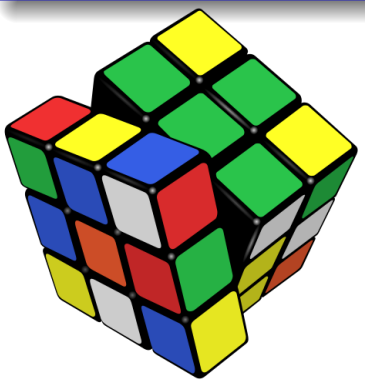
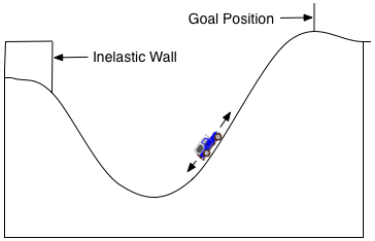
Reinforcement Learning

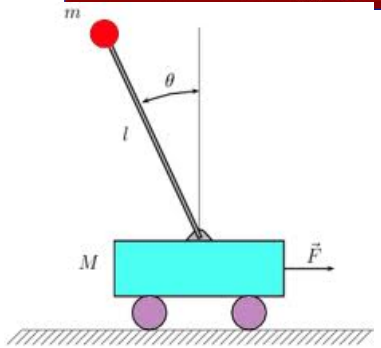
Introduction



Marcello Restelli

February, 2024





Outline

1 Course Information

2 What is Reinforcement Learning?

3 Sequential Decision Problem Examples

4 Modeling the Problem

5 Algorithmic Solutions in RL

Who am I?

- Associate Professor at PoliMi
- Courses
  - Machine Learning
  - Information Retrieval and Data Mining
  - Robotics
- Research interests
  - Reinforcement Learning
  - Multi-agent Learning
  - Online Learning
- Industrial collaborations
  - E-commerce
  - Finance
  - Automotive
  - Industry 4.0
- Co-founded ML cube in 2020

Course Goals

- Learn to correctly **model** sequential decision problems
- Learn **techniques** and fundamental ideas of RL
- Learn to **apply** RL to practical problems
- Learn **limitations** of RL techniques
- Provide the basic background to do **research** in this field
- My expectations
  - ask questions
  - interact
  - get involved

Schedule

Introduction to RL and MDPs	06/02/2024 (09:30 - 12:30) – online
Solving MDPs	07/02/2024 (09:30 - 12:30) – online
RL in finite MDPs	08/02/2024 (09:30 - 12:30) – online
RL in continuous MDPs	12/02/2024 (09:30 - 12:30) – Room E. Gatti
Model-based RL	13/02/2024 (09:30 - 12:30) – Room E. Gatti
Policy Gradient	14/02/2024 (09:30 - 12:30) – Room E. Gatti
Deep RL	15/02/2024 (09:30 - 12:30) – Room E. Gatti
Practical Session	16/02/2024 (09:00 - 13:00) – Room E. Gatti

Teaching Material

- Slides
- Lecture recordings
- All materials will be made available on `https://webeep.polimi.it/course/view.php?id=10728`

Textbooks

- Sutton and Barto, “Reinforcement Learning: an Introduction”, MIT Press, 2018. `http://incompleteideas.net/book/the-book-2nd.html`
- Bertsekas and Tsitsiklis, “Neuro–Dynamic Programming”, Athena Scientific, 1996.
- Szepesvari, “Algorithms for Reinforcement Learning”, Morgan and Claypool, 2010.
- Agarwal, Jiang, Kakade, and Sun, “Reinforcement Learning: Theory and Algorithms”, 2021. `https://rltheorybook.github.io/`.
- Bertsekas, “Dynamic Programming and Optimal Control, Vol.II, 4th Edition: Approximate Dynamic Programming”. Athena Scientific, 2012.

Assessment

- PhD students
  - Reproducibility Challenge
  - `https://paperswithcode.com/rc2022`
- MSc students
  - Oral exam

But Who's Counting?

But Who's Counting?

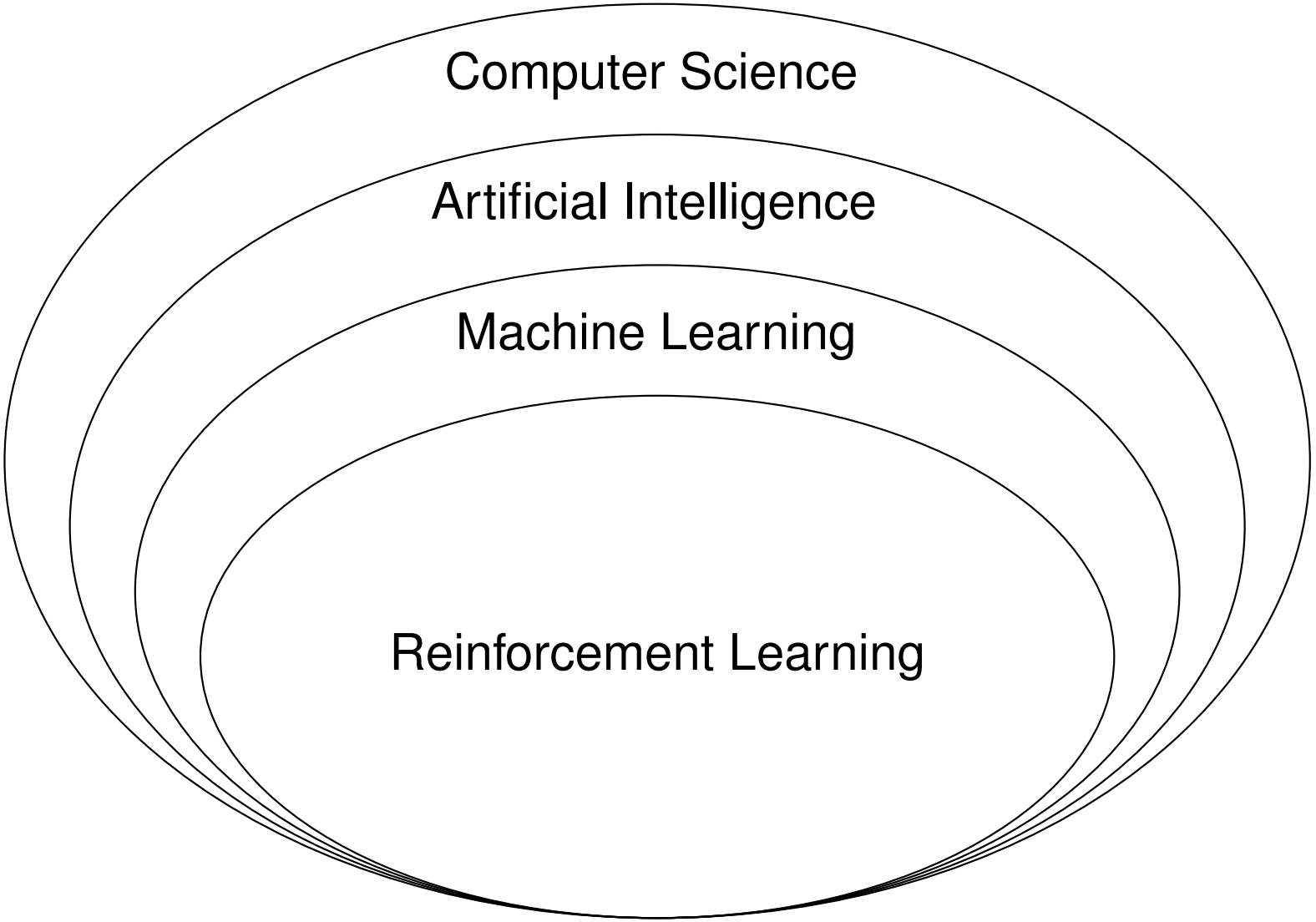
But Who's Counting?

- First game
  - Best possible value: 75421
  - Value following the optimal policy: 75142
- Second game
  - Best possible value: 76530
  - Value following the optimal policy: 75630

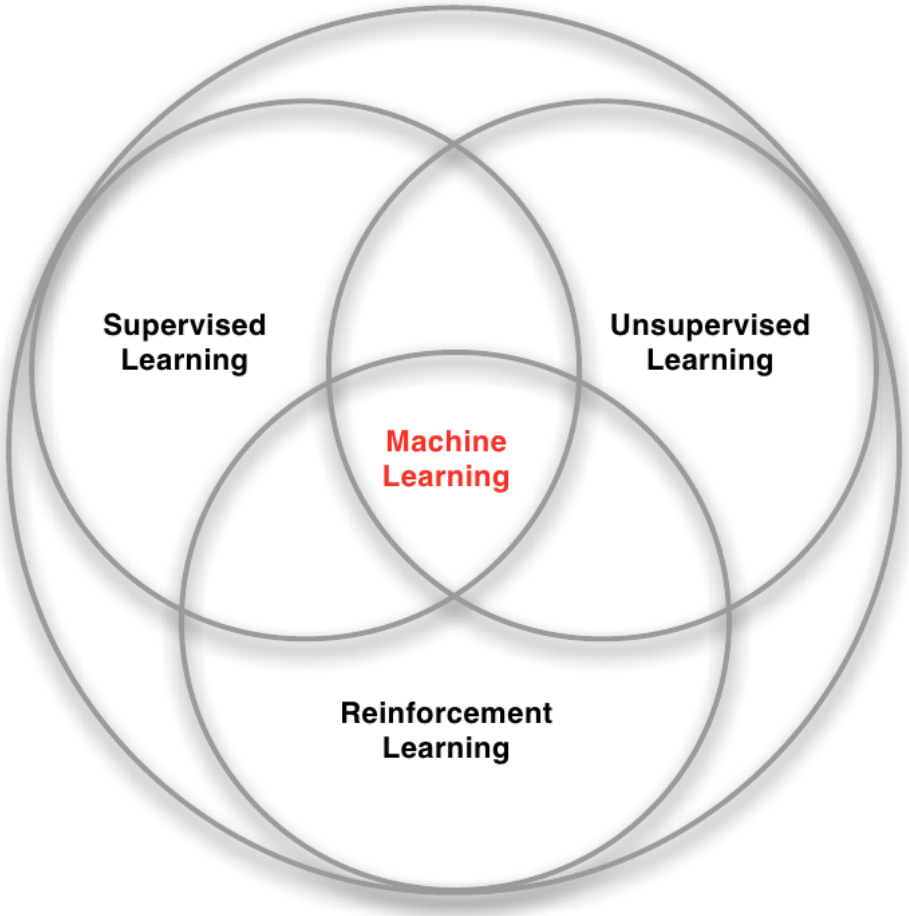
Let's play Blackjack!

- Goal:
  - Making a higher point than the dealer without going over 21
- Rules:
  - the cards are drawn: 2 for the player and 1 for the dealer
  - starts the player, she can choose between **hit** (a new card is drawn to the player) or **stand**
  - if the player goes over 21 (**busts**) or she chooses **stand**, begins the dealer turn
  - the dealer draws cards until the value of her hand is lower than **17**
- Result:
  - if the player busted, the player **loses**, otherwise
  - if the dealer busted, the player **wins**, otherwise
  - if the player hand is higher than the dealer one, the player **wins**, otherwise
  - if the player hand is equal to the dealer one, the player **make a push**, otherwise
  - the player **loses**

Where RL comes from?

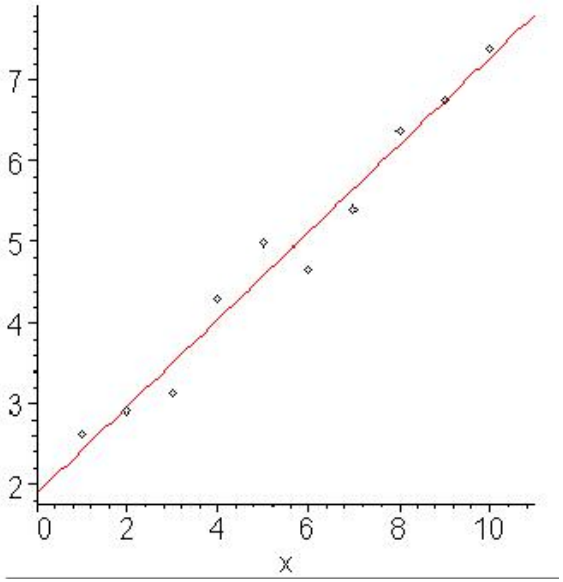


Machine Learning



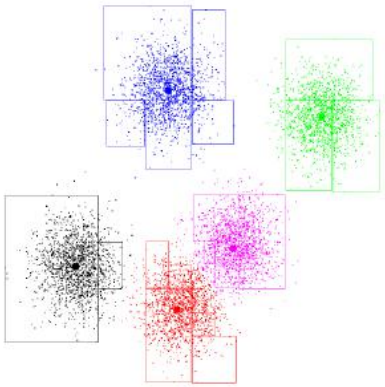
Machine Learning Models

- Supervised Learning
  - Learn the model
- Unsupervised Learning
  - Learn the representation
- Reinforcement Learning
  - Learn to control



Machine Learning Models

- Supervised Learning
  - Learn the model
- Unsupervised Learning
  - Learn the representation
- Reinforcement Learning
  - Learn to control



Machine Learning Models

- Supervised Learning
  - Learn the model
- Unsupervised Learning
  - Learn the representation
- Reinforcement Learning
  - Learn to control

Where RL comes from?

Marcello Restelli

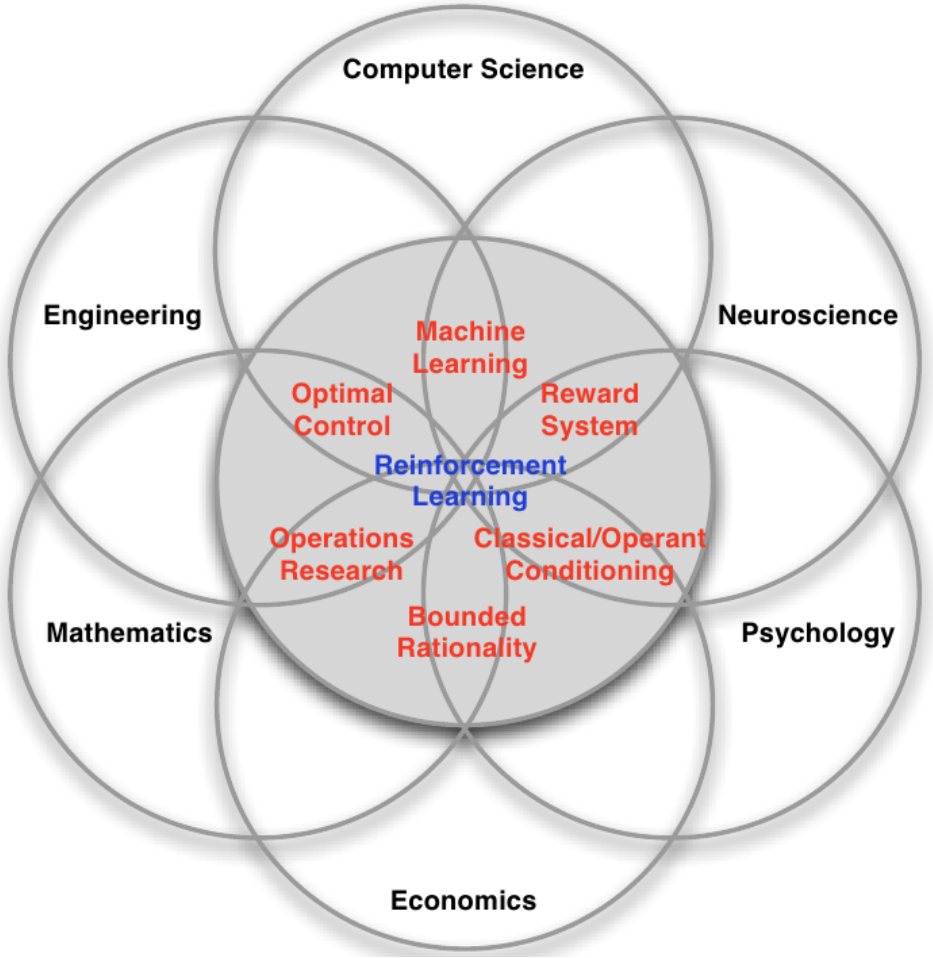
Course Information

What is Reinforcement Learning?

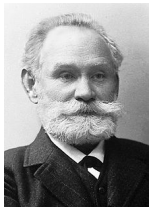
Sequential Decision Problem Examples

Modeling the Problem

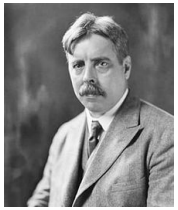
Algorithmic Solutions in RL



History of RL



Pavlov (1903)
Classical conditioning



Thorndike (1905)
Law of effect

Minsky (1961)
Credit–assignment problem

Psychology, Trial and error

Watkins (1989)
Q–learning

Littman (1994)
minimax–*Q*

Samuel (1956)
Checkers

Reinforcement Learning

Sutton & Barto (1984)
Temporal Difference

Tesauro (1992)
TD–Gammon

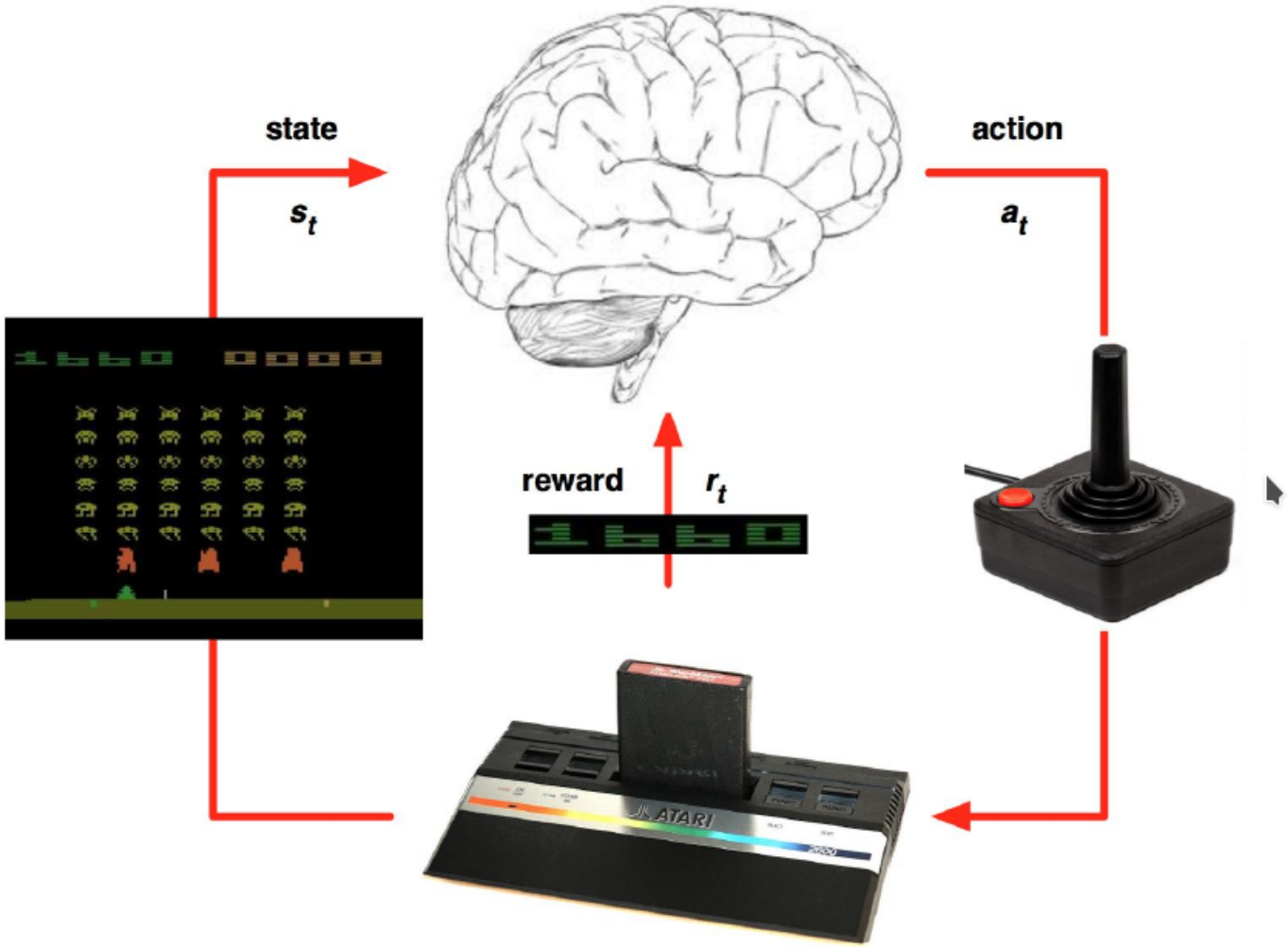
Optimal Control

Bellman (1957)
Dynamic Programming

Howard (1960)
Policy Iteration

Recent Successes

2013: Playing Atari Games



Recent Successes

2013: Playing Atari Games

Marcello Restelli

Course Information

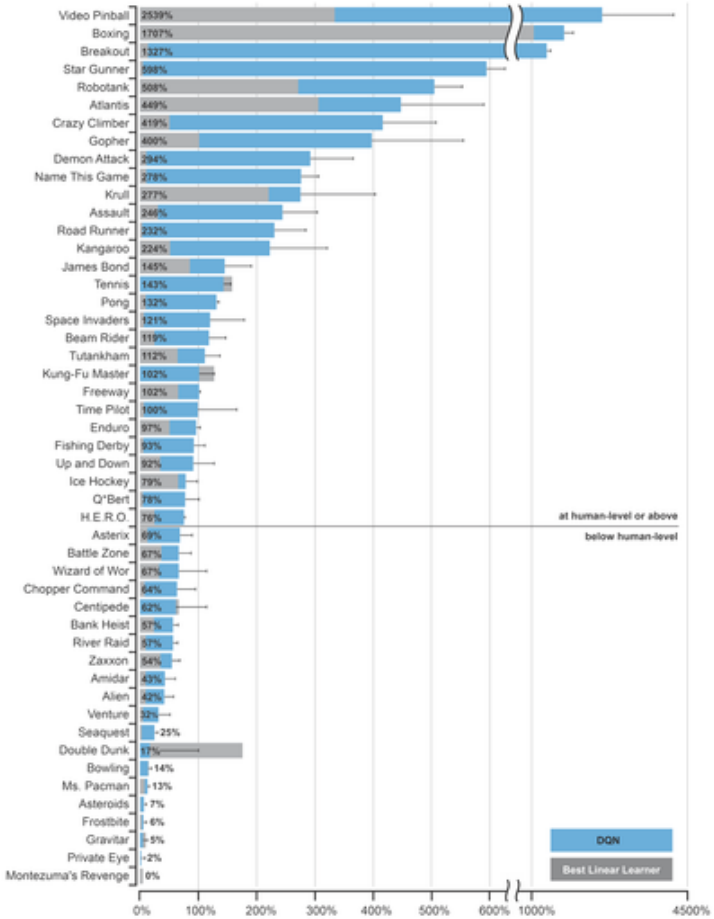
What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

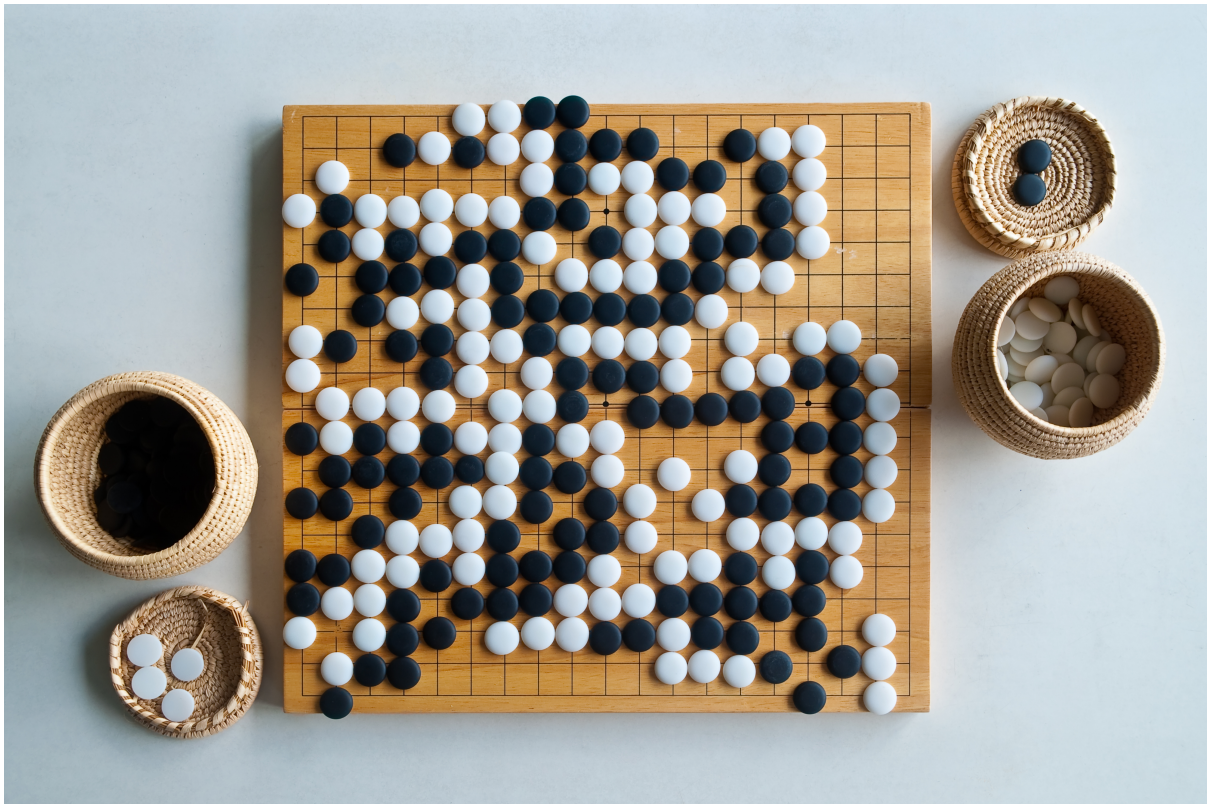
DQN



Recent Successes

2016: AlphaGo vs Lee Sedol

According to AI experts, computers would have beaten humans no sooner than 2100



10^{170} configurations

Recent Successes

2016: AlphaGo vs Lee Sedol

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

10^{170} configurations According to AI experts, computers would have beaten humans no sooner than ~~2100~~ **2016**

AlphaGo 4 - 1 Seedol

Recent Successes

2018: Learning to run

Learning to run

Recent Successes

2019: AlphaStar masters StarCraft II

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

Recent Successes

2022: RL with Human Feedback in ChatGPT

Marcello Restelli

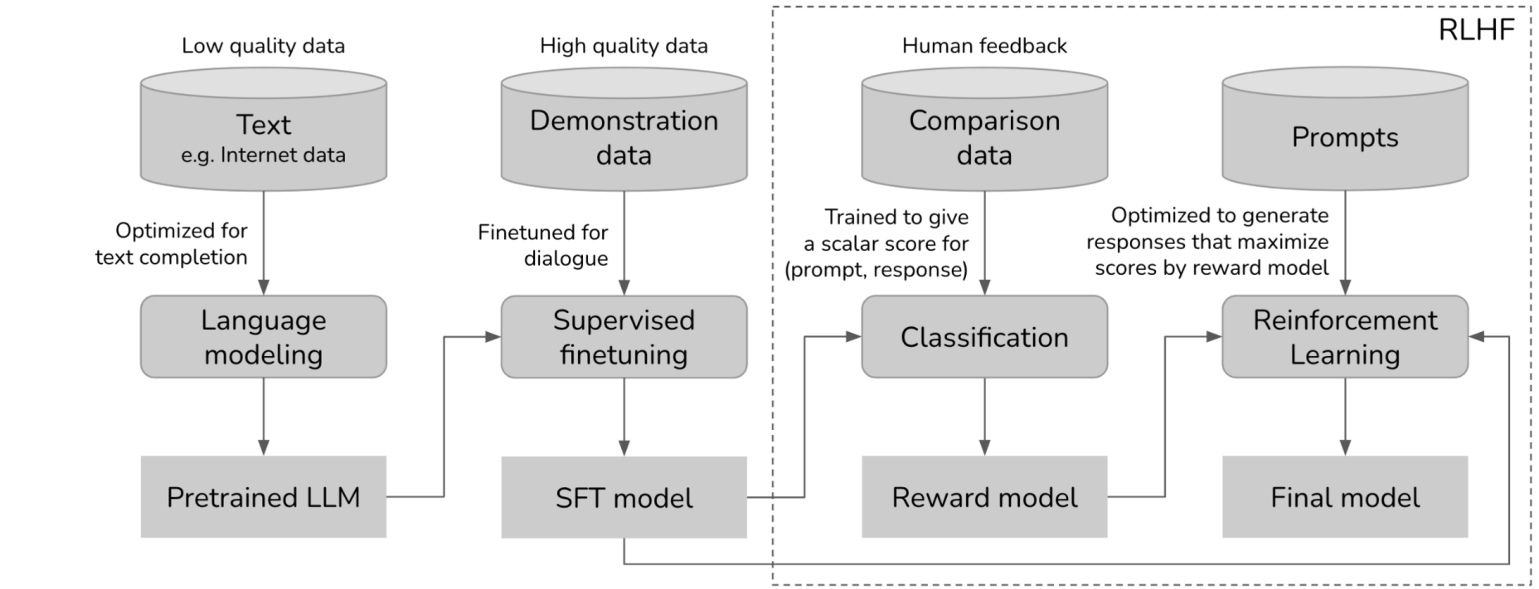
Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL



RL papers

Marcello Restelli

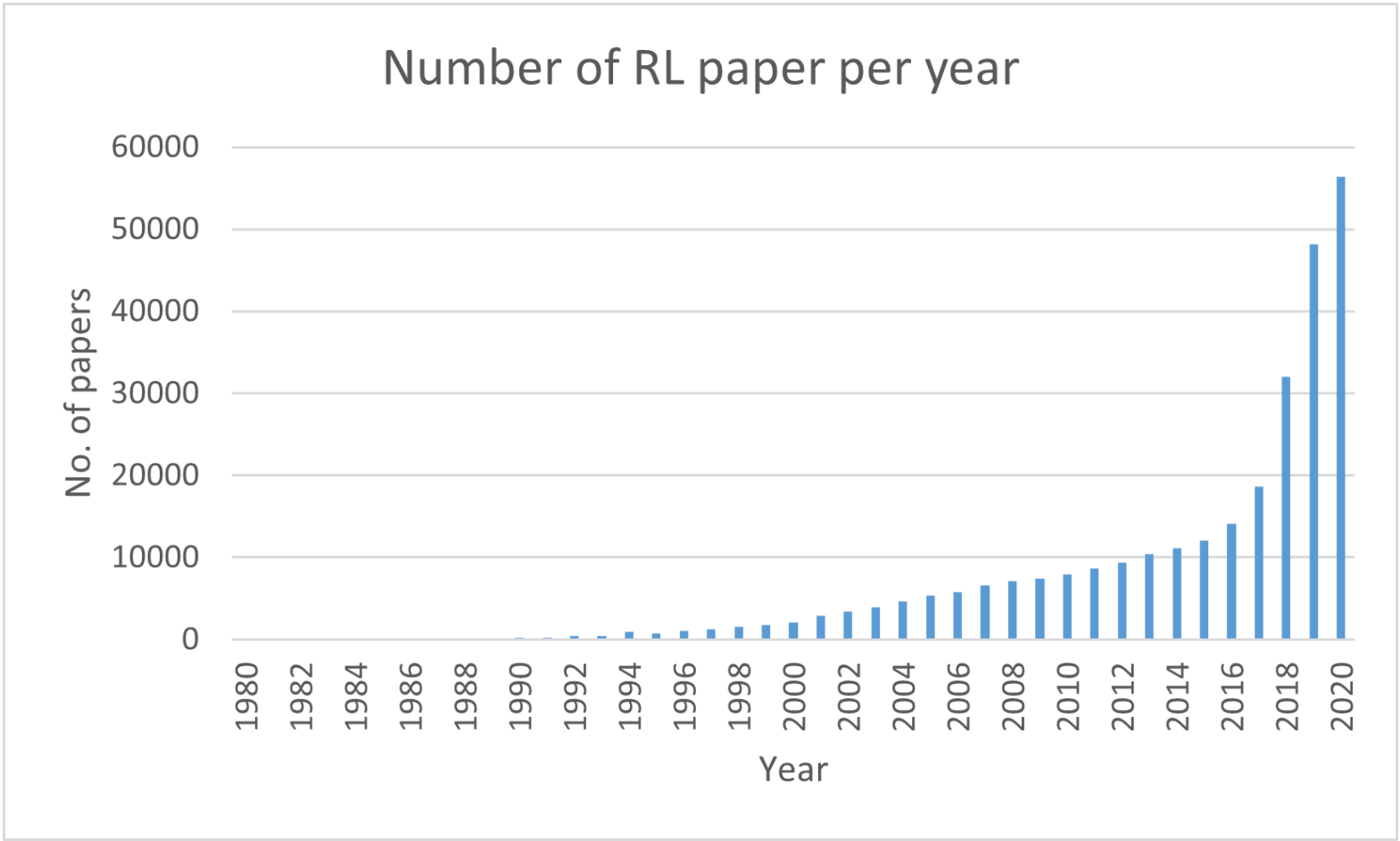
Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL



RL Top Venues

- Journals
  - Journal of Machine Learning Research (JMLR)
  - Machine Learning Journal (MLJ)
  - Journal of Artificial Intelligence Research (JAIR)
- Conferences
  - International Conference on Machine Learning (ICML)
  - Neural Information and Processing Systems (NeurIPS)
  - American Association on Artificial Intelligence (AAAI)
  - International Joint Conference on Artificial Intelligence (IJCAI)
  - Uncertainty in Artificial Intelligence (UAI)
  - Artificial Intelligence and Statistics (AI&Stats)
  - Conference on Learning Theory (CoLT)
- RL meetings
  - IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL)
  - European Workshop on Reinforcement Learning (EWRL)

Sequential Decision Making

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

- **Goal**: select actions to maximize cumulative rewards
- 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–Environment Interface

Marcello Restelli

Course Information

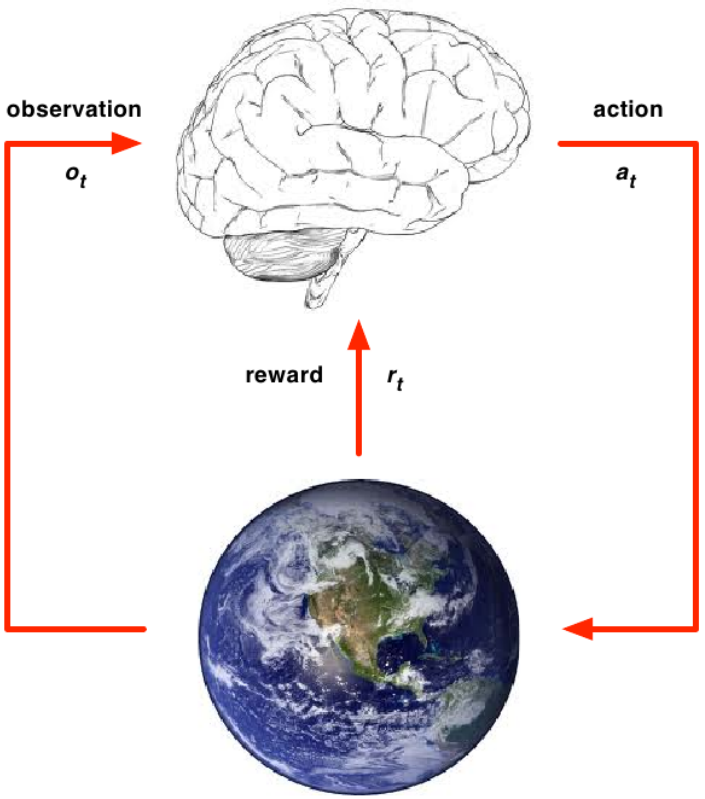
What is Reinforcement Learning?

Sequential Decision Problem

Examples

Modeling the Problem

Algorithmic Solutions in RL



- 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
  - Emits scalar reward r_t

History and State

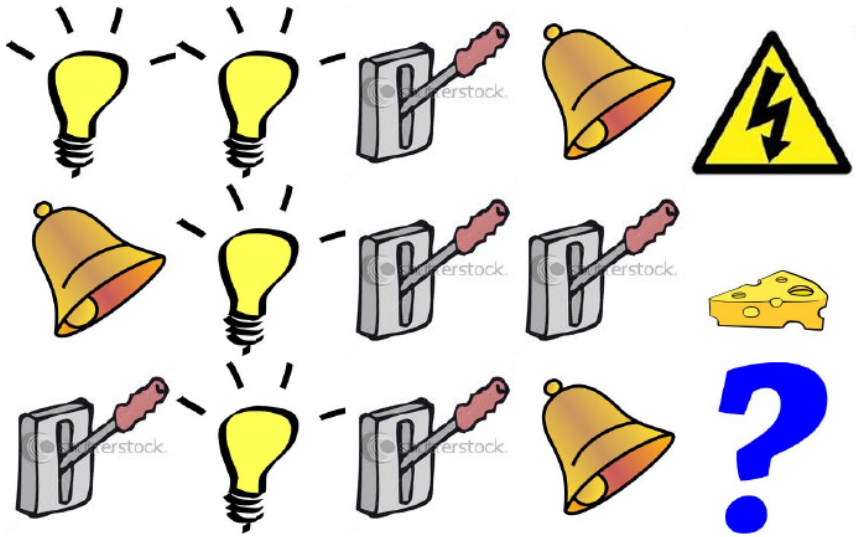
- The **history** is the sequence of observations, actions, rewards

$$h_t = a_1, o_1, r_1, \ldots, a_t, o_t, r_t$$

  - all observable variables up to time t
  - the **sensorimotor** stream of a robot or embodied agent
- What happens next depends on the history
  - agent selects actions
  - environment selects observations and rewards
- **State** is the information used to determine what happens next
- Formally, state is a function of the history:

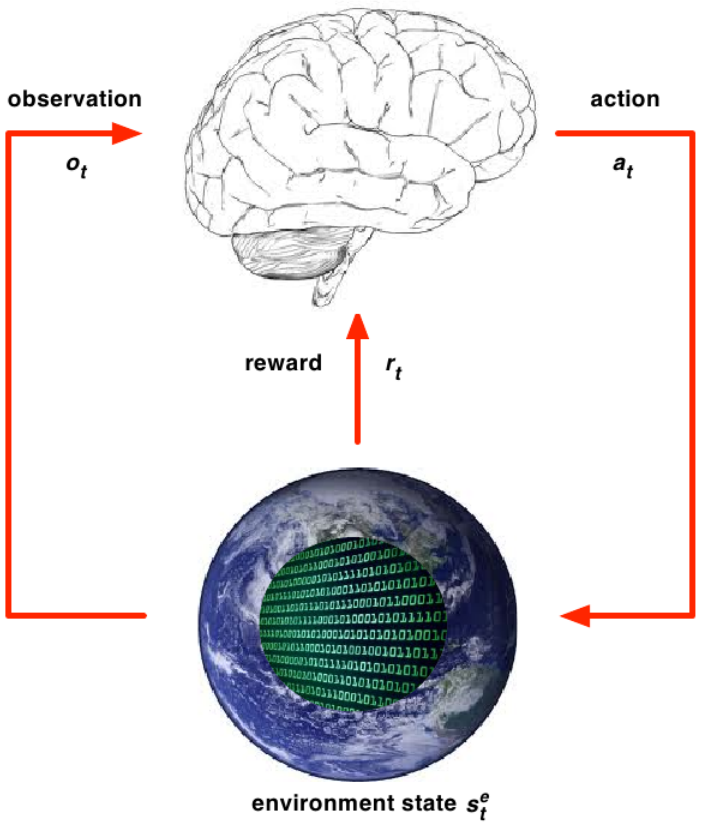
$$s_t = f(a_1, o_1, r_1, \ldots, a_t, o_t, r_t)$$

Rat Example



- What if agent state = last 3 observations ?
- What if agent state = counts of different observations?
- What if agent state = complete sequence?

Environment State



- The environment state s_t^e is the environment's private representation
  - whatever representation the environment uses to produce 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

Marcello Restelli

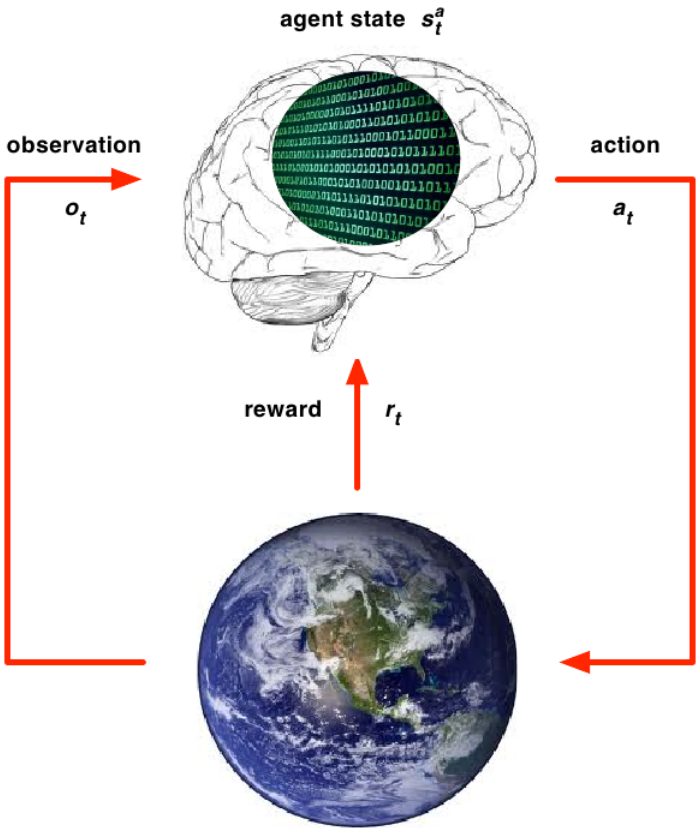
Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL



- The agent state is the agent's internal representation
  - whatever information the agent uses to **select** the next action
  - is the information used by RL agents
- It can be any function history: $s_t^a = f(h_t)$

Fully Observable Environments

Marcello Restelli

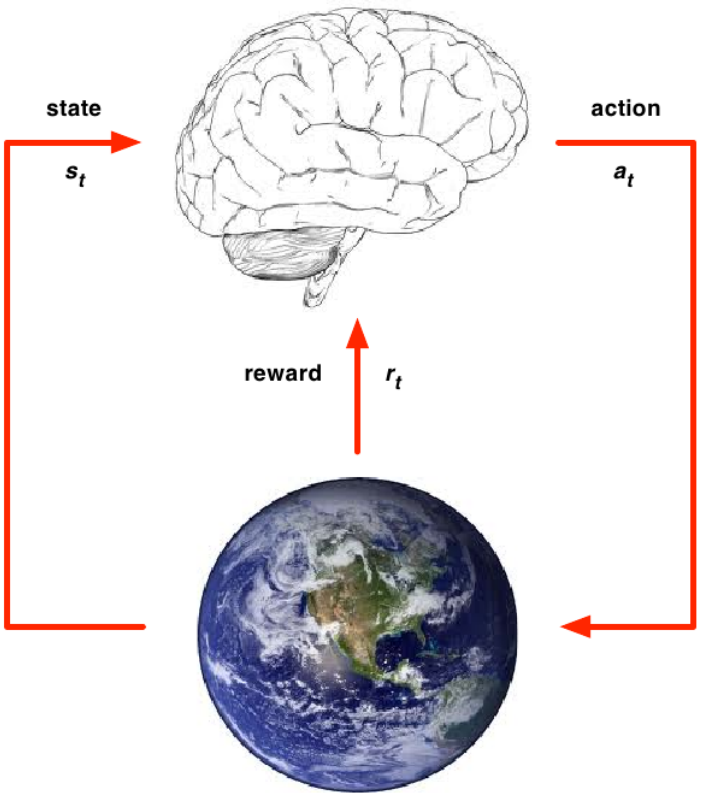
Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL



- **Full observability**: agent **directly** observes environment state

$$o_t = s_t^a = s_t^e$$

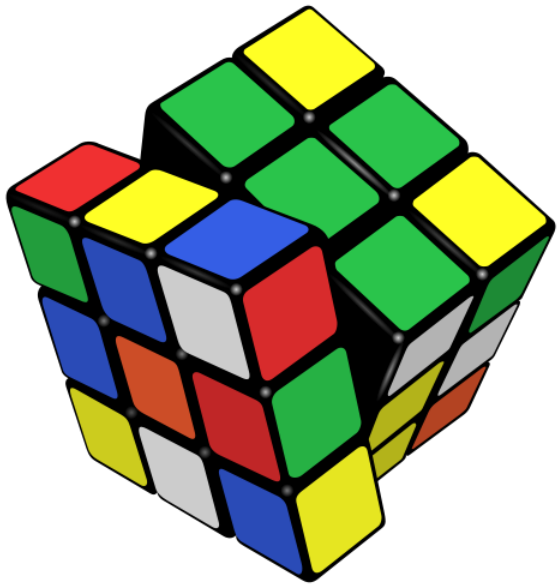
- Formally, this is a **Markov Decision Process** (MDP)
- The majority of this course will consider the MDP case

When is RL useful?

- When the dynamics of the environment are **unknown** or difficult to be modeled
  - e.g., trading, betting
- When the model of the environment is too **complex** to be solved exactly, so that **approximate** solutions are searched for
  - e.g., humanoid robot control, group elevator dispatching

Example 1: Rubik's Cube

- Invented in 1974 by Ernõ Rubik
- Formalization
  - State space: $\sim 4.33 \times 10^{19}$
  - Actions: 12 for each state
  - Deterministic state transitions
  - Rewards: -1 for each step
  - Undiscounted
- The cube can be solved in 20 moves or fewer



Example 2: Blackjack

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

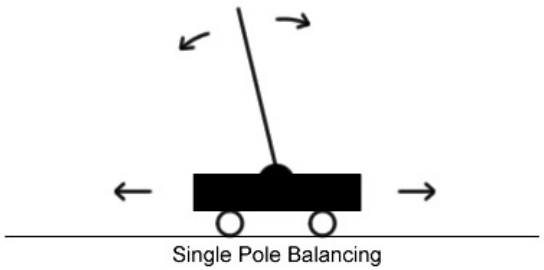
Modeling the Problem

Algorithmic Solutions in RL

- The most played casino game
- Formalization
  - State space: totals ~ 800, composition $\sim 104,000$
  - Actions: from 2 to 4 according to the state
  - Stochastic state transitions
  - Rewards: 0 for each step, $\{-2,-1,0,1,1.5,2\}$ at the end
  - Undiscounted
- Using the optimal policy, the house edge is very low ($\sim 0,4\% - 0,7\%$)

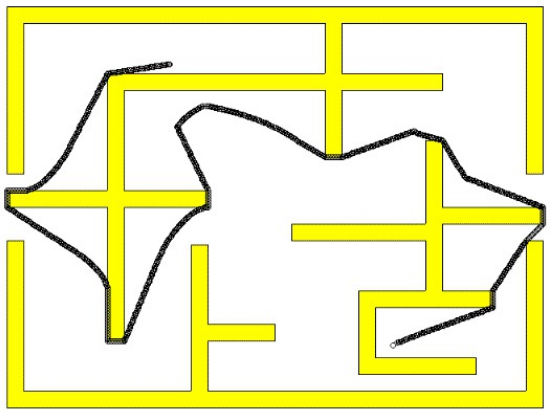
Example 3: Pole balancing

- A classical RL benchmark
- Formalization
  - State space: four continuous state variables $x, \dot{x}, \theta, \dot{\theta}$
  - Actions: two actions $\{-N, N\}$
  - Deterministic state transitions
  - Rewards:
    - 0 when in the goal region
    - -1 when outside goal region
    - -100 when outside feasible region



Example 4: Robot Navigation

- The most important task in mobile robotics
- Formalization
  - State space: robot coordinates
  - Actions: moving actions
  - Stochastic state transitions
  - Rewards: -1 until goal is reached
  - Undiscounted/discounted
- Often the state cannot be observed
- Shift to POMDP framework



Example 5: Web Banner Advertising

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

- We have to choose which banner ad showing in a certain slot of our web page
- Formalization
  - State space: single state or multiple states (contexts)
  - Actions: one for each banner
  - No dynamics
  - Rewards: probability of click times the cost per click
- Multi-armed bandit: exploration vs exploitation

Example 6: Chess

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

- Very popular board game
- Formalization
  - State space: $\sim 10^{47}$
  - Actions: from 0 to 218
  - Deterministic opponent-dependent state transitions
  - Rewards: 0 each step, $\{-1, 0, 1\}$ at the end
  - Undiscounted
- The size of the game tree is 10^{123}

Example 7: Texas Hold'em

- Recently, the most played poker version
- Formalization
  - State space: huge, and not observable
  - Actions: fold, call, and raise
  - Stochastic opponent-dependent state transitions
  - Rewards: 0 each step, $\{-1, 0, 1\}$ at the end
  - Undiscounted
- The size of the limit game with 2 players is 10^{18}

Discrete vs Continuous Decision Problems

Function approximation

- State-action space is discrete: **tabular** approach
- Many state-action pairs (curse of dimensionality) or continuous state–action domains: **function approximation**
- Continuous **state** space
  - Linear function approximation, ANN, SVM, regression trees, etc.
  - Convergence and stability issues
  - Bootstrap
- Continuous **action** space
  - Optimization problem
  - Splitting policy and value function
- Continuous **time**
  - Hamilton-Jacobi-Bellman equation
  - Semi-MDP

Fully Observable vs Partially Observable MDPs

POMDPs

Marcello Restelli

Course Information

What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

- In many problems, the system state cannot be fully observed.
- Partially Observable MDPs (POMDPs) model such situations
- $\langle S, A, O, P, \Omega, R \rangle$
  - O is a set of observations
  - Ω is a set of conditional observation probabilities
- **Belief MDP** $\langle B, A, \tau, r \rangle$
  - $b'(s') = \eta \Omega(o|s', a) \sum_{s \in S} P(s'|s, a) b(s)$
  - The belief space is **continuous**
- POMDPs are often **computationally intractable**
  - grid-based algorithms
  - sampling techniques

Stationary vs Non-stationary MDPs

Multi-agent learning

- In many real–world problems, transition dynamics and the reward function may be **time dependent**.
- In the **cyclostationary** case, time can be added to the state space
- What happens in presence of **other agents**?
  - if other agents' policies are stationary, we can ignore them
  - if other agents are learning, things get interesting...
- **multi-agent learning** is much more complex than single-agent learning
  - strictly related to Game Theory
  - optimal policy is replaced by best response and equilibrium policy
  - competitive agents
  - cooperative agents

Discounted vs Average Reward MDPs

The infinite horizon case

- **Discounted** MDPs
  - most RL algorithms have been designed for this framework
  - some algorithms provably converge to the optimal solution (e.g., Q-learning)
- **Average-reward** MDPs
  - mathematically are more complex
  - good algorithms have been proposed (e.g., R-learning), but no proof of convergence to the optimal solution has been produced so far

Single-objective vs Multi-objective MDPs

MOMDPs

Marcello Restelli

Course Information

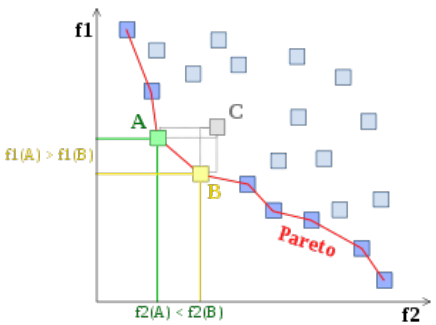
What is Reinforcement Learning?

Sequential Decision Problem Examples

Modeling the Problem

Algorithmic Solutions in RL

- In some cases agents may have **multiple objectives** (i.e., reward functions)
- According to the **importance** given to the objectives the optimal policy may change
- The goal is to find the policies on the **Pareto frontier**



  - each policy has a performance value for each objective
  - if a policy performs worse than another for each objective, it is said to be **dominated**
  - the Pareto frontier is the set of **non–dominated** solutions
- The solution is a **set** of (eventually infinite) policies

MDP without Reward

Intrinsically motivated and Inverse Reinforcement learning

- It may happen that the reward function is **not available**
- This framework is interesting in two scenarios:
  - Intrinsically motivated learning

    - Is that kind of learning that allows puppies and babies to **autonomously develop skills**
    - The reward signal is **not extrinsic**, but is produced by the learning algorithm itself according to the environmental characteristics
  - Inverse reinforcement learning

    - In many real-world applications, we do not want an agent to learn from scratch, performing random exploring actions
    - By observing the behavior of an **expert**, we want to **infer** which is the reward function she is optimizing
    - It is related to **imitation learning**

Model–free vs Model–based

- In both cases, no prior knowledge about the model is available
- **Model–based** methods
  - explicitly estimate a model from experience
  - use dynamic-programming algorithms on the approximated model
  - effective use of experience
  - high computational costs
- **Model–free** methods
  - directly learn the solution
  - low memory and computational costs
  - no guarantees about explore/exploit trade-off
  - learning may be slower

On–policy vs Off–policy

- The learned policy may be different from the behavior policy
- **On–policy** learning (e.g., SARSA)
  - the behavior policy is the learned policy
  - policy evaluation and policy improvement happen simultaneously
  - empirically have been proved good with function approximation
  - require well-designed exploration functions
- **Off-policy** learning (e.g., Q-learning)
  - learning of the optimal policy independently of its execution
  - possibility to use effective exploration strategies
  - learns faster
  - at each step requires to maximize over actions
  - problem with function approximation and eligibility traces

Exploration/Exploitation Trade–off

- One of the most important topics in RL research
- Several theoretical studies to achieve efficient algorithms (bandit–arm algorithms)
  - R-max: initialize optimistically
  - E^3: tries to reach unknown states
  - Model–based interval estimation
  - Bayesian RL
- Particularly critical in multiagent systems

Online vs Offline

- It is just a matter of learning over the **entire dataset** (offline) versus learning **incrementally**
- Supervised learning was mainly offline, recently is going online
- Reinforcement learning was mainly online, recently is going offline
- Offline RL (e.g., fitted Q–iteration) has several **interesting features**
  - can avoid bootstrap problems
  - works well with function approximation
  - learns with very few experience samples
- ... and some **problems**
  - how to collect samples
  - how to use new samples

Value-based vs Policy-based

- **Value-based** methods (e.g., Q-learning, SARSA)
  - the optimal (action-)value function
  - the policy is implicitly stored in the value function
  - unfortunately good value function approximations may correspond to bad policies and vice versa
- **Policy-search** methods
  - directly search in the policy space
  - defines the RL problem as an optimization problem
    - gradient-based algorithms
    - stochastic optimization: simulated annealing, cross entropy, evolutionary computation
  - the performance of each policy is estimated via simulation
  - the policy space is much larger than the value-function space
  - negatively affect by noisy environments
  - effective when simulation is fast and some prior knowledge on the optimal policy is available
- **Actor-critic** methods
  - the policy is implemented by the actor
  - the value is stored in the critic

Flat vs Hierarchical Learning

- Usually, RL algorithms are flat: all the actions have the same complexity
- In real–world applications we need to consider actions with different complexity (and duration)
- **Hierarchical** RL studies how to learn using actions arranged into hierarchies
  - problems formalized as SMDPs
  - several algorithms: Options, HAMs, MAX-Q
  - exploit hierarchy structure to speed-up learning
- Autonomous **sub-goal discovery**
  - the action hierarchy is learned
  - usually produces skills to reach interesting states, e.g., bottlenecks

Learning from Scratch vs Reuse of Knowledge

- Usually, RL algorithms are designed to learn from scratch
- Animals and human beings never learn from scratch
- New tasks can be solved by **reusing** knowledge learned when solving similar tasks
- **Transfer** Learning
- What kind of knowledge can be transferred?
  - value functions
  - policies
  - experience samples
- Problems
  - when two tasks are similar?
  - how to prevent negative transfer?