Reinforcement Learning

Jay Urbain, PhD

Credits:

Reinforcement Learning: An Introduction (2nd Edition),
Richard S. Sutton and Andrew G. Barto.
David Silver's Reinforcement Learning Course
https://github.com/dennybritz/reinforcement-learning

How to learn a new skill?”

- One of the most fundamental question for scientists has been – “How to learn a new skill?”
- If we can understand this, we can enable human species to do things we might not have thought before.
- Alternately, we can train machines to do more “human” tasks and create true artificial intelligence.

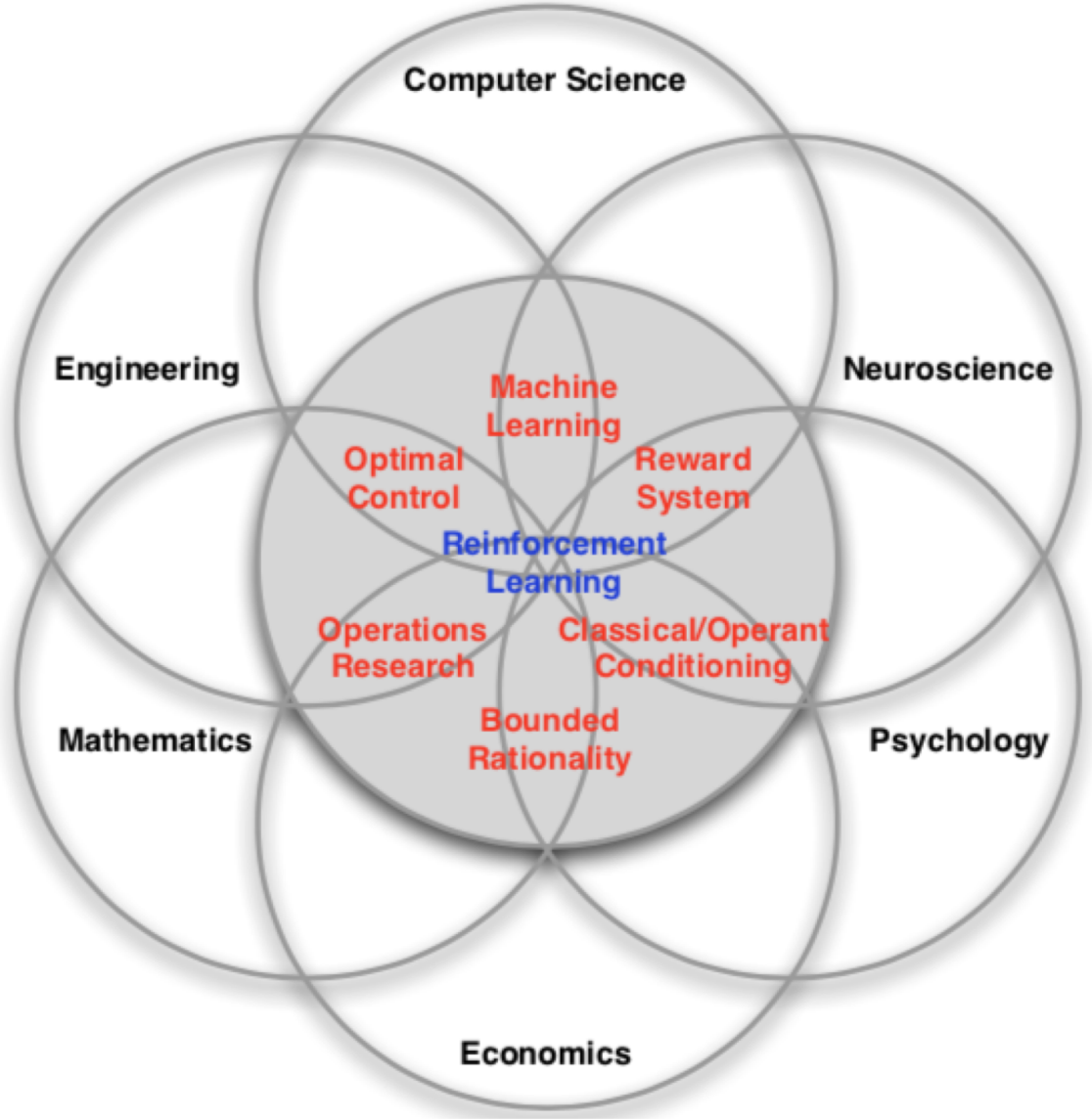
How to learn a new skill?”

- We don’t have a complete answer to the above question yet, there are a few things which are clear.
- Irrespective of the skill, we first learn by *interacting with the environment*.
- Whether we are learning to drive a car or an infant learning to walk, the learning is based on the interaction with the environment.
- Learning from interaction is the foundational underlying concept for all theories of learning and intelligence.

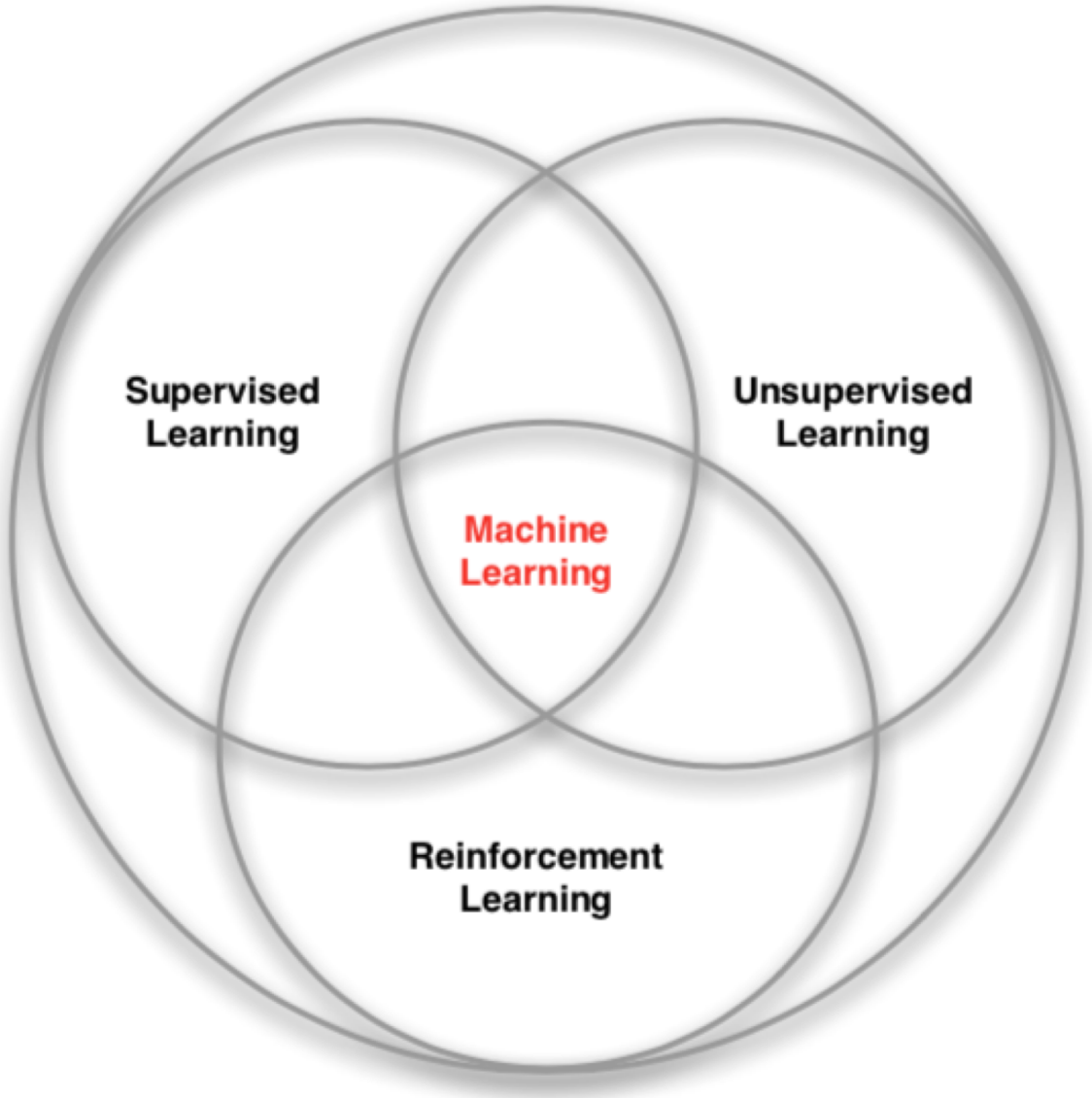
Reinforcement Learning

– Reinforcement Learning – a goal-oriented learning based on interaction with environment. Reinforcement Learning is said to be the hope of true artificial intelligence.

Many Faces of Reinforcement Learning



Branches of Machine Learning



Characteristics of Reinforcement Learning

What makes reinforcement learning different from other machine learning paradigms?

- There is no supervisor, only a reward signal
- Feedback is delayed, not instantaneous
- Time really matters (sequential, non i.i.d data)
- Agent's actions affect the subsequent data it receives

Examples of Reinforcement Learning

- Fly stunt manoeuvres in a helicopter
- Defeat the world champion at Backgammon
- Manage an investment portfolio
- Control a power station
- Make a humanoid robot walk
- Play many different Atari games better than humans
- Beat the world's best in Go

RL examples: Playing Atari with Deep Reinforcement Learning

YouTube:

https://www.youtube.com/watch?v=MKtNv1UOaZA

Reference:

https://deepmind.com/research/publications/playing-atari-deep-reinforcement-learning/

RL examples: Stanford Autonomous Helicopter

Stanford:
http://heli.stanford.edu/

RL examples: DeepMind game of Go deep

YouTube:
https://www.youtube.com/watch?v=g-dKXOlsf98

Reference:
https://deepmind.com/research/alphago/

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:*

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

– *Do you agree with this statement?*

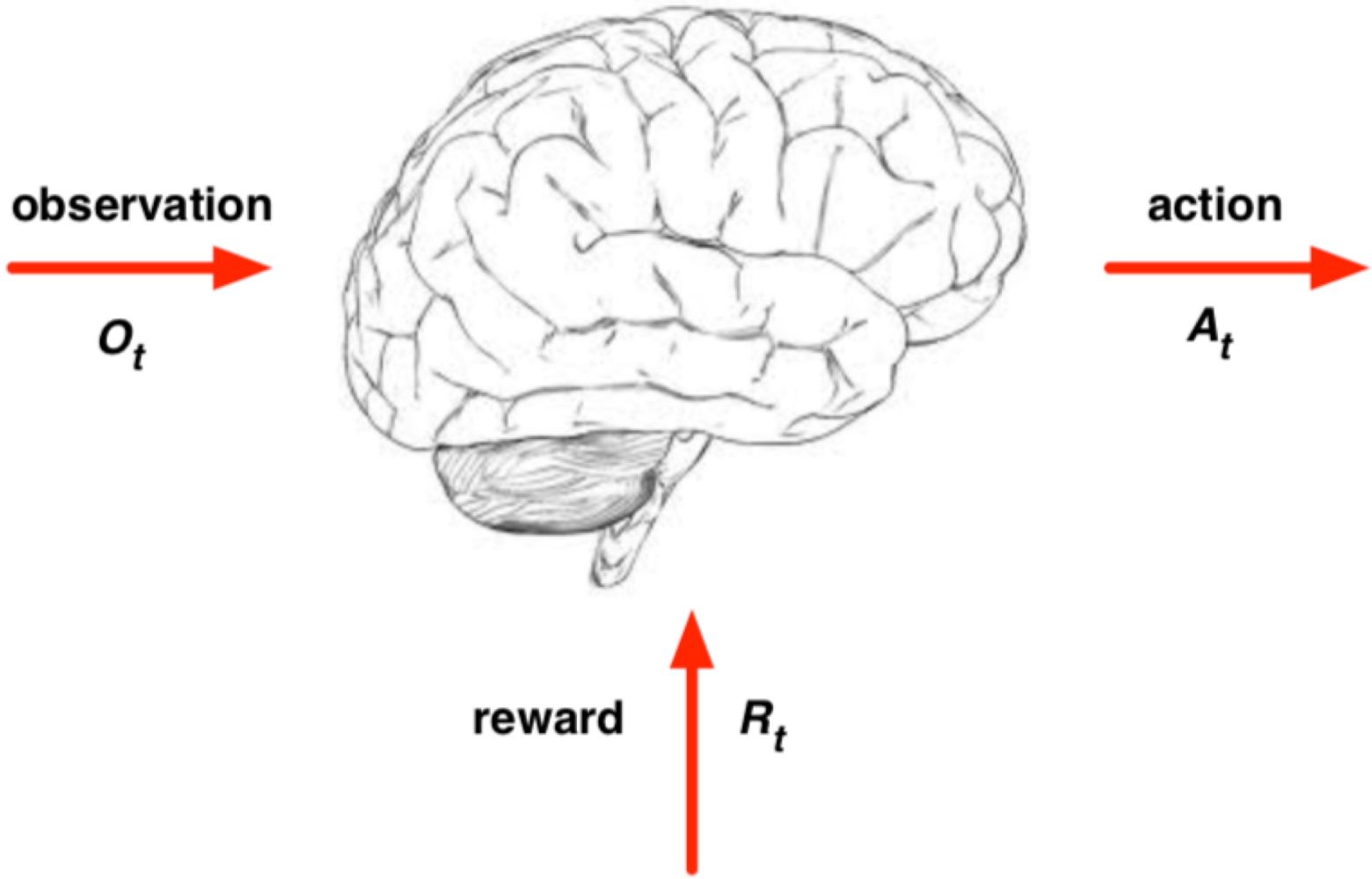
Examples of rewards

- **Fly stunt maneuvers in a helicopter**
  - +ve reward for following desired trajectory
  - −ve reward for crashing
- **Defeat the world champion at Backgammon**
  - +/−ve reward for winning/losing a game
- **Manage an investment portfolio**
  - +ve reward for each $ in bank
- **Control a power station**
  - +ve reward for producing power
  - −ve reward for exceeding safety thresholds
- **Make a humanoid robot walk**
  - +ve reward for forward motion
  - −ve reward for falling over
- **Play many different Atari games better than humans**
  - +/−ve reward for increasing/decreasing score

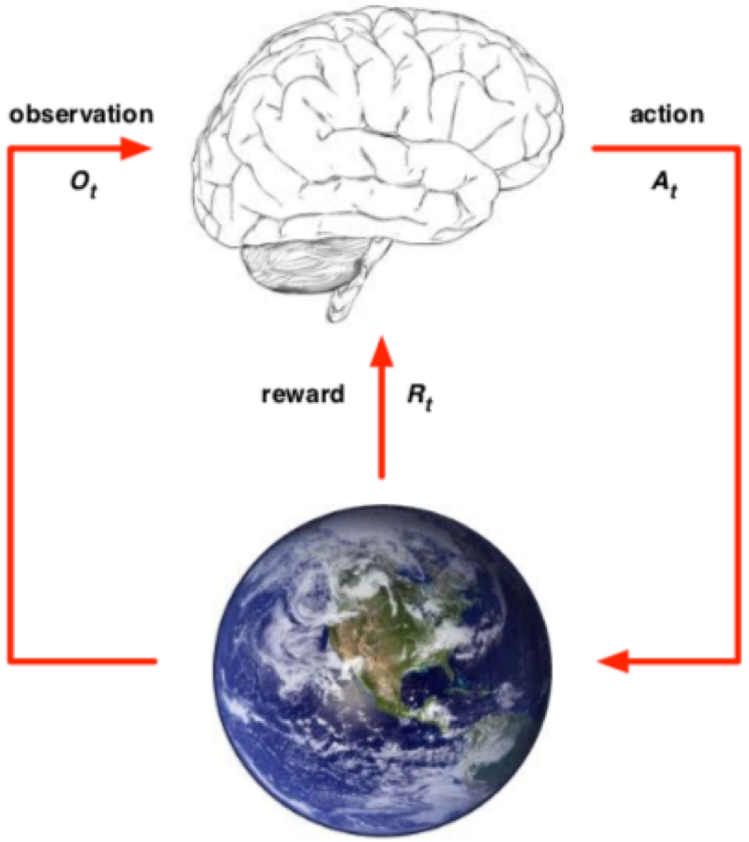
Sequential Decision Making

- 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
- Examples:
  - 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 and Environment



Agent and 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

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

- 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:

$$S_t = f(H_t)$$

History and State

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

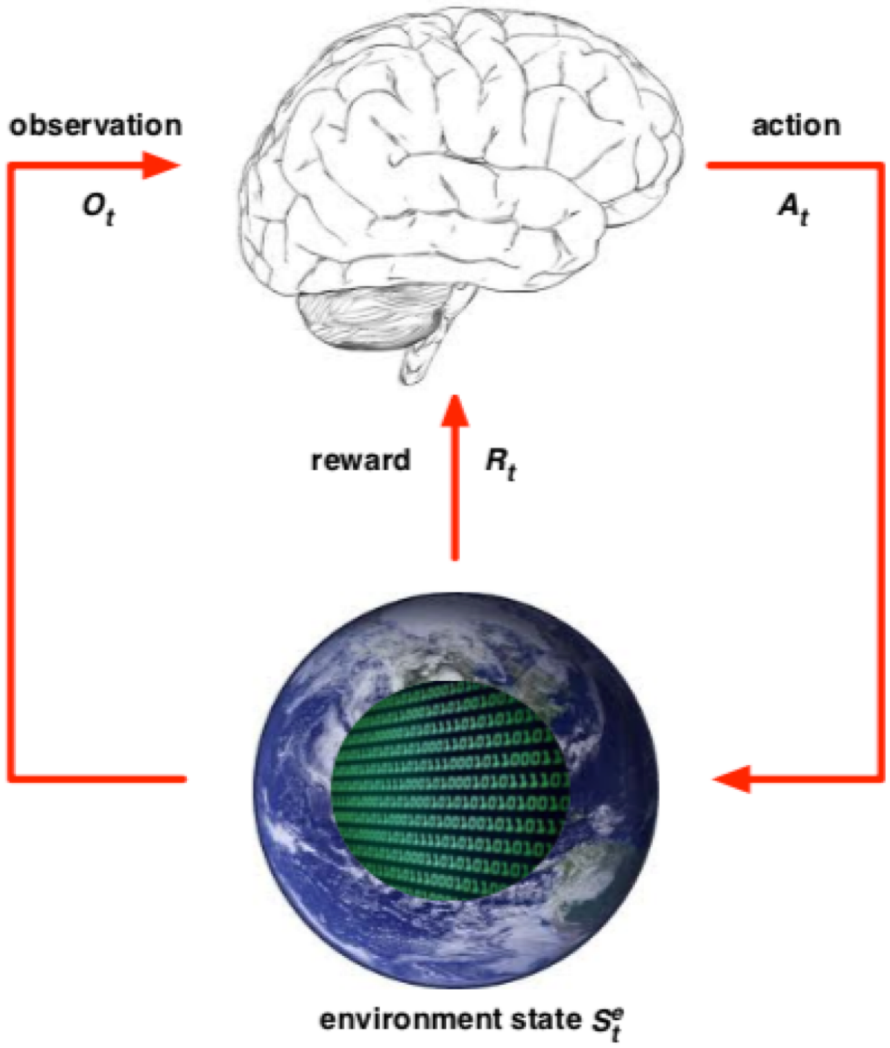
$$H_t = O_1, R_1, A_1, ..., A_{t-1}, O_t, R_t$$

- 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

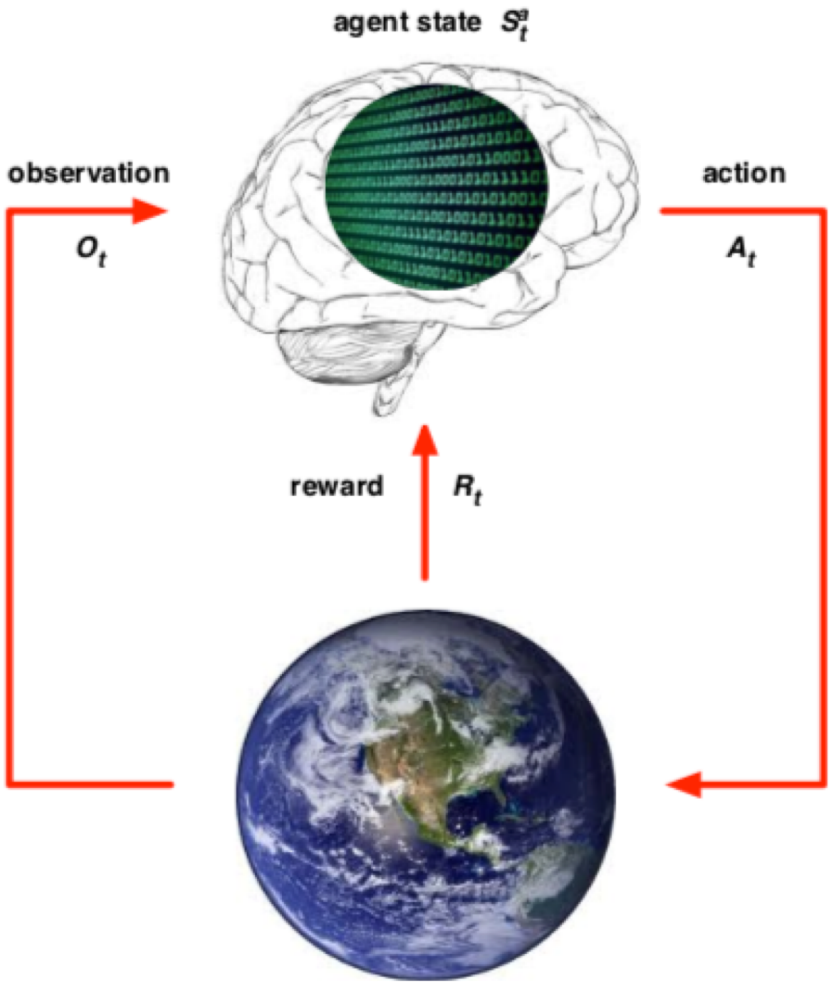
- The environment state S^t_e is the environment's private representation
- i.e. whatever data the environment uses to pick the next observation/reward
- The environment state is not usually visible to the agent
- Even if S^t_e is visible, it may contain irrelevant information



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:

$$S_t^a = f(H_t)$$



Information State

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

A state S_t is Markov if and only if:

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

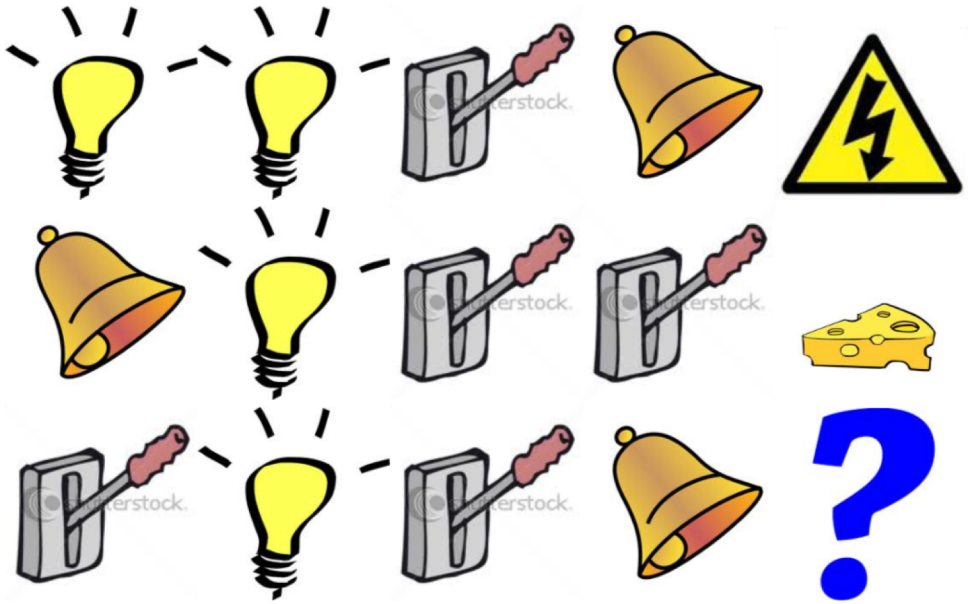
– "The future is independent of the past given the present"

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

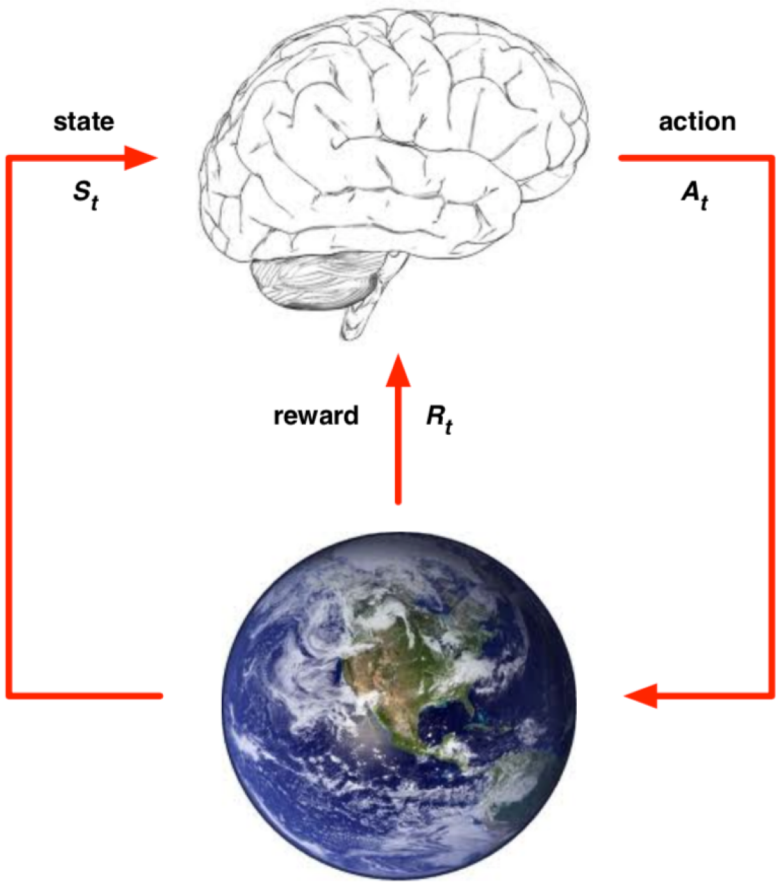
– Once the state is known, the history may be thrown away i.e. The state is a sufficient statistic of the future
– The environment state S^t_e is Markov
– The history H_t is Markov

Rat Example

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



Fully Observable Environments



Full observability: agent directly observes environment state

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

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

Fully 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 $\neq$ environment state
- Formally this is a partially observable Markov decision process (POMDP)
- Agent must construct its own state representation S_t^a, e.g.
  - Complete history: $S_t^a = H_t$
  - Beliefs of environment state: $S_t^a = (\mathbb{P}[S_t^e = s^1], ..., \mathbb{P}[S_t^e = s^n])$
  - 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 behaviour
- It is a map from state to action, e.g.
- Deterministic policy: $a = \pi(s)$
- Stochastic policy: $\pi(a|s) = \mathbb{P}[A_t = a|S_t = s]$

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.

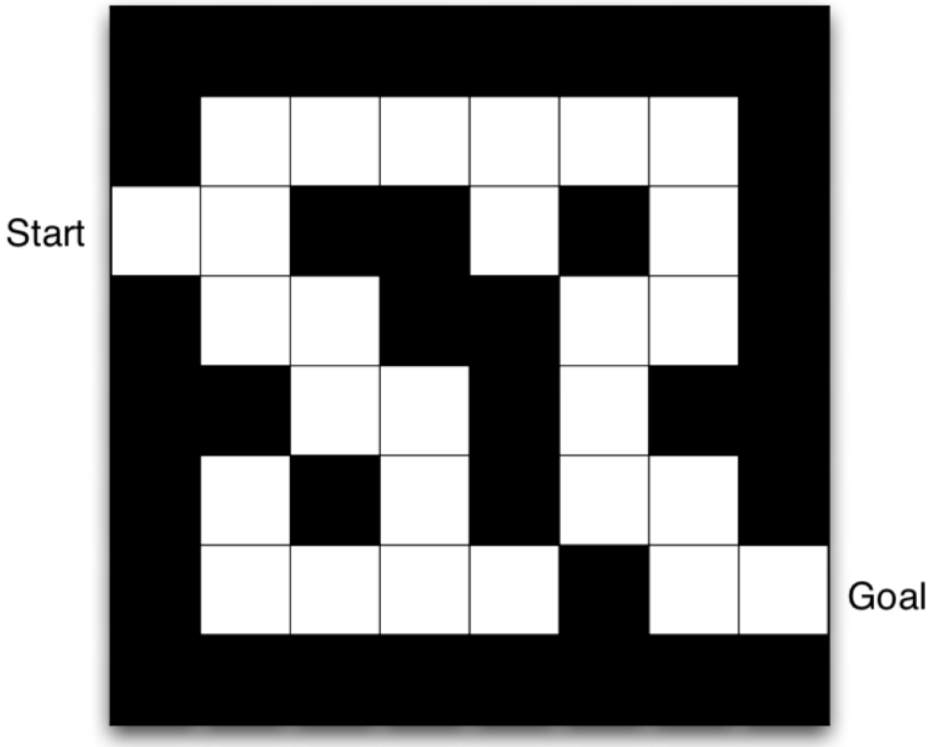
$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s \right]$$

Model

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

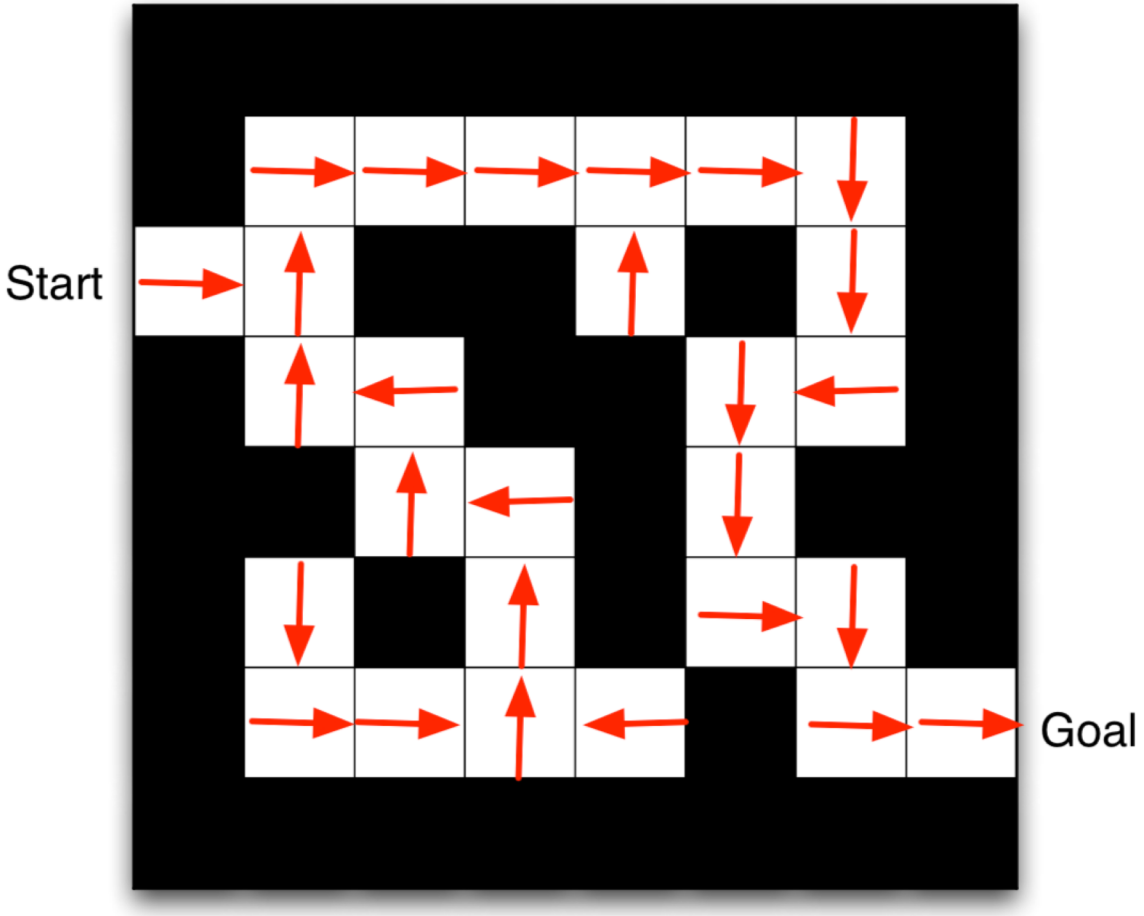
$$\mathcal{P}^a_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$$
$$\mathcal{R}^a_s = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$$

Maze Example



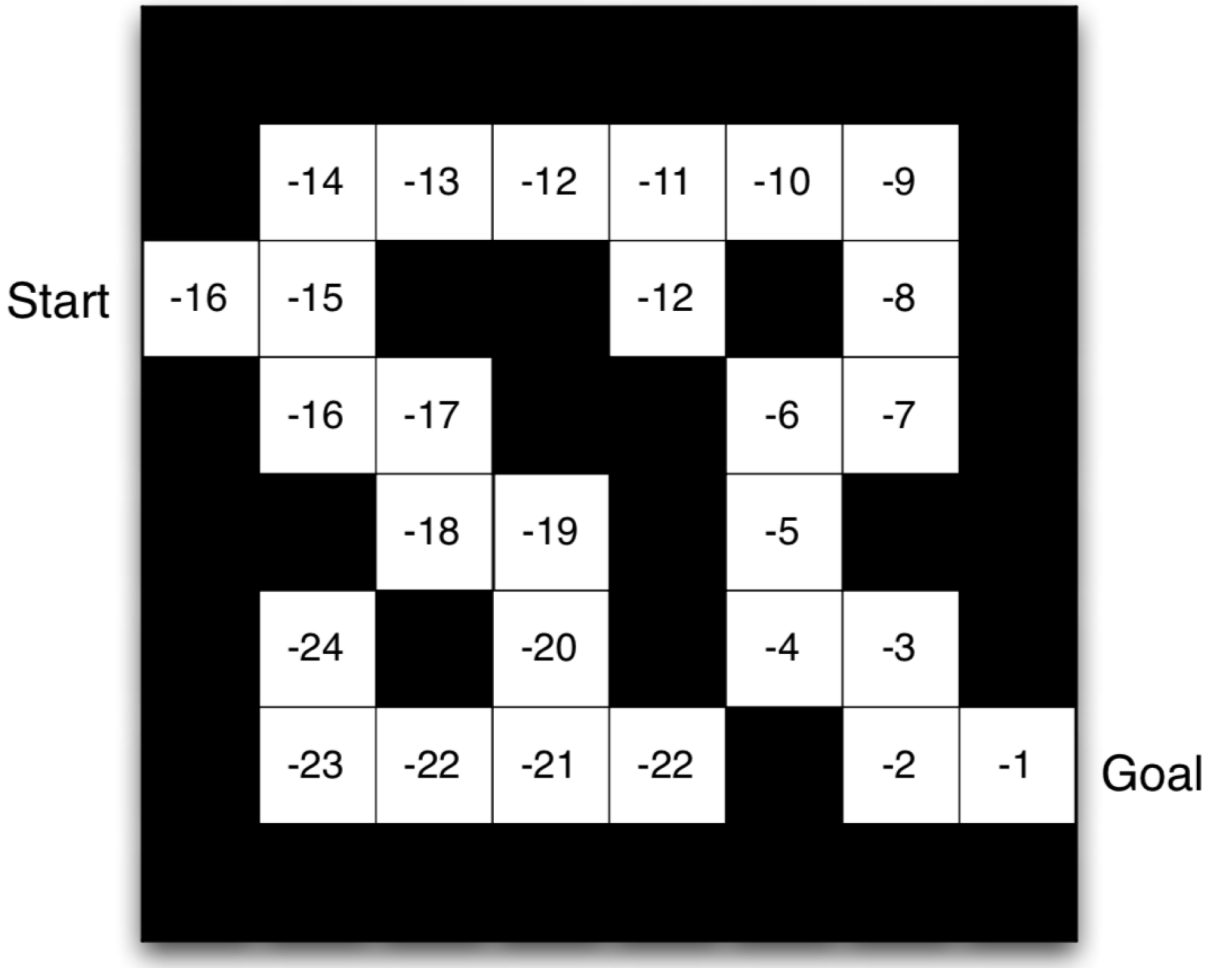
- Rewards: -1 per time-step
- Actions: N, E, S, W
- States: Agent's location

Maze Example: Policy



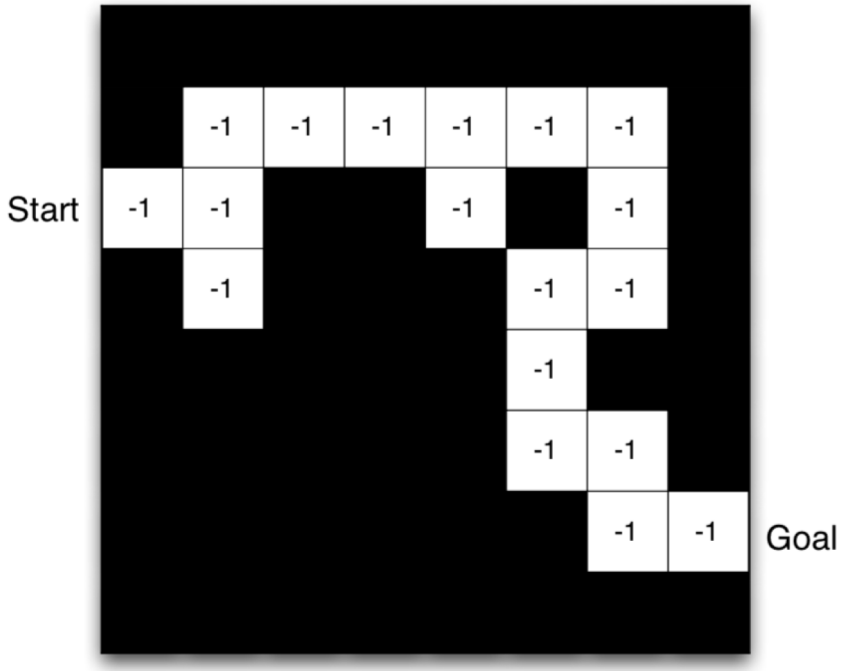
- Arrows represent policy $\pi(s)$ for each state s

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)