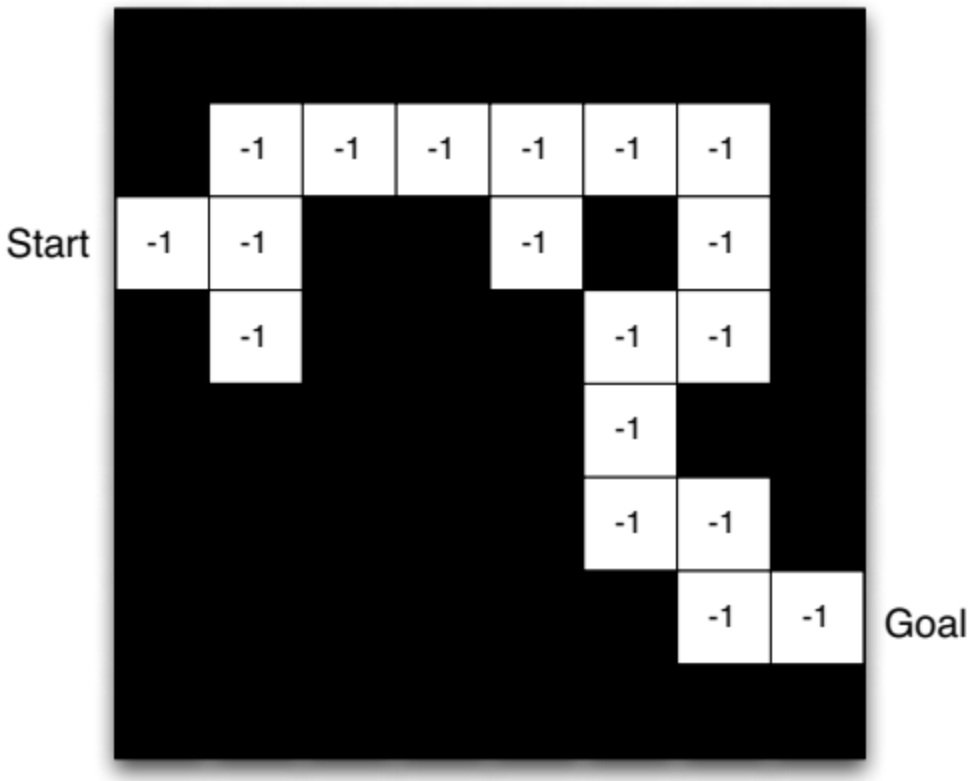
Maze Example: Policy



Maze Example: Value Function



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 $\boldsymbol{P}_{ss'}^{a}$
- Numbers represent immediate reward $\boldsymbol{R}_{s}^{a}$ from each state s (same for all a)

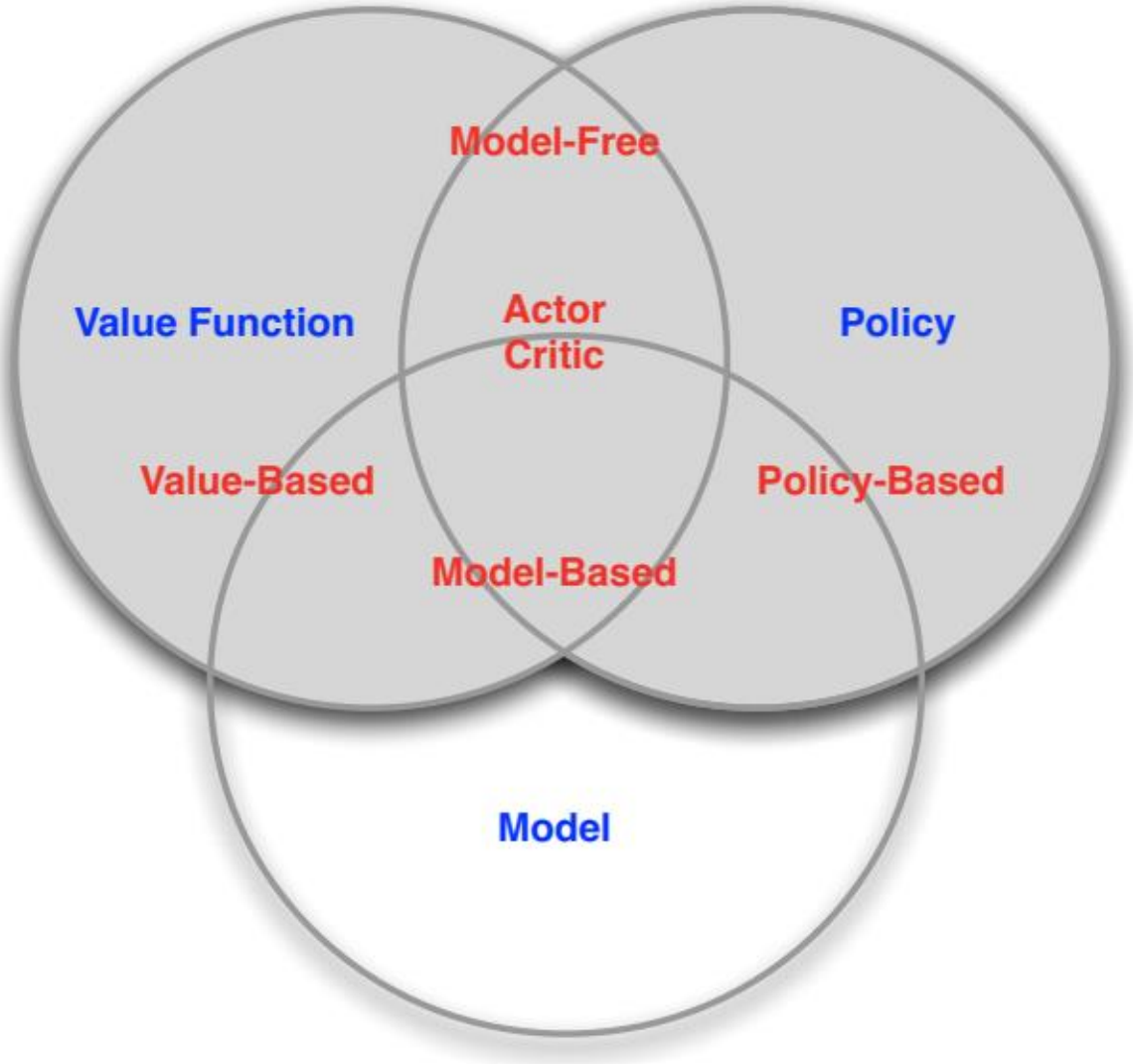
Classification RL Agents

- Value Based:
  - No Policy (Implicit)
  - Value Function
- Policy Based:
  - Policy
  - No Value Function
- Actor Critic
  - Policy
  - Value Function

Classification RL Agents

- Model Free:
  - Policy and/or Value Function
  - No Model
- Model Based:
  - Policy and/or Value Function
  - Model

RL Agents Taxonomy

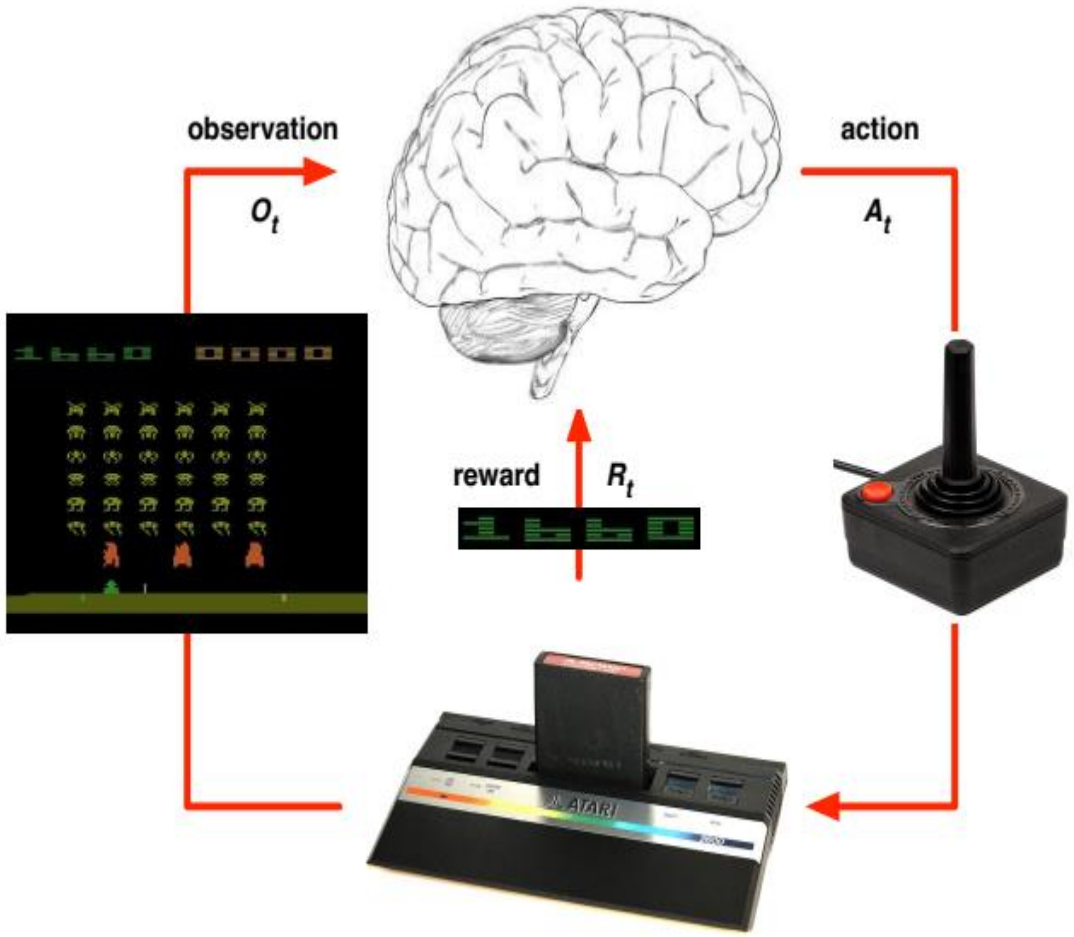


Learning & Planning

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

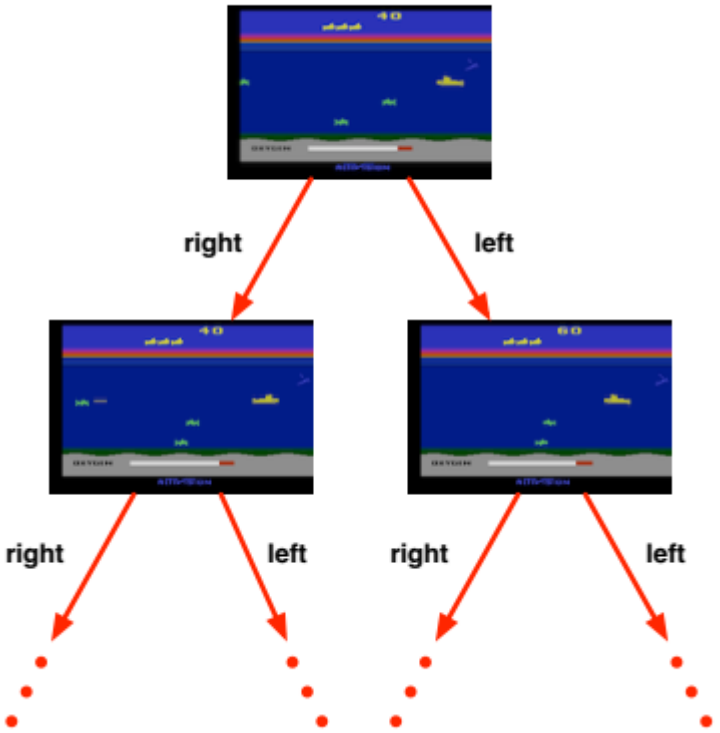
Atari Example: Learning



- Rules of the game are unknown
- Learn directly from interactive game-play
- Pick actions on joystick, see pixels and scores

Atari Example: Planning

- Rules of the game are known
- Can query emulator
  - perfect model inside agent's brain
- If I take action a from state s:
  - what would the next state be?
  - what would the score be?
- Plan ahead to find optimal policy
  - e.g. tree search

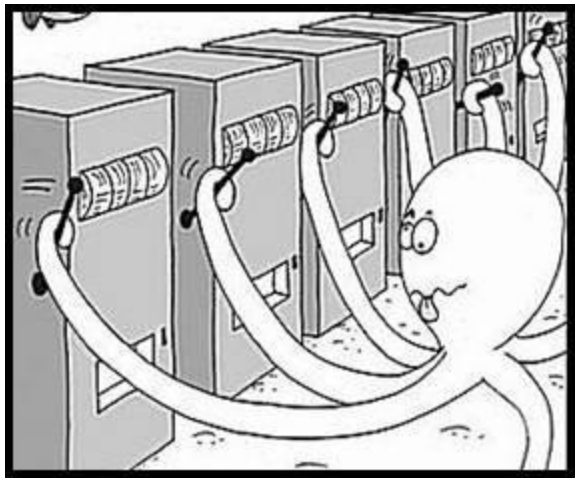


Exploration & Exploitation

- Reinforcement learning is like *trial-and-error* learning
- The agent should discover a good policy
- From its experiences of the environment
- Without losing too much reward along the way

Exploration & Exploitation

- *Exploration* finds more information about the environment
- *Exploitation* exploits known information to maximize reward
- It is usually important to explore as well as exploit



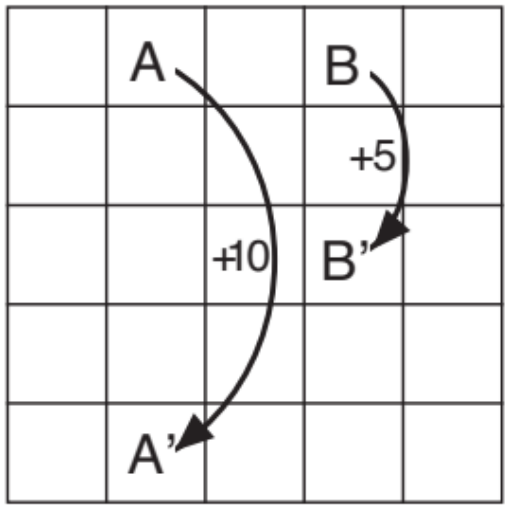
Examples:

- Bar Selection
  - Exploitation: Go to your favorite bar
  - Exploration: Try a new bar
- Online Banner Advertisements
  - Exploitation: Show the most successful advert
  - Exploration: Show a different advert
- Oil Drilling
  - Exploitation: Drill at the best known location
  - Exploration: Drill at a new location
- Game Playing
  - Exploitation: Play the move you believe is best
  - Exploration: Play an experimental move

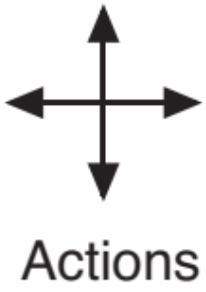
Prediction & Control

- Prediction: evaluate the future
  - Given a policy
- Control: optimize the future
  - Find the best policy

Grid World Example: Prediction



(a)

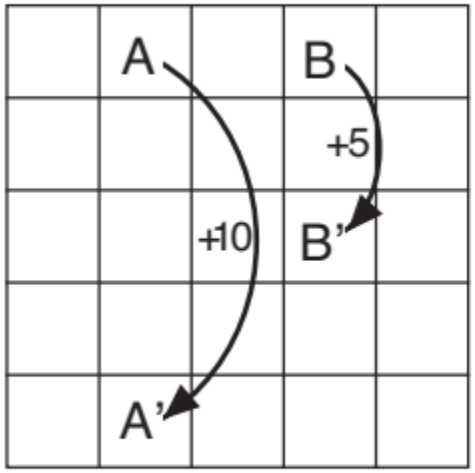


3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

(b)

- What is the value function for random uniform policy?

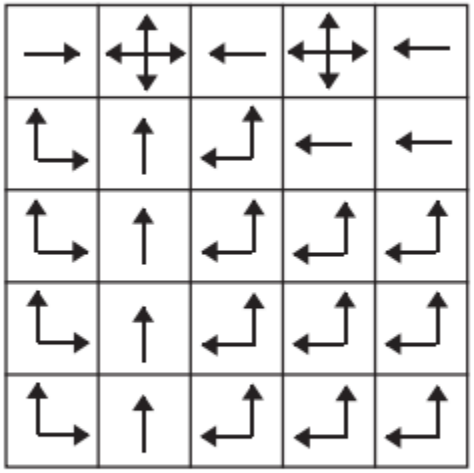
Grid World Example: Control



a) gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

b) v_*



c) π_*

- What is the optimal value function for all possible policies?
- What is the optimal policy?

Resume

Questions?