Lecture 1: Introduction to RL

Ciprian Paduraru

- Based on the following courses :
  - RL Courses from Stanford CS234 and Waterloo 885
  - Deep RL Course from Berkeley 285

Today's Plan

- Overview of reinforcement learning
- Course logistics
- Introduction to sequential decision making under uncertainty

Reinforcement Learning

- The fundamental challenge in artificial intelligence and machine learning is learning to make good decisions under uncertainty

- Applications of RL:

  - Self driving cars
  - Recommendation systems
  - Games
  - Robotics
  - Planning
  - Etc..

➢ Key concepts in an RL problem

- Optimization
- Delayed consequences
- Exploration
- Generalization

➢ Plan:

- ❖ Explore the world
- ❖ Use experience to guide future decisions

Optimization

- Goal is to find an optimal way to make decisions
  - Yielding best outcomes or at least very good outcomes
- We need an explicit notion of utility of decisions
- Example: finding minimum distance route between two cities given network of roads

Delayed

- Decisions now can impact things much later...
  - Saving for retirement
  - Finding a key in video game Montezuma's revenge
- Introduces two challenges
  - When planning: decisions involve reasoning about not just immediate benefit of a decision but also its longer term ramifications
  - When learning: temporal credit assignment is hard (what caused later high or low rewards?)

Exploration

- Learning about the world by making decisions
  - Agent explores the world
  - E.g. Learn to ride a bike by trying (and failing)
  - Finding a key in Montezuma's revenge
- Censored data
  - Only get a reward (label) for decision made
  - Don't know what would have happened if we had taken red pill instead of blu pill (Matrix movie reference)
- Decisions impact what we learn about
  - If we choose to go to Stanford instead of MIT, we will have different later experiences...

Generalizatio

- Policy is mapping from past experience to action
- Why not just pre-program a policy?

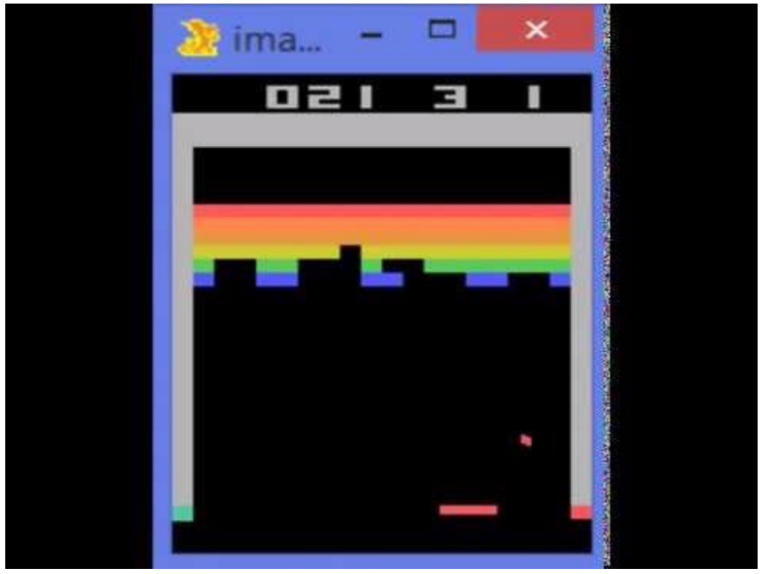


Figure:DeepMind Nature, 2015

- How many possible images are there?
  - $\left(256^{100\times200}\right)^{3}$

RL vs Other AI and Machine Learning

	AI Planning	SL	UL	RL	IL
Optimization	X				
Learns from experience					
Generalization	X				
Delayed Consequences	X				
Exploration					

- SL = Supervised learning; UL = Unsupervised learning; RL = Reinforcement Learning; IL = Imitation Learning
- AI planning assumes have a model of how decisions impact environment

RL vs Other AI and Machine Learning

	AI Planning	SL	UL	RL	IL
Optimization	X				
Learns from experience		X			
Generalization	X	X			
Delayed Consequences	X				
Exploration					

- SL = Supervised learning; UL = Unsupervised learning; RL = Reinforcement Learning; IL = Imitation Learning
- Supervised learning is provided correct labels

RL vs Other AI and Machine Learning

	AI Planning	SL	UL	RL	IL
Optimization	X				
Learns from experience		X	X		
Generalization	X	X	X		
Delayed Consequences	X				
Exploration					

- SL = Supervised learning; UL = Unsupervised learning; RL = Reinforcement Learning; IL = Imitation Learning
- Unsupervised learning is provided no labels

RL vs Other AI and Machine Learning

	AI Planning	SL	UL	RL	IL
Optimization	X			X	
Learns from experience		X	X	X	
Generalization	X	X	X	X	
Delayed Consequences	X			X	
Exploration				X	

- SL = Supervised learning; UL = Unsupervised learning; RL = Reinforcement Learning; IL = Imitation Learning
- Reinforcement learning is provided with censored labels

Sidenote: Imitation Learning

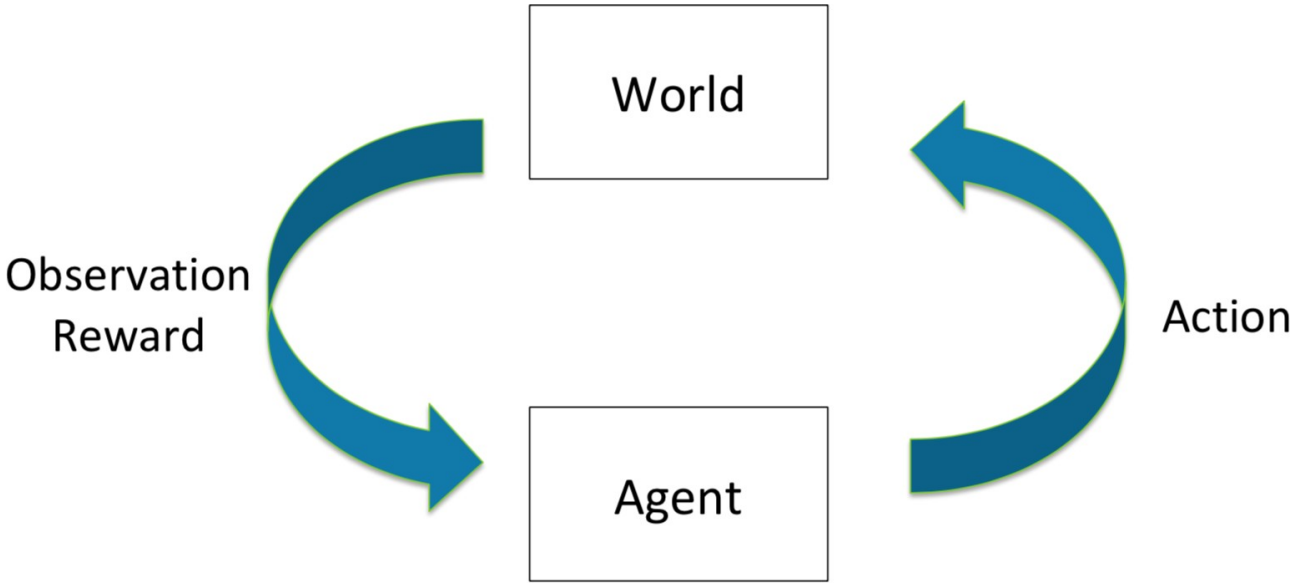
	AI Planning	SL	UL	RL	IL
Optimization	X			X	X
Learns from experience		X	X	X	X
Generalization	X	X	X	X	X
Delayed Consequences	X			X	X
Exploration				X	

- SL = Supervised learning; UL = Unsupervised learning; RL = Reinforcement Learning; IL = Imitation Learning
- Imitation learning assumes input demonstrations of good policies
- IL reduces RL to SL. IL + RL is promising area

High Level Learning Goals from this course

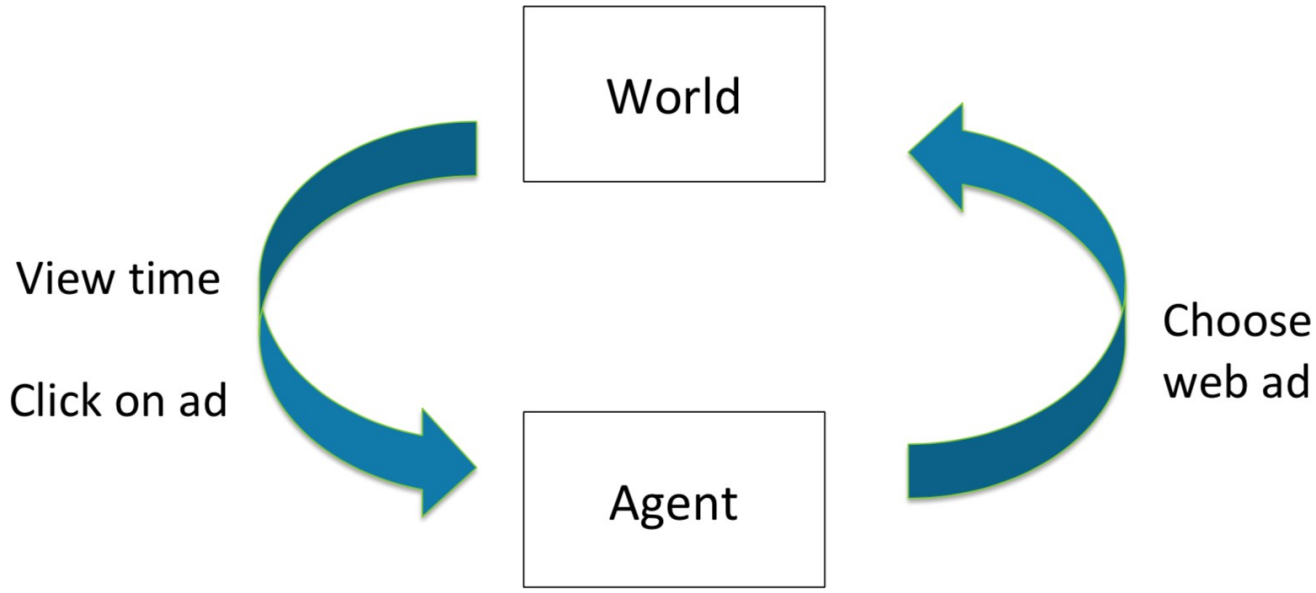
- Define the key features of RL
- Given an application probem how (and whether) to use RL for it
- Compare and contrast RL algorithms on multiple criteria
- Define a RL environment and Implement an algorithm that drives the optimizations and inference properly.

Sequential Decision Making



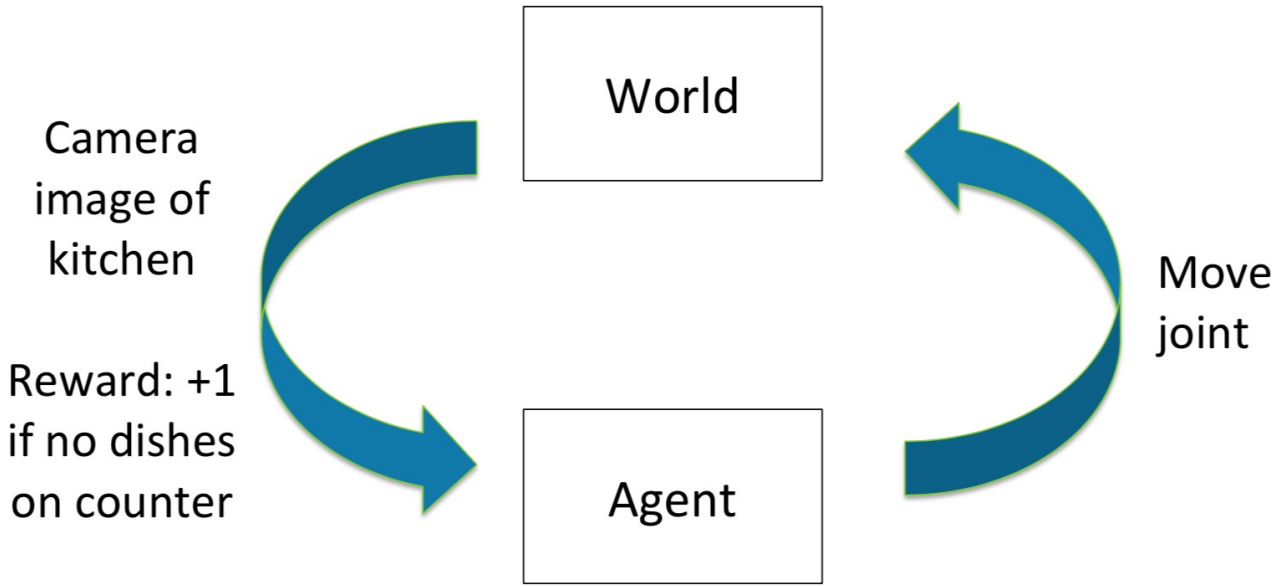
- Goal: Select actions to maximize total expected futurereward
- May require balancing immediate & long term rewards

Example: Web Advertising



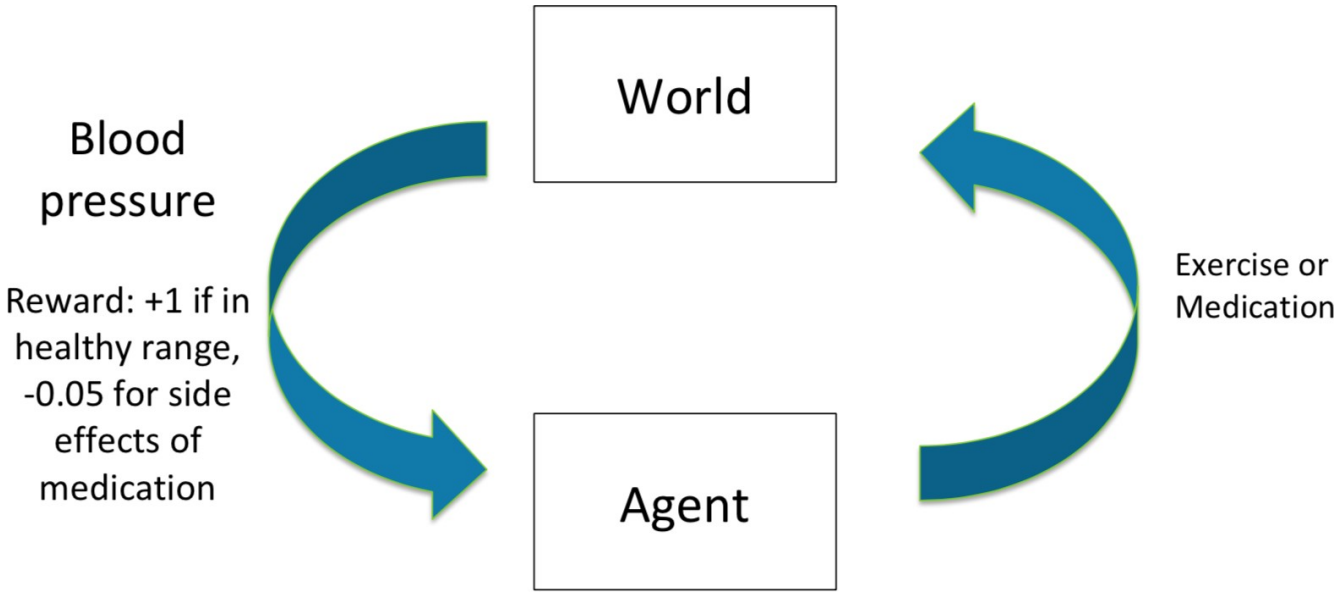
- Goal: Select actions to maximize total expected futurereward
- May require balancing immediate & long term rewards

Example: Robot Unloading Dishwasher



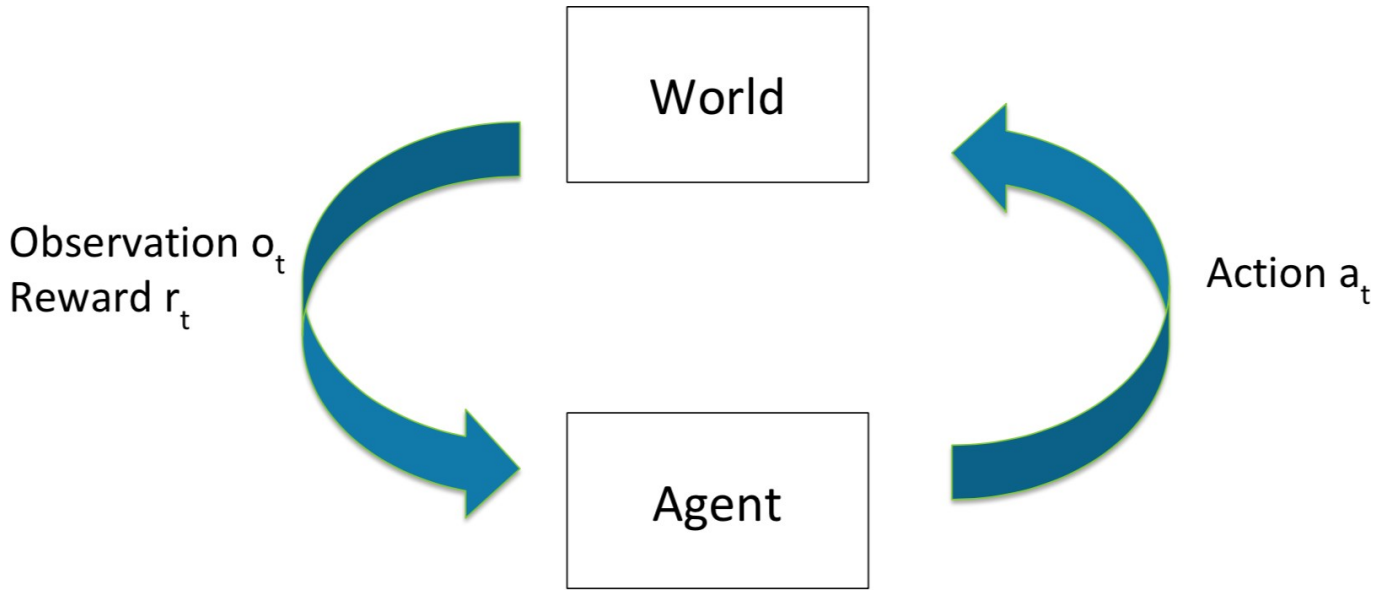
- Goal: Select actions to maximize total expected futurereward
- May require balancing immediate & long term rewards

Example: Blood Pressure Control



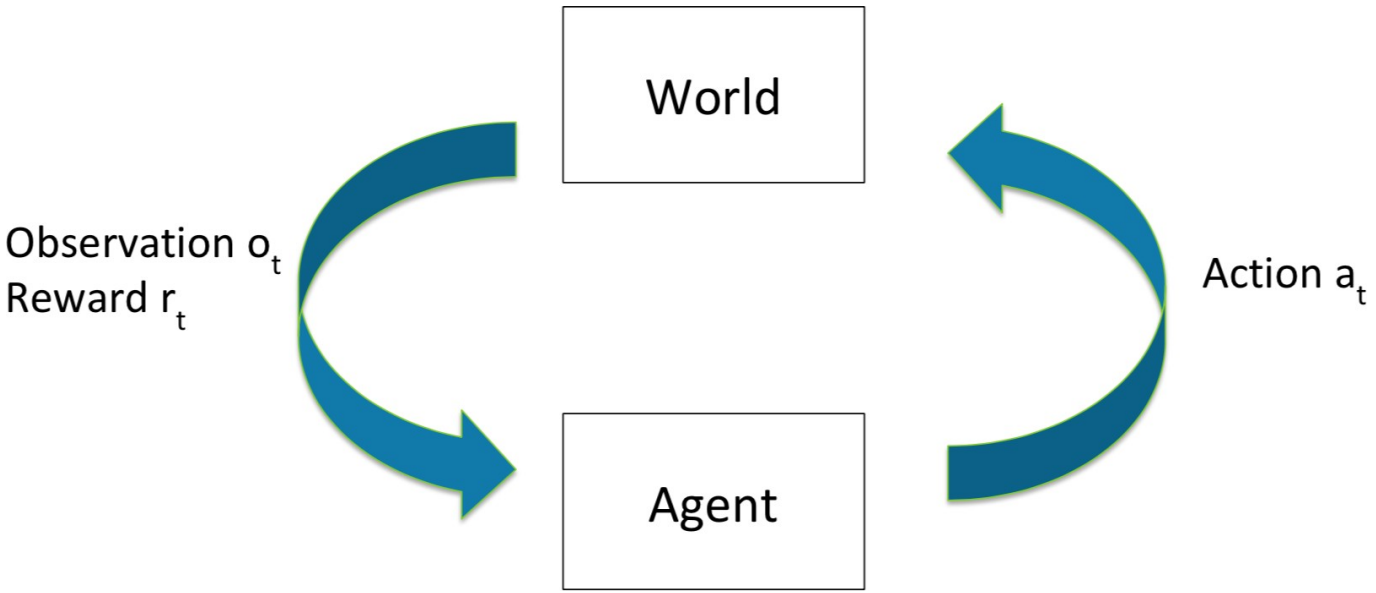
- Goal: Select actions to maximize total expected futurereward
- May require balancing immediate & long term rewards

Sequential Decision Process: Agent & the World (Discrete Time)



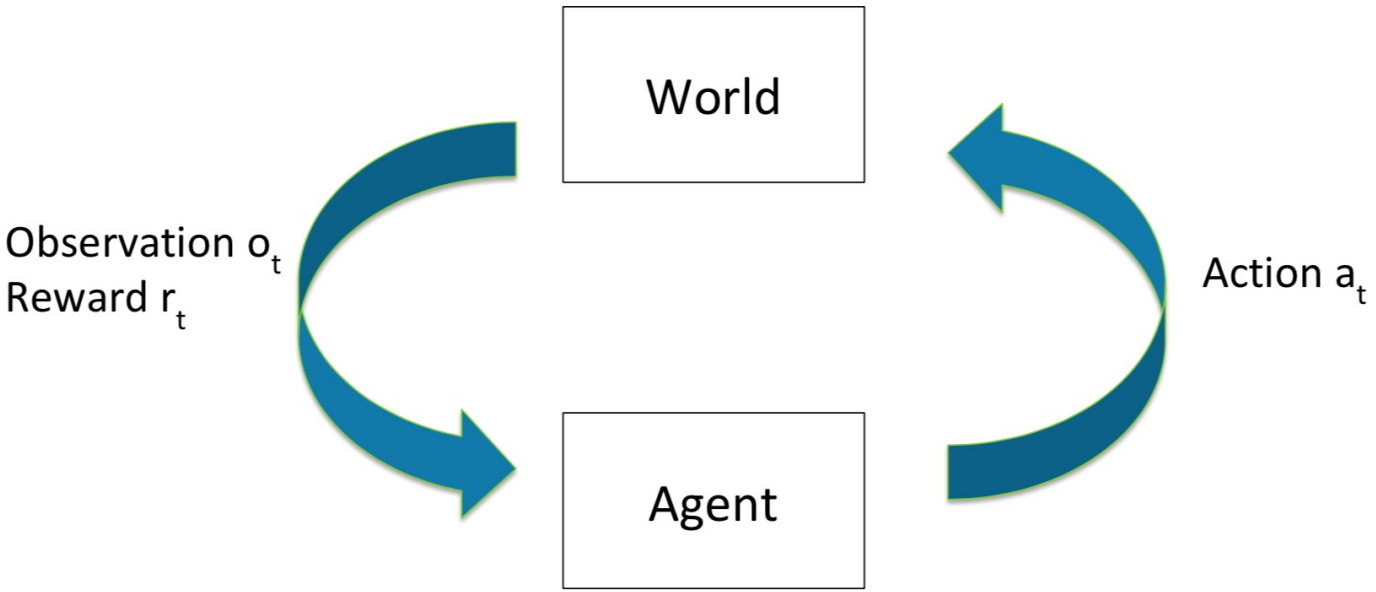
- Each time step t:
  - Agent takes an action a_t
  - World updates given action a_t, emits observation o_t and reward r_t
  - Agent receives observation o_t and reward r_t

History: Sequence of Past Observations, Actions & Rewards



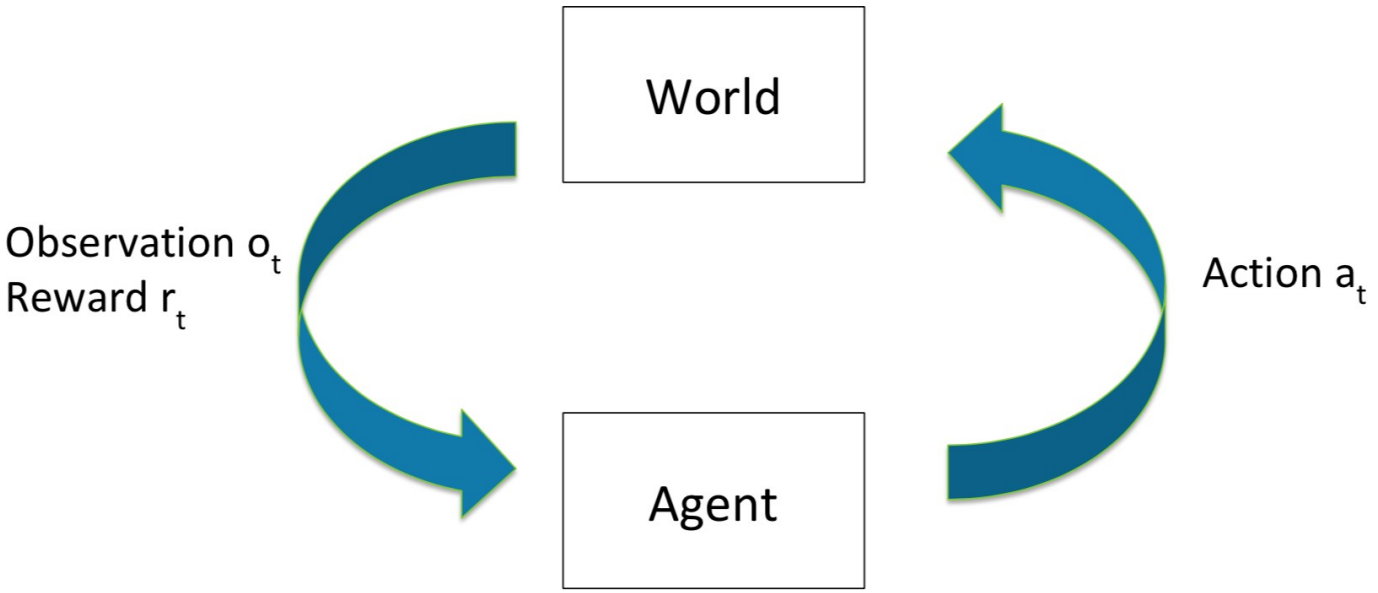
- History $h_t = (a_1, o_1, r_1, \ldots, a_t, o_t, r_t)$
- Agent chooses action based on history
- State is information assumed to determine what happens next
  - Function of history: $s_t = (h_t)$

World State



- This is true state of the world used to determine how world generates next observation and reward
- Often hidden or unknown to agent
- Even if known may contain information not needed by agent

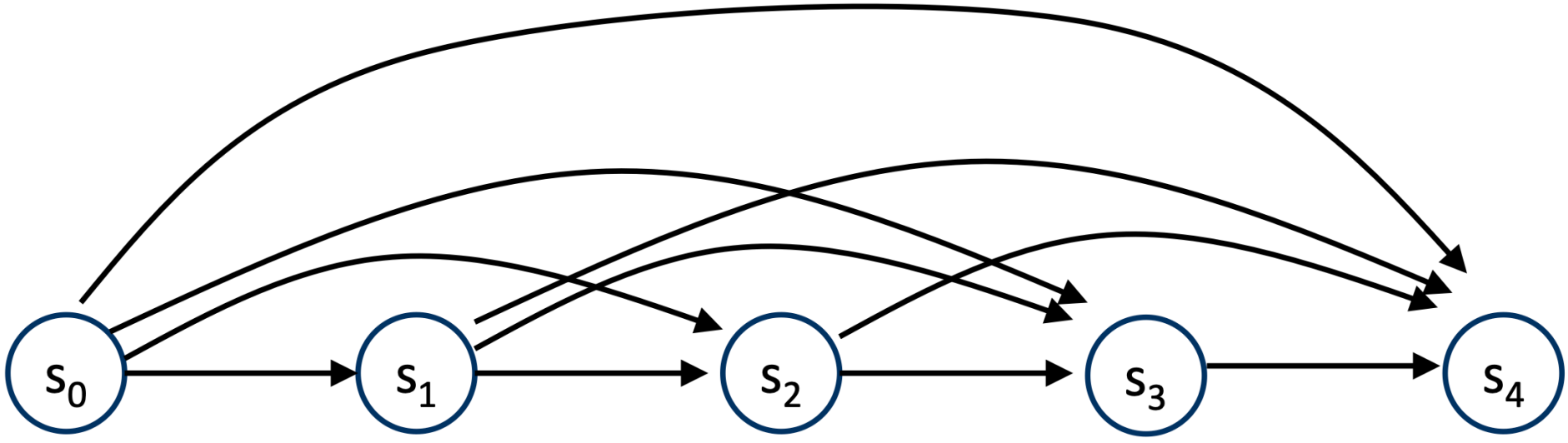
Agent State: Agent's Internal Representation



- What the agent / algorithm uses to make decisions about how to act
- Generally a function of the history: $s_t = f(h_t)$
- Could include meta information like state of algorithm (how many computations executed, etc) or decision process (how many decisions left until an episode ends)

Stochastic Process

- Consider the sequence of states only
- Definition
  - Set of States: S
  - Stochastic dynamics: $Pr(s_t | s_{t-1}, \ldots, s_0)$

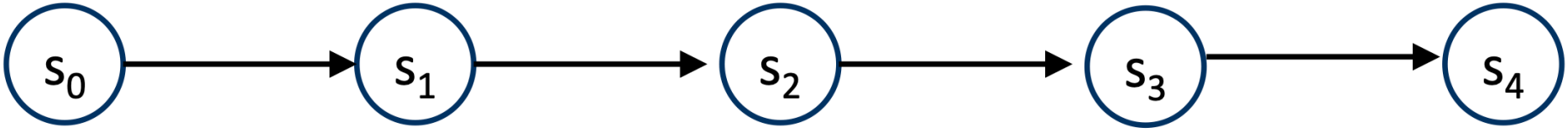


Stochastic Process

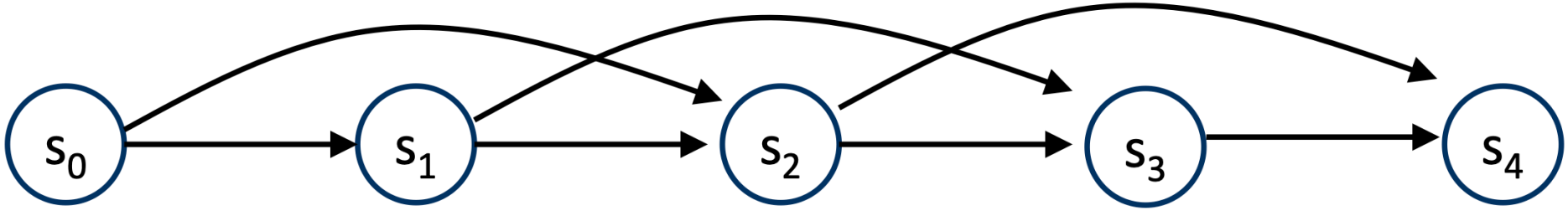
- Problem:
  - Infinitely large conditional distributions
- Solutions:
  - Stationary process: dynamics do not change over time
  - Markov assumption: current state depends only on a finite history of past states

K-order Markov Process

- Assumption: last k states sufficient
- First-order Markov Process
  - $Pr(s_t | s_{t-1}, \ldots, s_0) = Pr(s_t | s_{t-1})$



- Second-order Markov Process
  - $Pr(s_t | s_{t-1}, \ldots, s_0) = Pr(s_t | s_{t-1}, s_{t-2})$



Markov Process

- By default, a Markov Process refers to a
  - First-order process

    $$\Pr(s_t|s_{t-1}, s_{t-2}, \ldots, s_0) = \Pr(s_t|s_{t-1}) \ \forall t$$
  - Stationary process

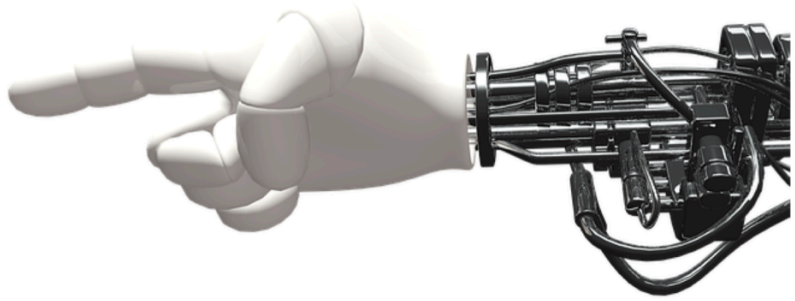
    $$\Pr(s_t|s_{t-1}) = \Pr(s_{t'}|s_{t'-1}) \, \forall t'$$

- Advantage: can specify the entire process with a single concise conditional distribution

  $$\Pr(s'|s)$$

Examples

- Robotic control
  - **States:** $\langle x, y, z, \theta \rangle$ coordinates of joints
  - **Dynamics:** constant motion



- Inventory management
  - **States:** inventory level
  - **Dynamics:** constant (stochastic) demand

Non-Markovian and/or non-stationary processes

- What if the process is not Markovian and/or not stationary?
- Solution: add new state components until dynamics are Markovian and stationary
  - Robotics: the dynamics of $\langle x, y, z, \theta \rangle$ are not stationary when velocity varies…
  - Solution: add velocity to state description e.g. $\langle x, y, z, \theta, \dot{x}, \dot{y}, \dot{z}, \dot{\theta} \rangle$
  - If acceleration varies… then add acceleration to state
  - Where do we stop?

Markovian Stationary Process

- **Problem:** adding components to the state description to force a process to be Markovian and stationary may significantly increase computational complexity

- **Solution:** try to find the smallest state description that is self-sufficient (i.e., Markovian and stationary)

Why is Markov Assumption Popular?

- Can always be satisfied
  - Setting state as history always Markov: $s_t = h_t$
- In practice often assume most recent observation is sufficient statistic of history: $s_t = o_t$
- State representation has big implications for:
  - Computational complexity
  - Data required
  - Resulting performance

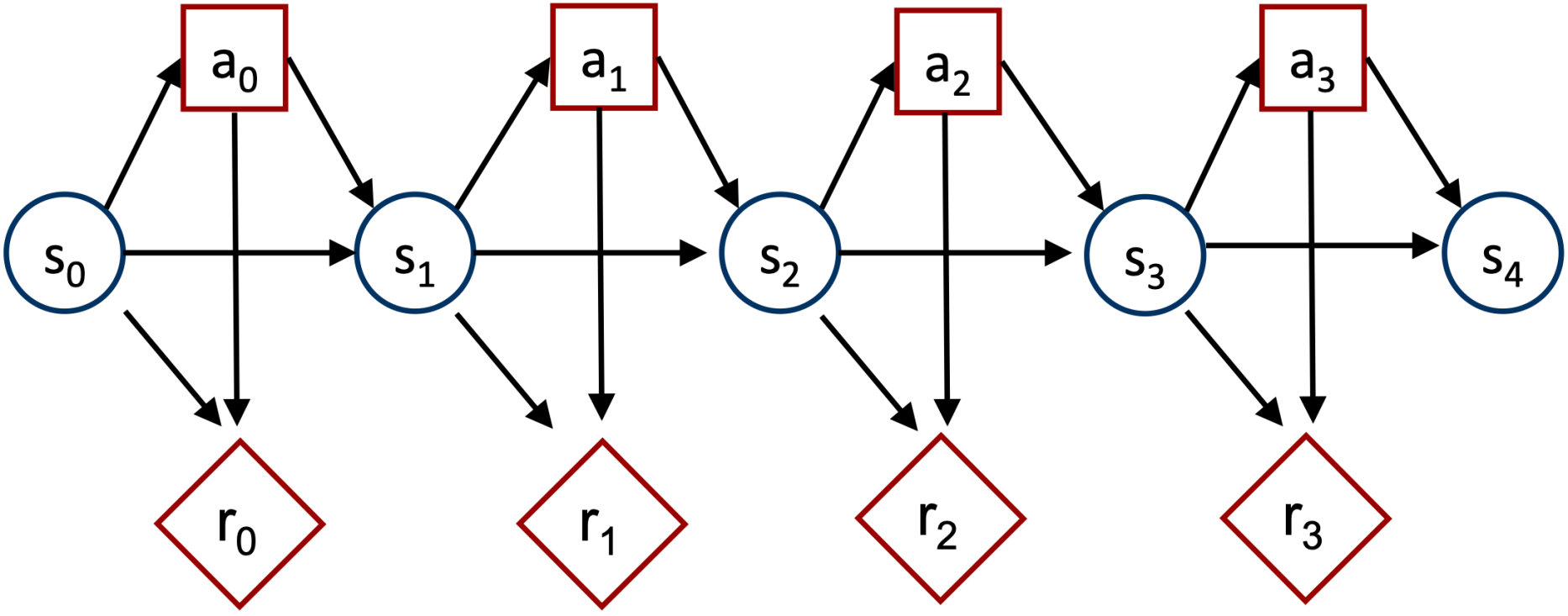
Decision Making

- Predictions by themselves are useless
- They are only useful when they will influence future decisions

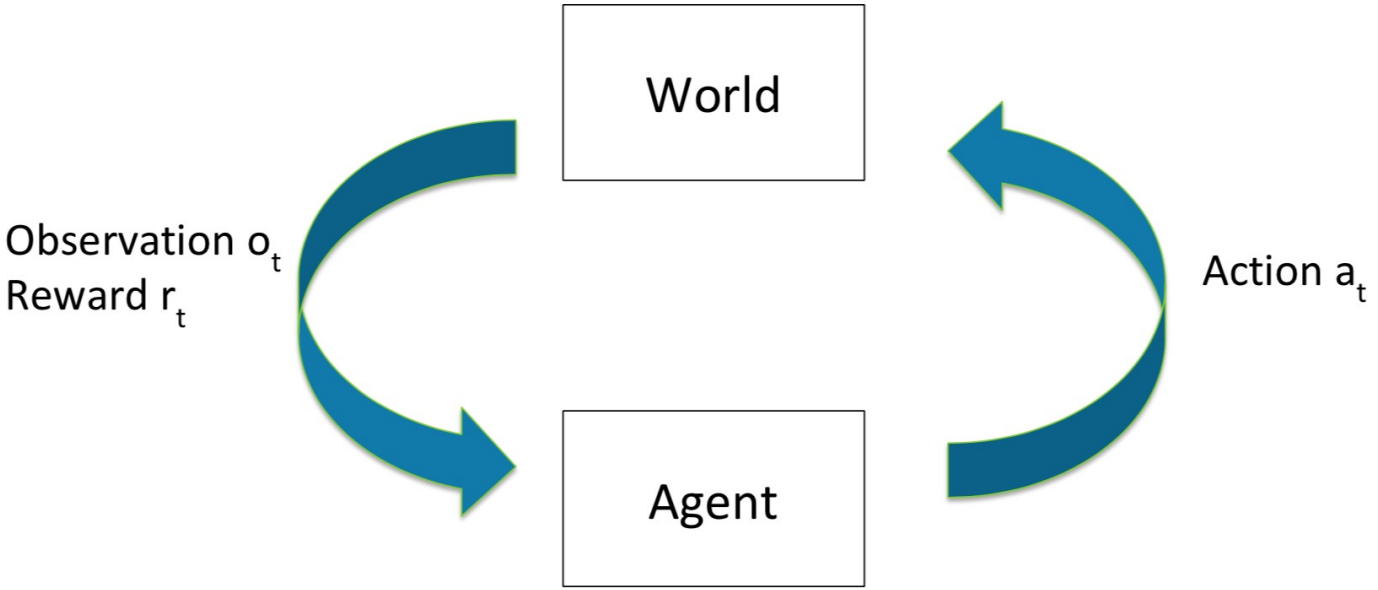
- Hence the ultimate task is decision making
- How can we influence the process to visit desirable states?
  - Model: Markov Decision Process

Markov Decision Process

- Markov process augmented with…
  - Actions e.g., a_t
  - Rewards e.g., r_t

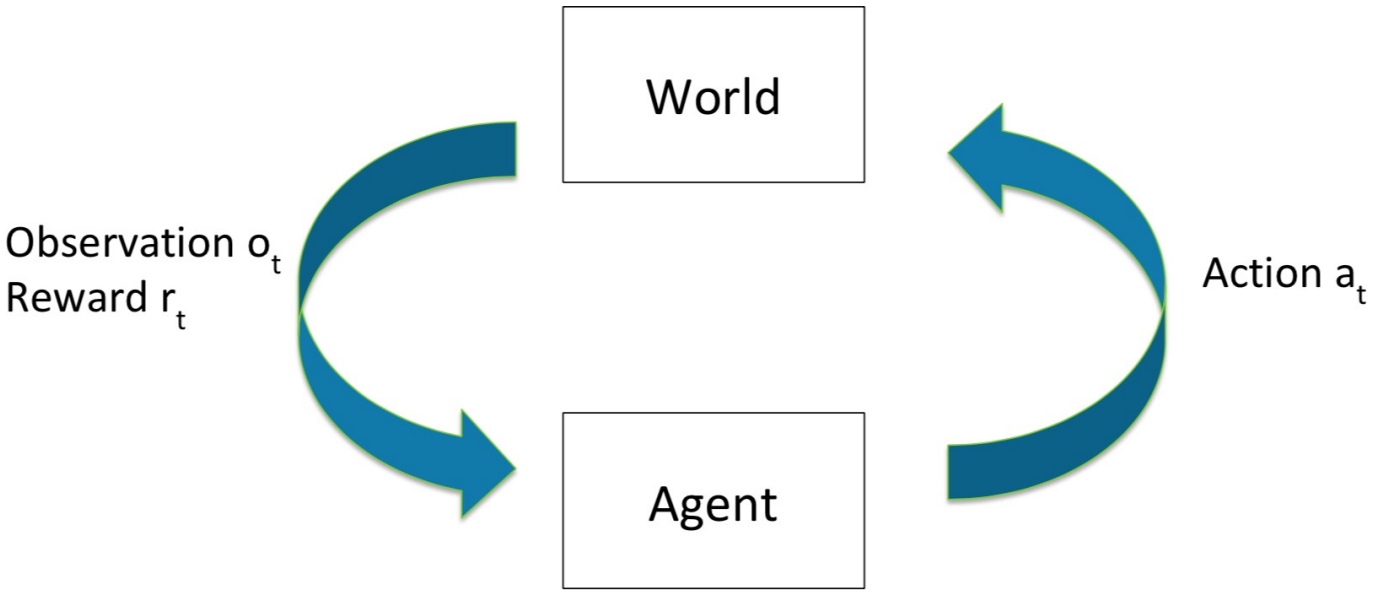


Full Observability / Markov Decision Process (MDP)



- Environment and world state $s_t = o_t$

Types of Sequential Decision Processes



- Is state Markov? Is world partially observable? (POMDP)
- Are dynamics deterministic or stochastic?
- Do actions influence only immediate reward or reward and next state?

- Algorithms includes one or more of: **Model**, Policy, Value Function

Model

- Agent's representation of how world changes given agent's action
- Transition / dynamics model predicts next agent state

$$p(s_{t+1} = s' | s_t = s, a_t = a)$$

- Reward model predicts immediate reward

$$r(s_t = s, a_t = a) = \mathbb{E}[r_t | s_t = s, a_t = a]$$

MDP Model

- Agent's representation of how world changes given agent's action
- Transition / dynamics model predicts next agent state

$$p(s_{t+1} = s^j \mid s_t = s,\ a_t = a)$$

- Reward model predicts immediate reward

$$r(s_t = s,\ a_t = a) = \mathbb{E}[r_t \mid s_t = s,\ a_t = a]$$

Example: Mars Rover Stochastic Markov Model

s_1	s_2	s_3	s_4	s_5	s_6	s_7
$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$	$\hat{r} = 0$

- Numbers above show RL agent's reward model
- Part of agent's transition model:
  - $0.5 = P(s_1|s_1, \text{TryRight}) = P(s_2|s_1, \text{TryRight})$
  - $0.5 = P(s_2|s_2, \text{TryRight}) = P(s_3|s_2, \text{TryRight}) \cdots$
- Model may be wrong

Policy

- Policy π determines how the agent chooses actions
- $\pi : S \rightarrow A$, mapping from states to actions
- Deterministic policy:

$$\pi(s) = a$$

- Stochastic policy:

$$\pi(a|s) = Pr(a_t = a|s_t = s)$$

Example: Mars Rover Policy

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

- $\pi(s_1) = \pi(s_2) = \cdots = \pi(s_7) = \text{TryRight}$
- Quick check: is this a deterministic policy or a stochastic policy?

Value Function

- Value function V^{π}: expected discounted sum of future rewards under a particular policy π

$$V^{\pi}(s_t = s) = \mathbb{E}_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots | s_t = s]$$

- Discount factor γ weighs immediate vs future rewards
- Can be used to quantify goodness/badness of states and actions
- And decide how to act by comparing policies

Example: Mars Rover Value Function

s_1	s_2	s_3	s_4	s_5	s_6	s_7
$V^{\pi}(s_1) = +1$	$V^{\pi}(s_2) = 0$	$V^{\pi}(s_3) = 0$	$V^{\pi}(s_4) = 0$	$V^{\pi}(s_5) = 0$	$V^{\pi}(s_6) = 0$	$V^{\pi}(s_7) = +10$

- Discount factor, $\gamma = 0$
- $\pi(s_1) = \pi(s_2) = \cdots = \pi(s_7) =$ TryRight
- Numbers show value $V^{\pi}(s)$ for this policy and this discount factor

Types of RL Agents

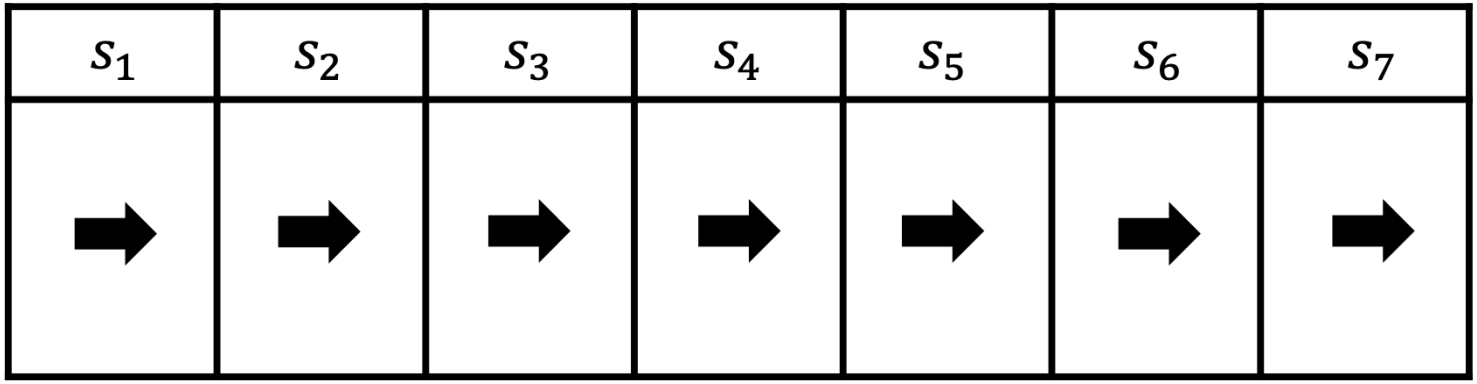
- Model-based
  - Explicit: Model
  - May or may not have policy and/or value function
- Model-free
  - Explicit: Value function and/or policy function
  - No model

Two main steps: Evaluation and Control

- Evaluation
  - Estimate/predict the expected rewards from following a given policy
  -
- Control

  Optimization: find the best policy

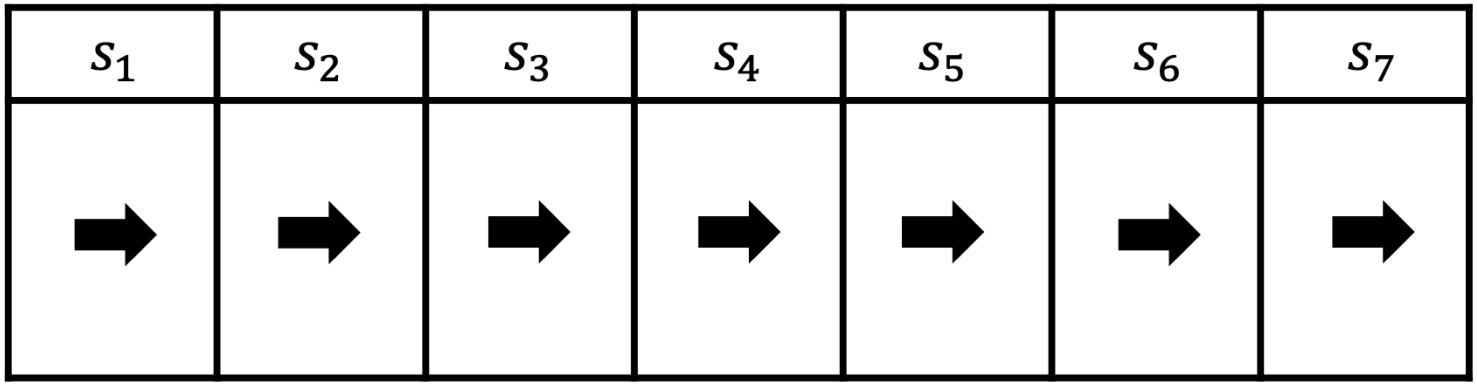
Example: Mars Rover Policy Evaluation



- $\pi(s_1) = \pi(s_2) = \cdots = \pi(s_7) = \text{TryRight}$
- Discount factor, $\gamma = 0$
- What is the value of this policy?

$$V^{\pi}(s_t = s) = \mathbb{E}_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots | s_t = s]$$

Example: Mars Rover Policy Control



- Discount factor, $\gamma = 0$
- What is the policy that optimizes the expected discounted sum of rewards?

Course Outline

- Markov decision processes & planning
- Model-free policy evaluation
- Model-free control
- Reinforcement learning with function approximation & Deep RL
- Policy Search
- Exploration
- Advanced Topics

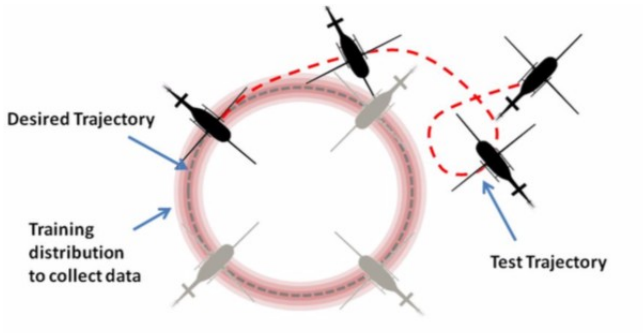
Evaluation

- In week 4 I will post a list of possible projects.
- In teams of 2-5 people, you will implement one of them.
- No written exam, just practical code implementation.

Imitation Learning

Figure:Abbeel, Coates and Ng helicopter team, Stanford

Imitation Learning



Ross & Bagnell 2013

- Reduces RL to supervised learning
- Benefits
  - Great tools for supervised learning
  - Avoids exploration problem
  - With big data lots of data about outcomes of decisions
- Limitations
  - Can be expensive to capture
  - Limited by data collected
- Imitation learning + RL promising!

Expanding Reach. NLP, Vision, ...

Figure:Yeung, Russakovsky, Mori, Li 2016.