Lecture 1: Introduction to RL

Emma Brunskill

CS234 RL

Winter 2019

- Today the 3rd part of the lecture is based on David Silver's introduction to RL slides

Today's Plan

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

Reinforcement Learning

Learn to make good sequences of decisions

Repeated Interactions with World

Learn to make good **sequences of decisions**

Reward for Sequence of Decisions

Learn to make **good** sequences of decisions

Don't Know in Advance How World Works

Learn to make good sequences of decisions

Fundamental challenge in artificial intelligence and machine learning is learning to make good decisions under uncertainty

RL, Behavior & Intelligence

Figure: Example from Yael Niv

- Childhood: primitive brain & eye, swims around, attaches to a rock
- Adulthood: digests brain, sits
- Suggests brain is helping guide decisions (no more decisions, no need for brain?)

Atari

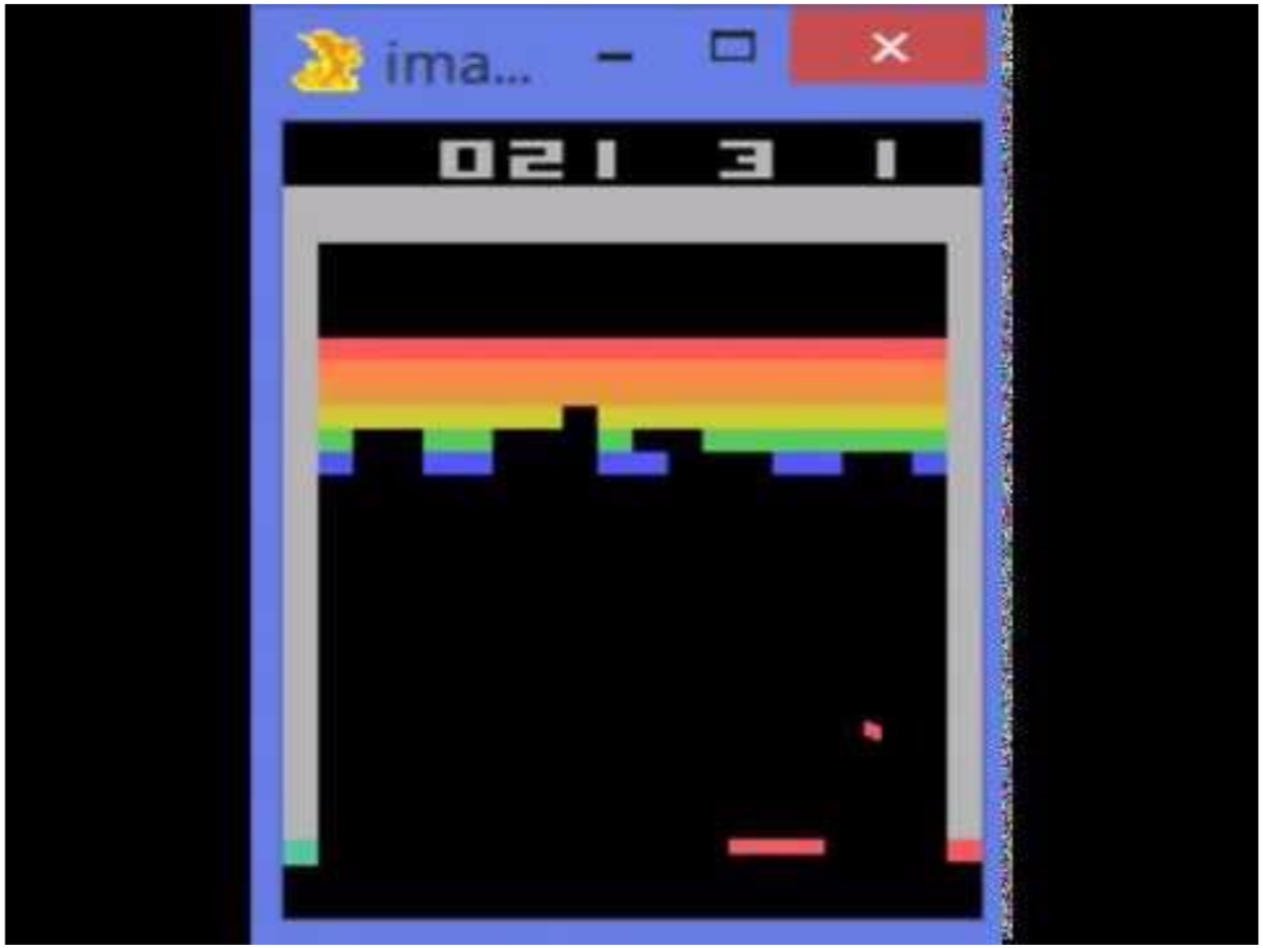


Figure: DeepMind Nature, 2015

Robotics

Educational Games

Figure: RL used to optimize Refraction 1, Madel, Liu, Brunskill, Popvic AAMAS 2014.

Healthcare

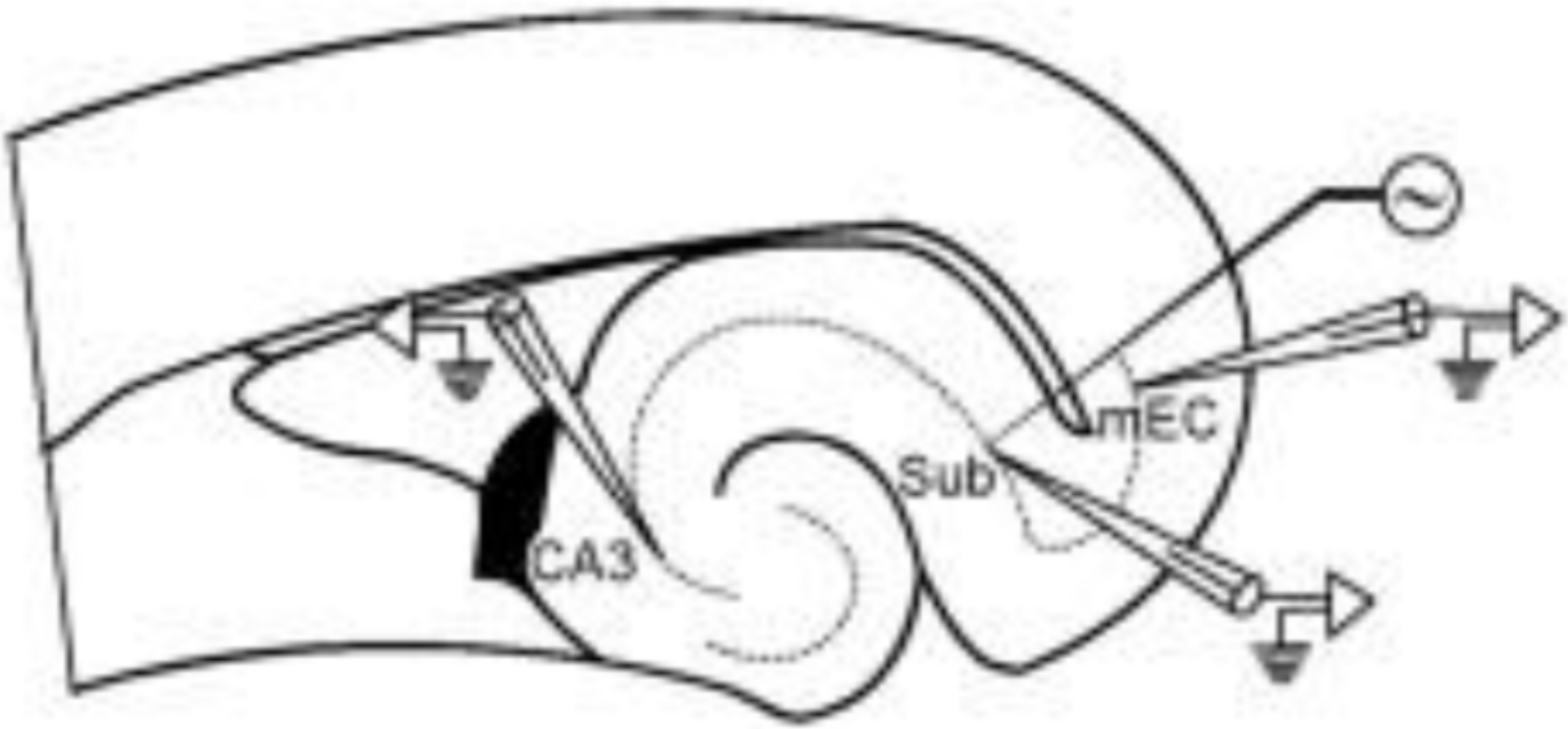


Figure: Adaptive control of epileptiform excitability in an in vitro model of limbic seizures. Panuccio, Quez, Vincent, Avoli, Pineau

NLP, Vision, ...

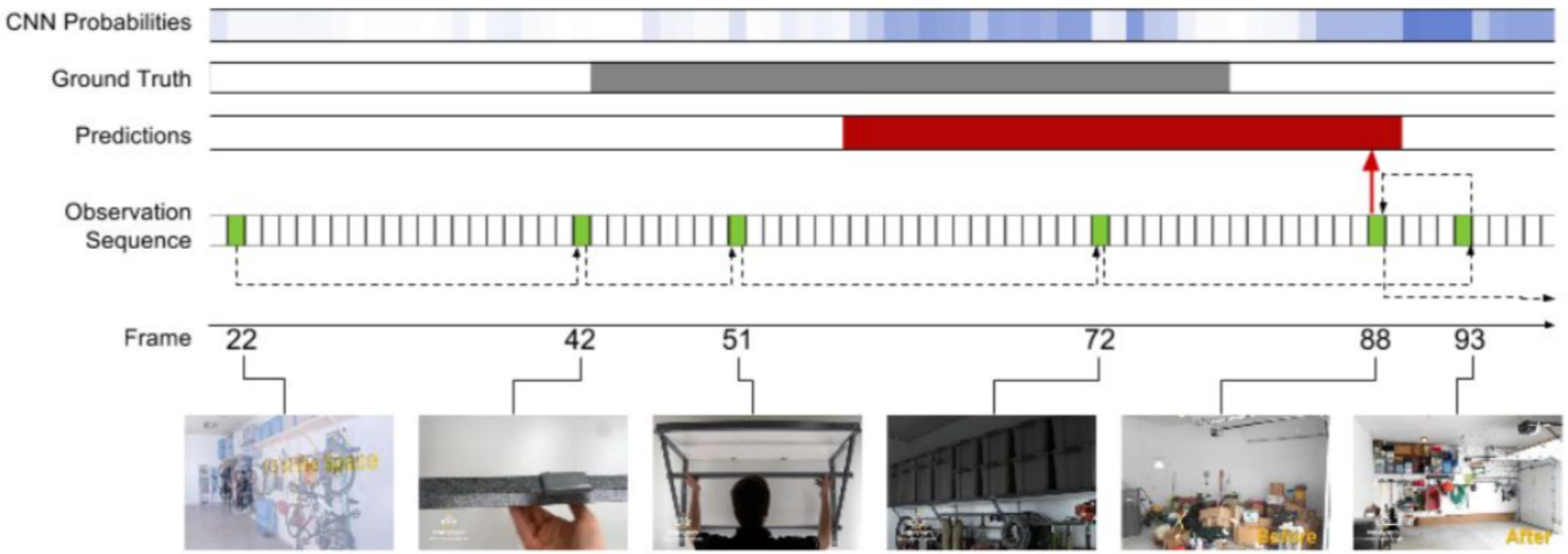


Figure: Yeung, Russakovsky, Mori, Li 2016.

Reinforcement Learning Involves

- Optimization
- Delayed consequences
- Exploration
- Generalization

Optimization

- Goal is to find an optimal way to make decisions
  - Yielding best outcomes
- Or at least a very good strategy

Delayed Consequences

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

Exploration

- Learning about the world by making decisions
  - Agent as scientist
  - 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 blue 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...

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

Generalization

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

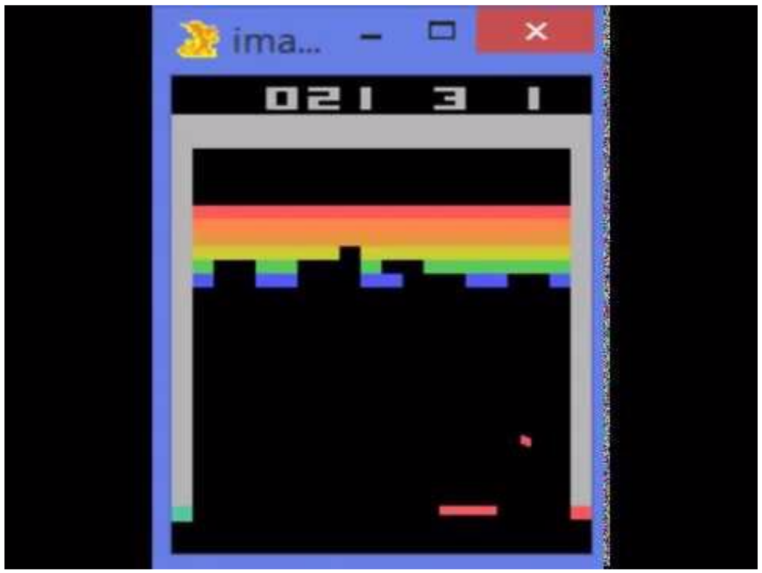


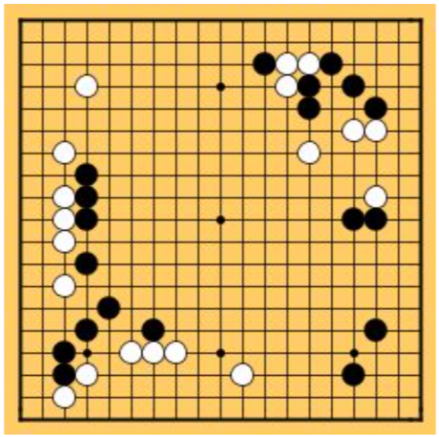
Figure: DeepMind Nature, 2015

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

Reinforcement Learning Involves

- Optimization
- Exploration
- Generalization
- Delayed consequences

AI Planning (vs RL)



- **Optimization**
- **Generalization**
- Exploration
- **Delayed consequences**

- Computes good sequence of decisions
- **But given model of how decisions impact world**

Supervised Machine Learning (vs RL)

- **Optimization**
- **Generalization**
- Exploration
- Delayed consequences

- Learns from experience
- **But provided correct labels**

Unsupervised Machine Learning (vs RL)

- **Optimization**
- **Generalization**
- Exploration
- Delayed consequences

- Learns from experience
- **But no labels from world**

Imitation Learning (vs RL)

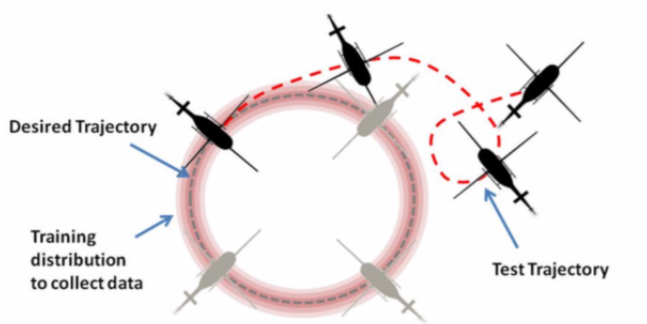
- **Optimization**
- **Generalization**
- Exploration
- **Delayed consequences**

- Learns from experience...**of others**
- **Assumes input demos of good policies**

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!

How Do We Proceed?

- Explore the world
- Use experience to guide future decisions

Other Issues

- Where do rewards come from?
  - And what happens if we get it wrong?
- Robustness / Risk sensitivity
- We are not alone...
  - Multi-agent RL

Today's Plan

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

Basic Logistics

- Instructor: Emma Brunskill
- CA's: Ramtin Keramati (Head CA), Patrick Cho, Anchit Gupta, Bo Liu, Sudarshan Seshadri, Jay Whang, Yongshang Wu, Andrea Zanette
- Time: Monday, Wednesday 11:30-12:50
- Location: Gates B1
- Additional information
  - Course webpage: http://cs234.stanford.edu
  - Schedule, Piazza, lecture slides

Prerequisites

- Python proficiency
- Basic probability and statistics
- Multivariate calculus and linear algebra
- Machine learning or AI (e.g. CS229, CS221)
- Loss functions, deritatives, gradient descent should be familiar
- Have heard of Markov decision processes and RL before in an AI or ML class
  - We will cover the basics, but quickly

End of Class Goals

- Define the key features of reinforcement learning that distinguish it from AI and non-interactive machine learning (as assessed by the exam)
- Given an application problem (e.g. from computer vision, robotics, etc) decide if it should be formulated as a RL problem, if yes be able to define it formally (in terms of the state space, action space, dynamics and reward model), state what algorithm (from class) is best suited to addressing it, and justify your answer. (as assessed by the project and the exam)
- Implement (in code) common RL algorithms including a deep RL algorithm (as assessed by the homeworks)
- Describe (list and define) multiple criteria for analyzing RL algorithms and evaluate algorithms on these metrics: e.g. regret, sample complexity, computational complexity, empirical performance, convergence, etc. (as assessed by homeworks and the exam)
- Describe the exploration vs exploitation challenge and compare and contrast at least two approaches for addressing this challenge (in terms of performance, scalability, complexity of implementation, and theoretical guarantees) (as assessed by an assignment and the exam)

Grading

- Assignment 1 — 10%
- Assignment 2 — 20%
- Assignment 3 — 16%
- Midterm — 25%
- Quiz — 5%
- Final Project — 24%
  - Proposal — 1%
  - Milestone — 2%
  - Poster presentation — 5%
  - Final Report — 16%

If you choose to do the default project/4th assignment, your breakdown will instead be

- Poster presentation — 5%
- Paper/assignment write up — 19%

Communication

- We believe students often learn an enormous amount from each other as well as from us, the course staff.
- We will use Piazza to facilitate discussion and peer learning
- Please use Piazza for all questions related to lectures, homeworks, and projects

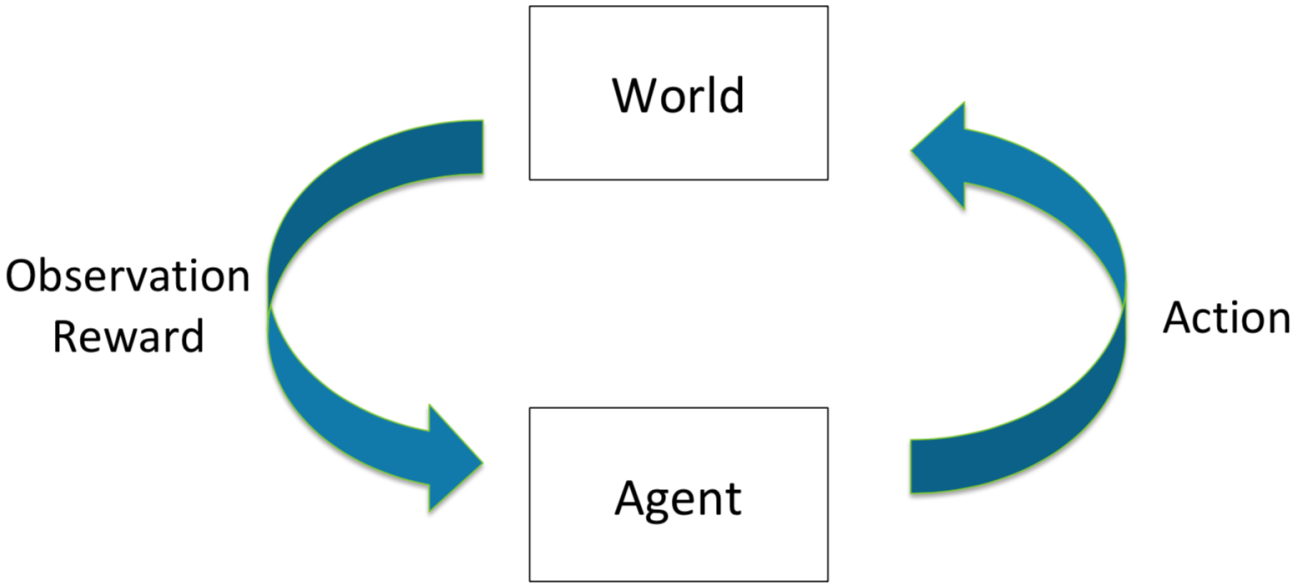
Grading

- Late policy
  - 6 free late days
  - See webpage for details on how may per assignment/project and penalties for using more
- Collaboration: see webpage and reach out to us if you have any questions about what is considered permissible collaboration

Today's Plan

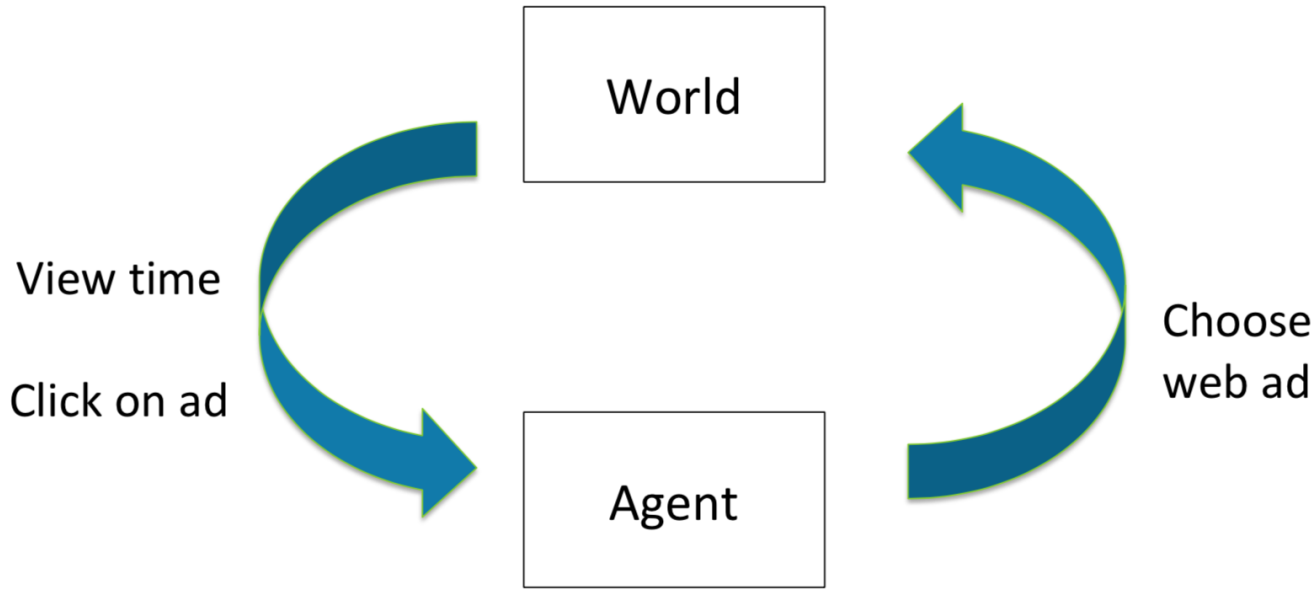
- Overview of reinforcement learning
- Course logistics
- **Introduction to sequential decision making under uncertainty**

Sequential Decision Making



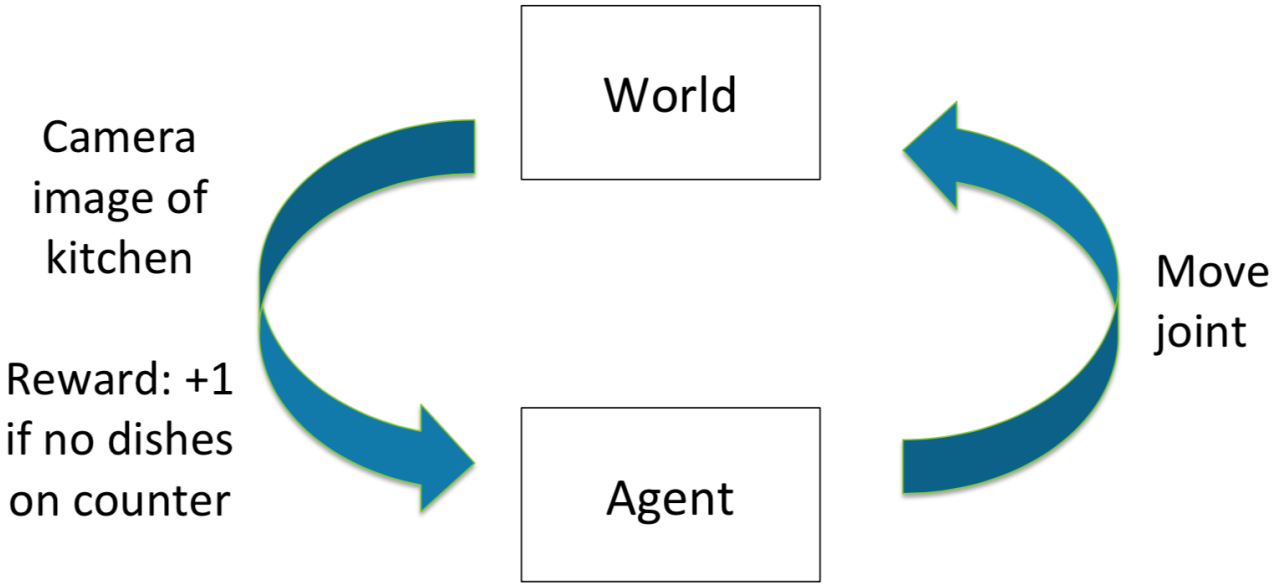
- Goal: Select actions to maximize total expected future reward
- May require balancing immediate & long term rewards
- May require strategic behavior to achieve high rewards

Example: Web Advertising



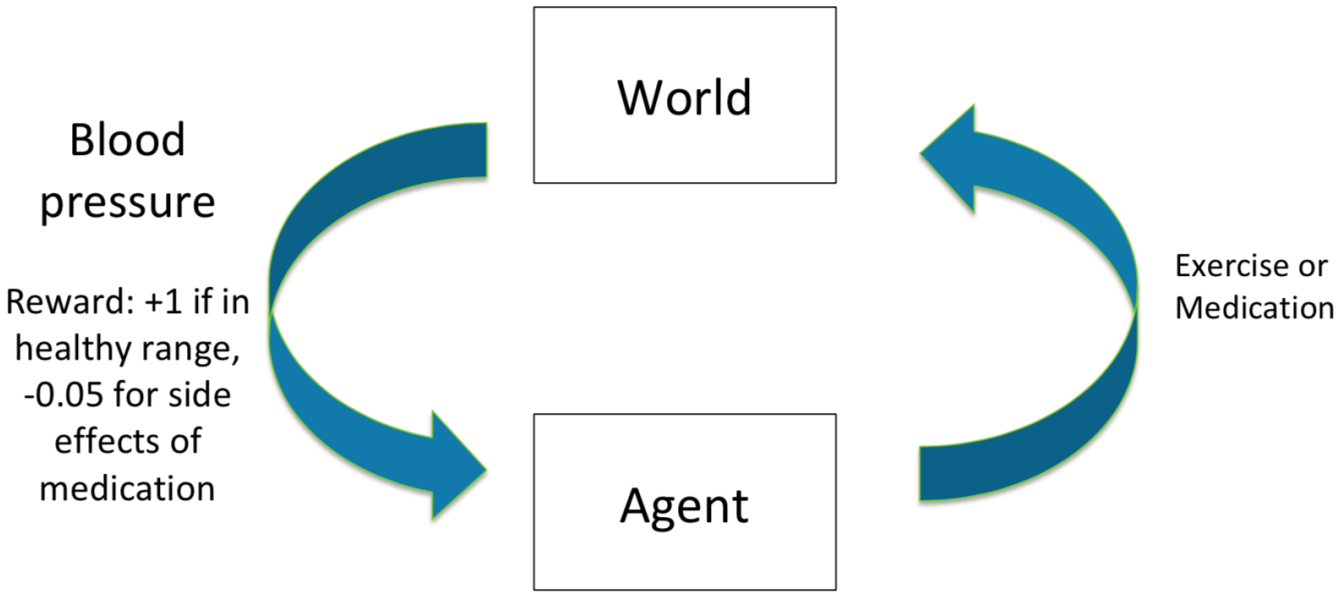
- Goal: Select actions to maximize total expected future reward
- May require balancing immediate & long term rewards
- May require strategic behavior to achieve high rewards

Example: Robot Unloading Dishwasher



- Goal: Select actions to maximize total expected future reward
- May require balancing immediate & long term rewards
- **May require strategic behavior to achieve high rewards**

Example: Blood Pressure Control



- Goal: Select actions to maximize total expected future reward
- **May require balancing immediate & long term rewards**
- **May require strategic behavior to achieve high rewards**

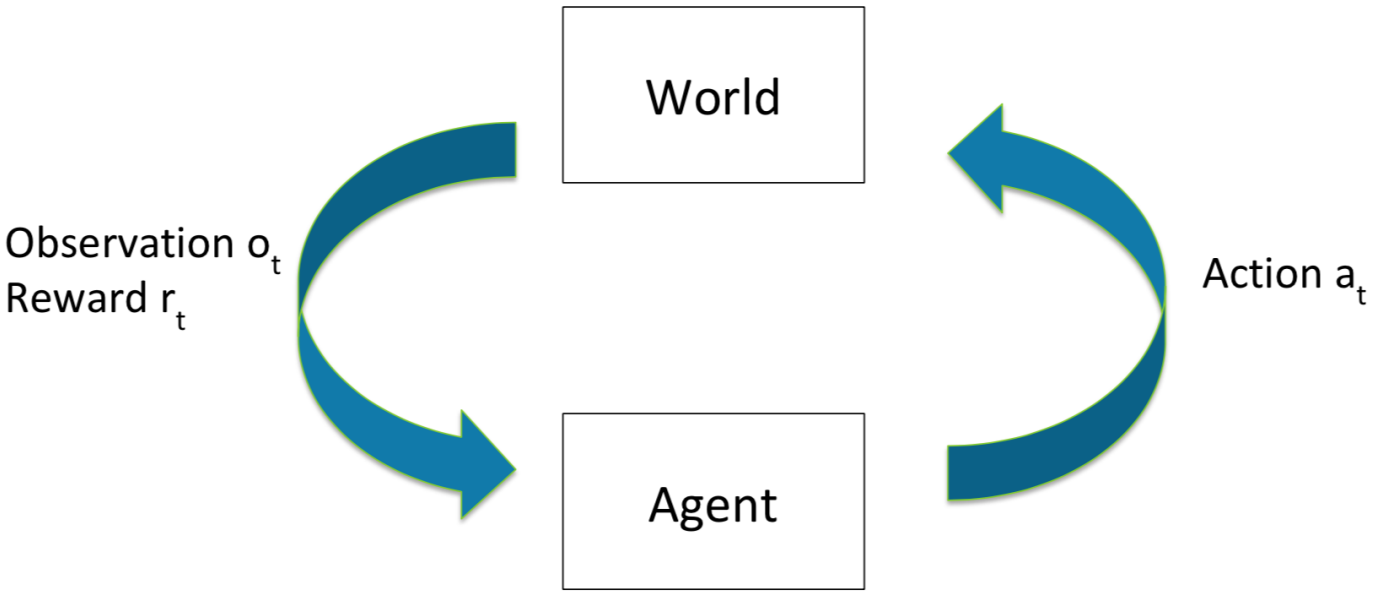
Check Your Understanding: Artificial Tutor

- Student initially does not know addition (easier) nor subtraction (harder)
- Teaching agent can provide activities about addition or subtraction
- Agent gets rewarded for student performance: $+1$ if student gets problem right, -1 if get problem wrong

Check Your Understanding: Artificial Tutor

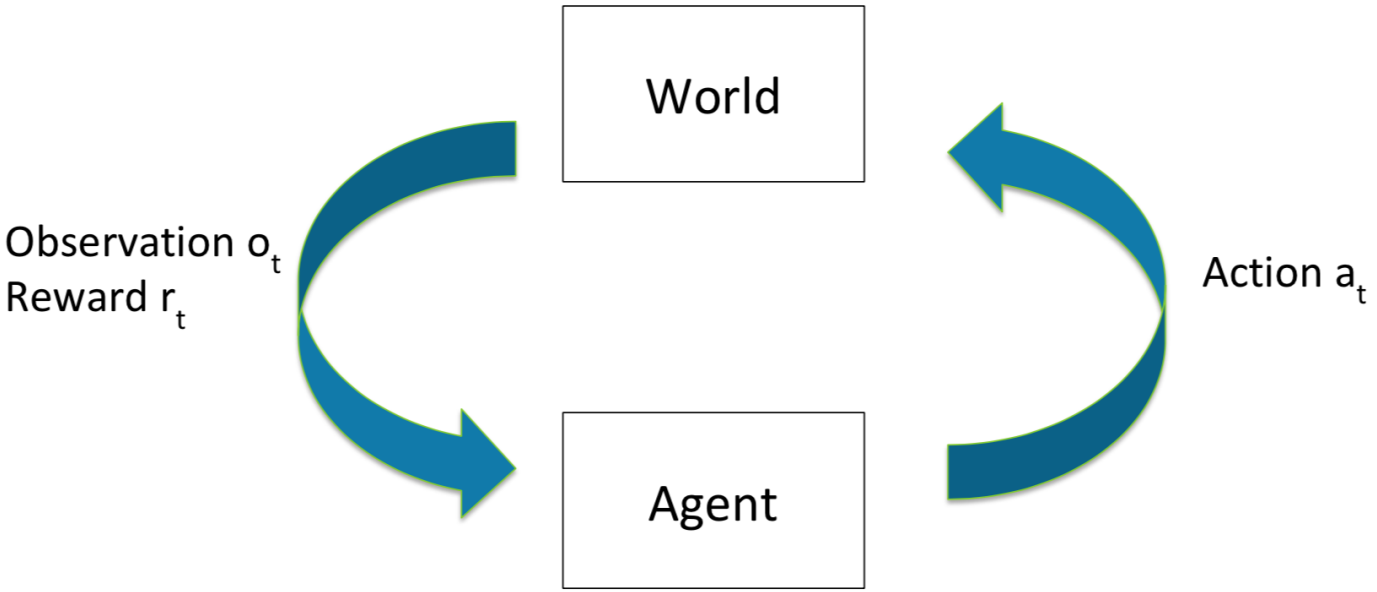
- Student initially does not know addition (easier) nor subtraction (harder)
- Teaching agent can provide activities about addition or subtraction
- Agent gets rewarded for student performance: $+1$ if student gets problem right, -1 if get problem wrong
- Which items will agent learn to give to max expected reward? Is this the best way to optimize for learning? If not, what other reward might one give to encourage learning?

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



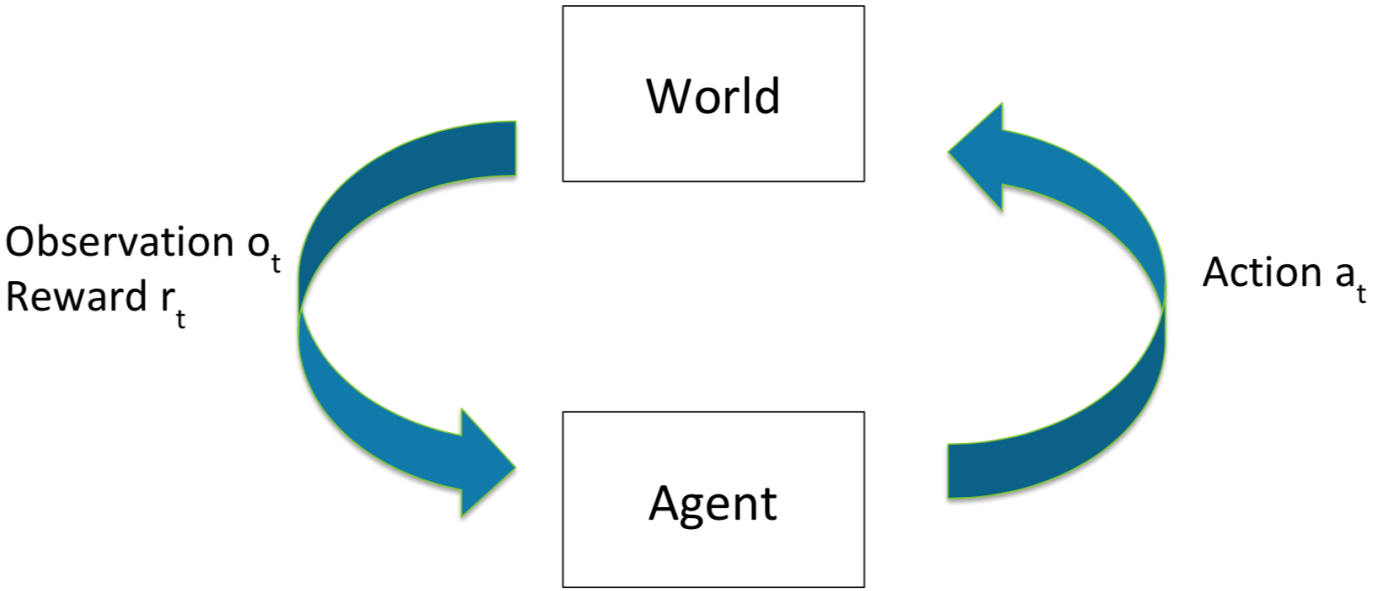
- Each time step t:
  - Agent takes an action a_t
  - World updates given action a_t, emits observation o_t and reward r_t
  - Agent receives observation o_t and reward r_t

History: Sequence of Past Observations, Actions & Rewards



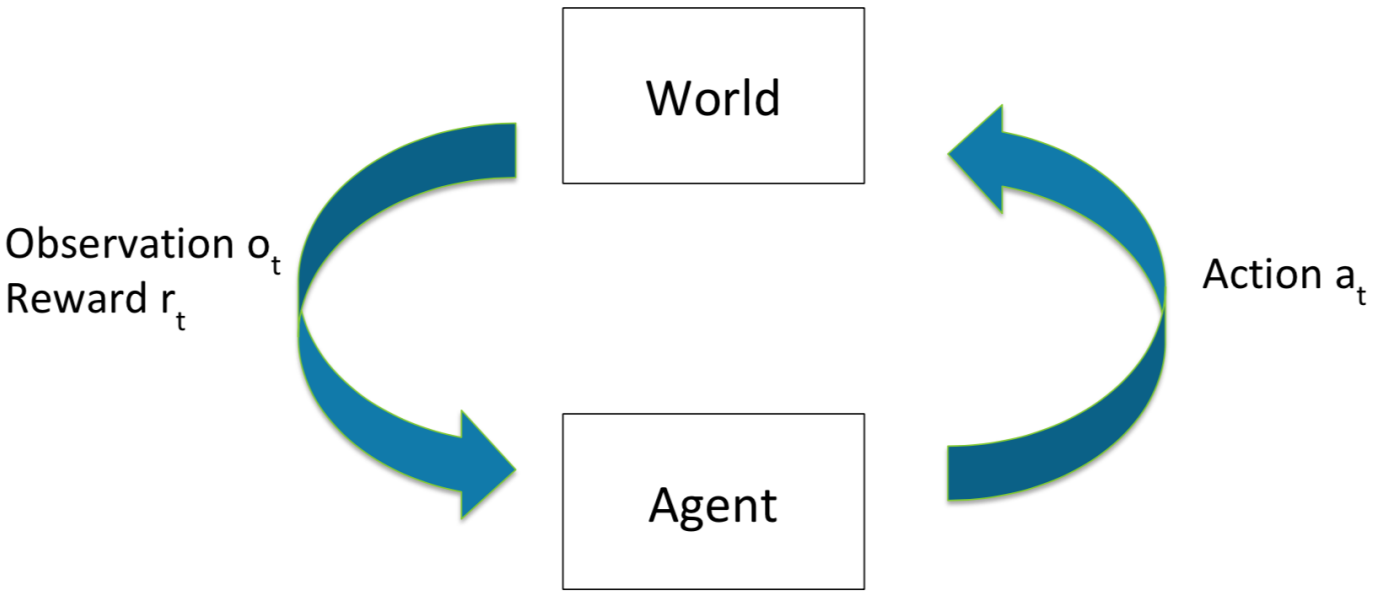
- History $h_t = (a_1, o_1, r_1, \ldots, a_t, o_t, r_t)$
- Agent chooses action based on history
- State is information assumed to determine what happens next
  - Function of history: $s_t = (h_t)$

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)

Markov Assumption

- Information state: sufficient statistic of history
- State s_t is Markov if and only if:

$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t)$$

- Future is independent of past given present

Markov Assumption for Prior Examples

- Information state: sufficient statistic of history
- State s_t is Markov if and only if:

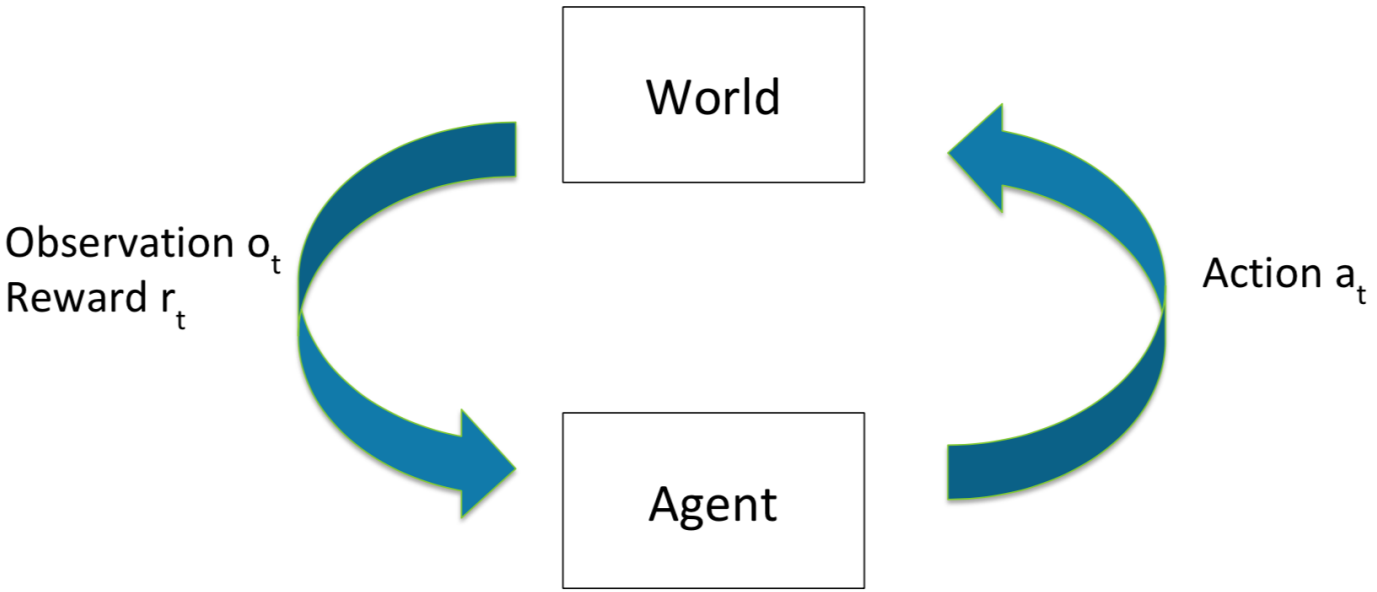
$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|h_t, a_t)$$

- Future is independent of past given present
- Hypertension control: let state be current blood pressure, and action be whether to take medication or not. Is this system Markov?
- Website shopping: state is current product viewed by customer, and action is what other product to recommend. Is this system Markov?

Why is Markov Assumption Popular?

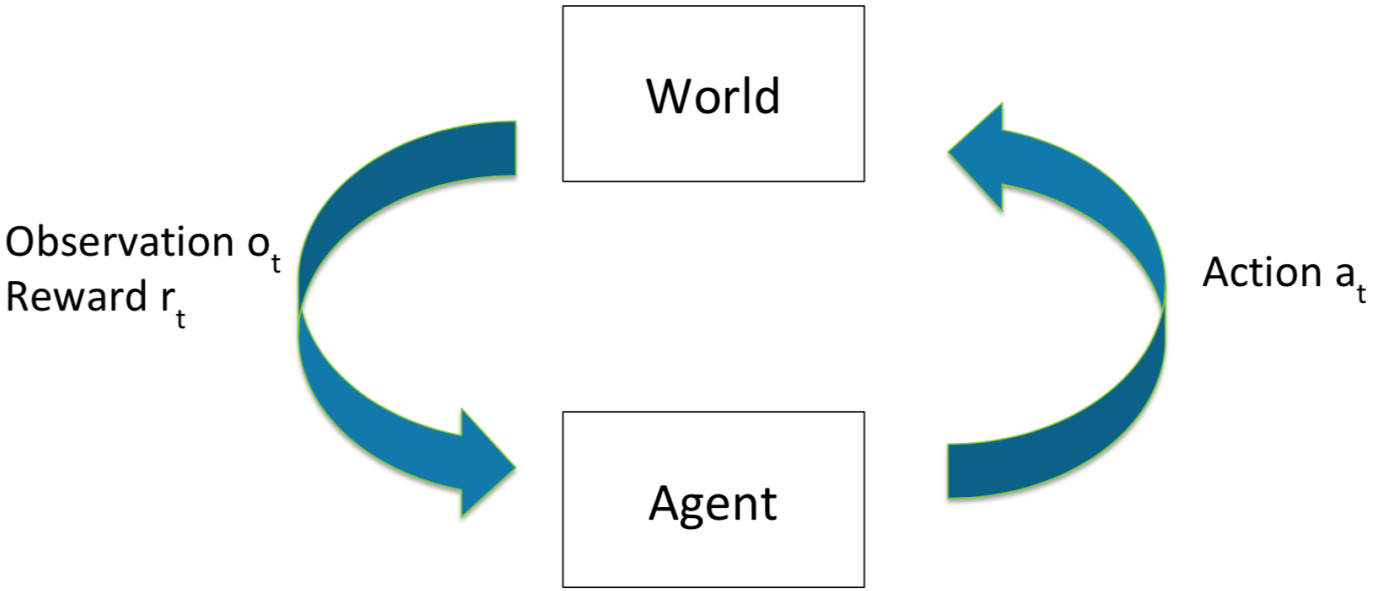
- Can always be satisfied
  - Setting state as history always Markov: $s_t = h_t$
- In practice often assume most recent observation is sufficient statistic of history: $s_t = o_t$
- State representation has big implications for:
  - Computational complexity
  - Data required
  - Resulting performance

Full Observability / Markov Decision Process (MDP)



- Environment and world state $s_t = o_t$

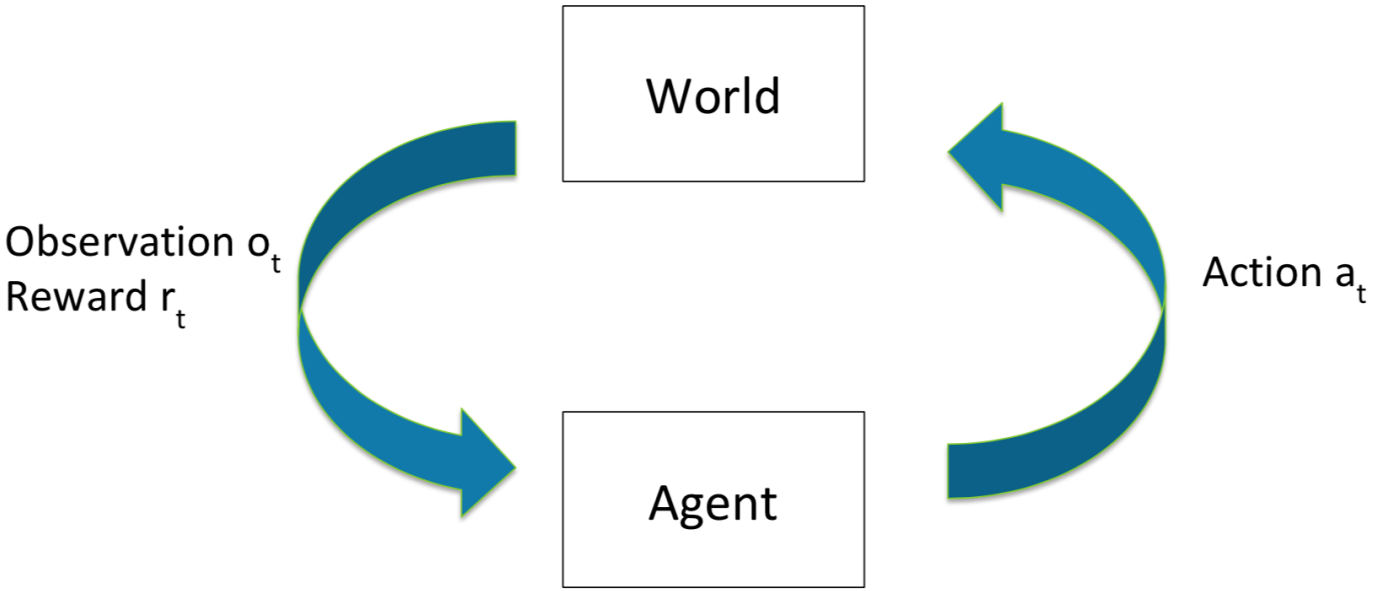
Partial Observability / Partially Observable Markov Decision Process (POMDP)



- Agent state is not the same as the world state
- Agent constructs its own state, e.g.
  - Use history $s_t = h_t$, or beliefs of world state, or RNN, ...

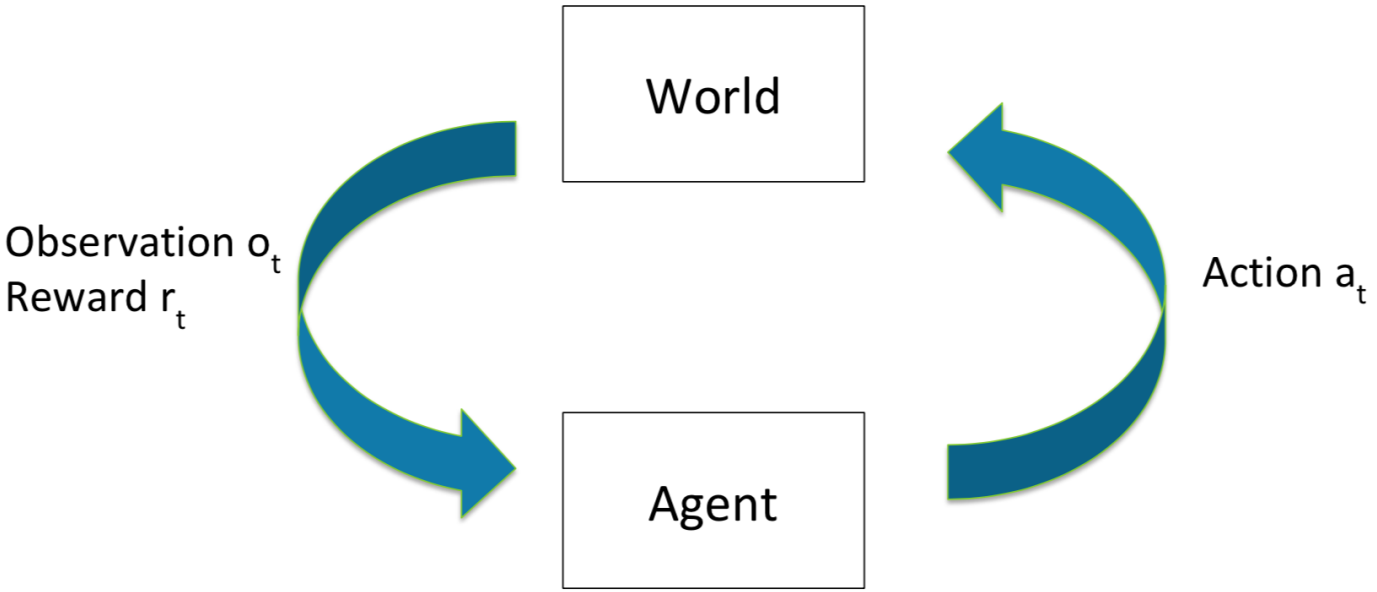
Partial Observability Examples

- Poker player (only sees own cards)
- Healthcare (don't see all physiological processes)



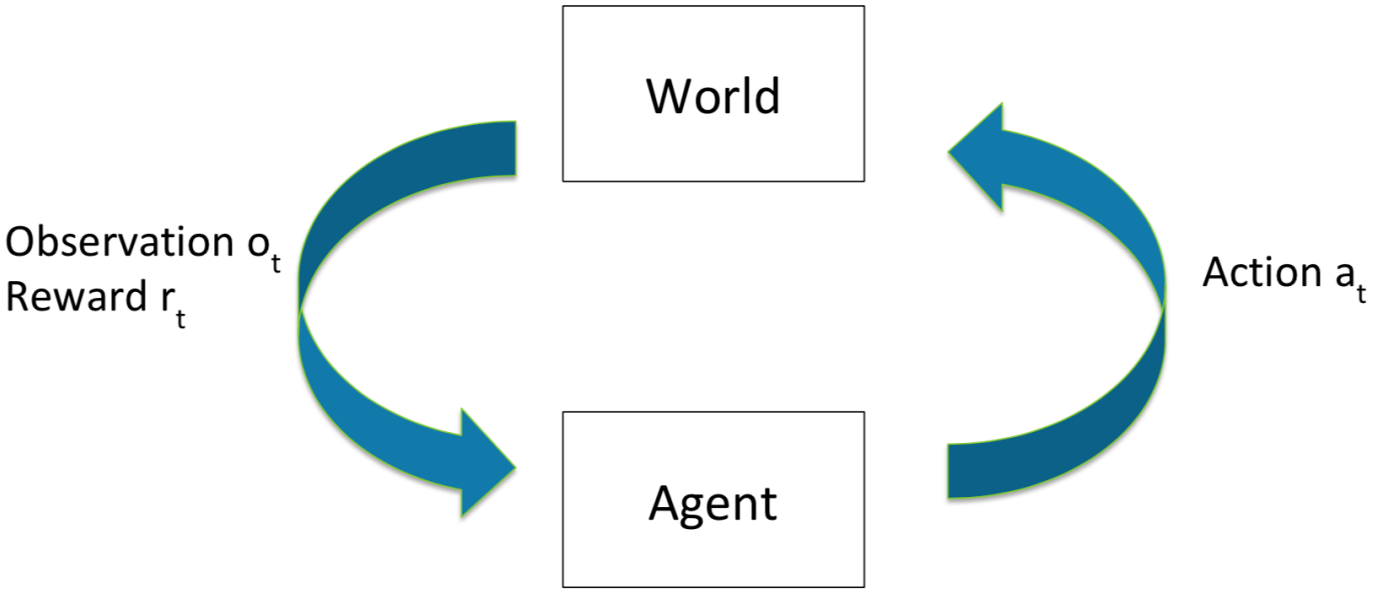
- Agent state is not the same as the world state
- Agent constructs its own state, e.g.
  - Use history $s_t = h_t$, or beliefs of world state, or RNN, ...

Types of Sequential Decision Processes: Bandits



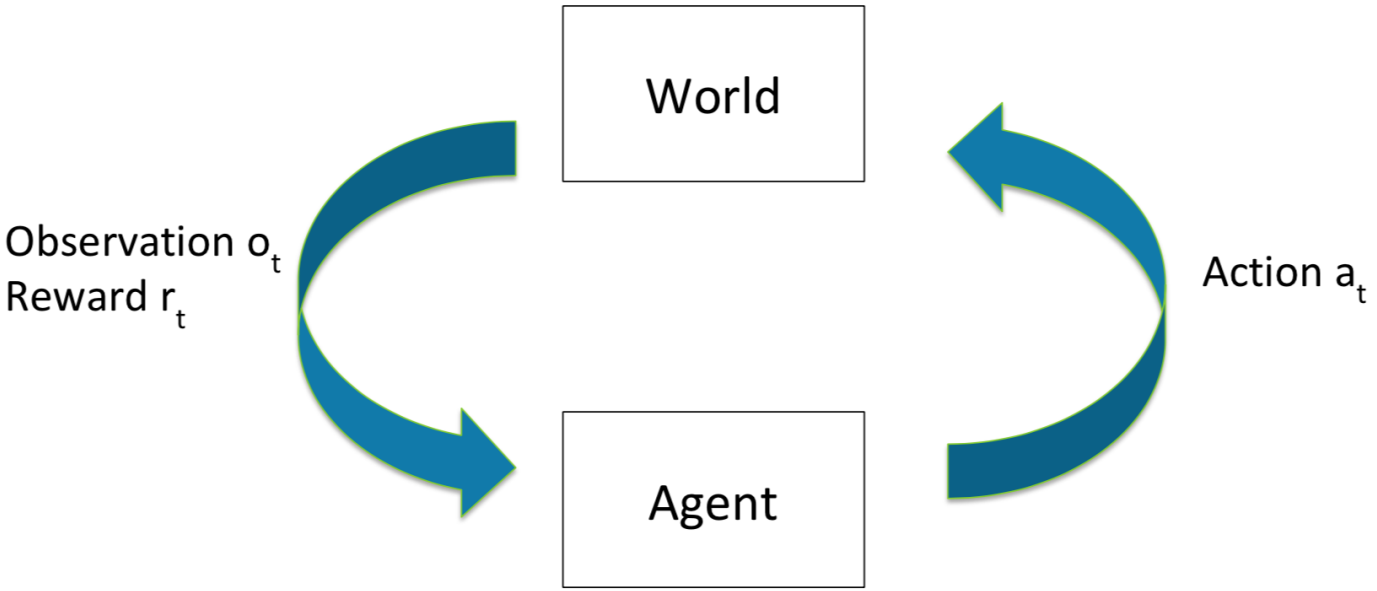
- Bandits: actions have no influence on next observations
- No delayed rewards

Types of Sequential Decision Processes: MDPs and POMDPs



- Actions influence future observations
- Credit assignment and strategic actions may be needed

Types of Sequential Decision Processes: How the World Changes



- **Deterministic**: Given history & action, single observation & reward
  - Common assumption in robotics and controls
- **Stochastic**: Given history & action, many potential observations & rewards
  - Common assumption for customers, patients, hard to model domains

Example: Mars Rover as a Particular Markov Decision Process

s_1	s_2	s_3	s_4	s_5	s_6	s_7
						

Figure: Mars rover image: NASA/JPL-Caltech

- States: Location of rover $(s_1, \ldots, s_7)$
- Actions: Left or Right
- Rewards:
  - $+1$ in state s_1
  - $+10$ in state s_7
  - 0 in all other states

RL Algorithm Components

- Often include one or more of
  - **Model:** Representation of how the world changes in response to agent's action
  - **Policy**: function mapping agent's states to action
  - **Value function**: Future rewards from being in a state and/or action when following a particular policy

Model

- Agent's representation of how the world changes in response to 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

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

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:
  - $0.5 = P(s_1|s_1, \text{right}) = P(s_2|s_1, \text{right})$
  - $0.5 = P(s_2|s_2, \text{right}) = P(s_3|s_2, \text{right}) \cdots$
- Model may be wrong

Policy

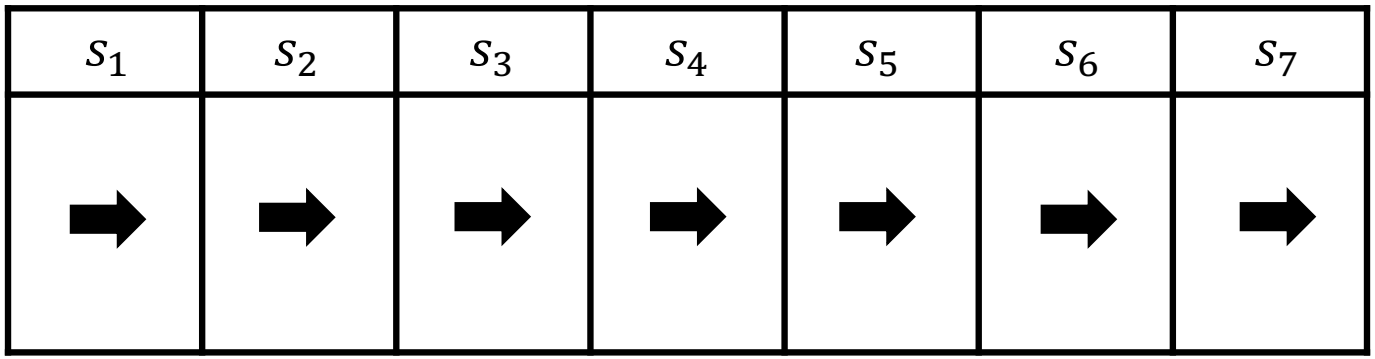
- Policy π determines how the agent chooses actions
- $\pi : S \rightarrow A$, mapping from states to actions
- Deterministic policy:

$$\pi(s) = a$$

- Stochastic policy:

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

Example: Mars Rover Policy



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

Value

- 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, $\gamma = 0$
- $\pi(s_1) = \pi(s_2) = \cdots = \pi(s_7) =$ right
- 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

RL Agents

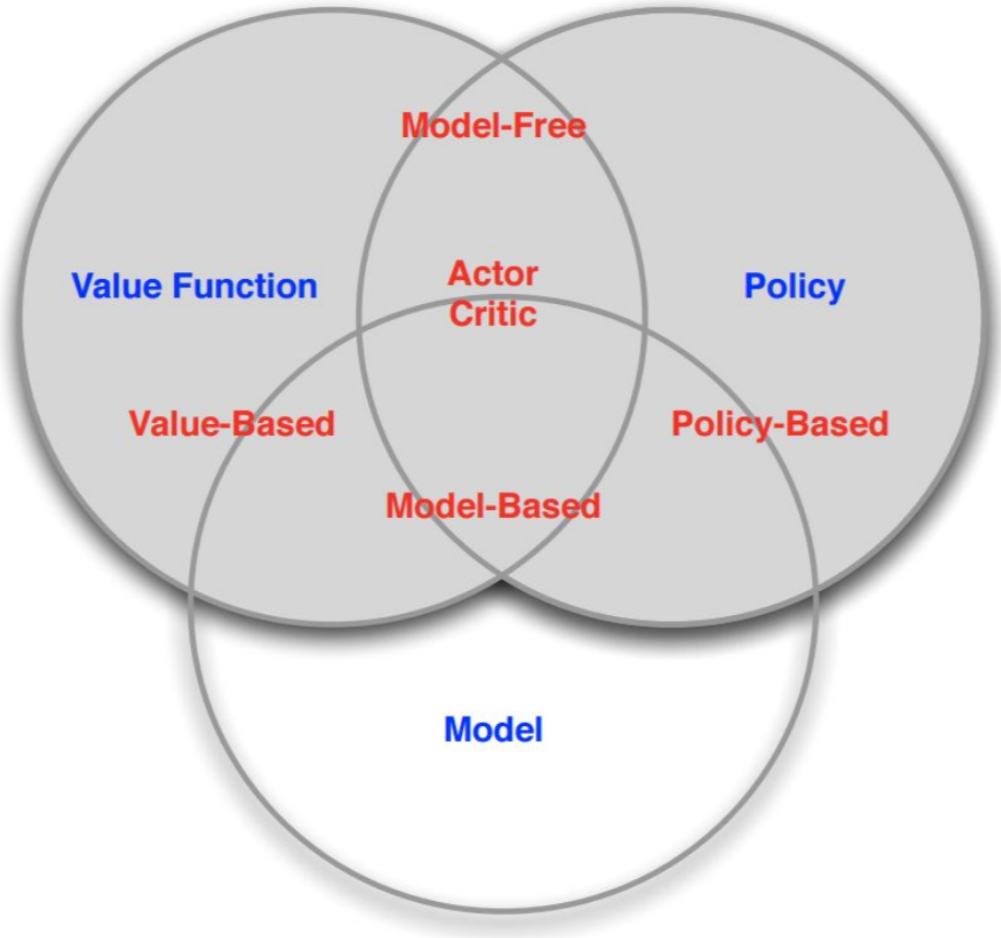


Figure: Figure from David Silver's RL course

Key Challenges in Learning to Make Sequences of Good Decisions

- Planning (Agent's internal computation)
  - Given model of how the world works
    - Dynamics and reward model
  - Algorithm computes how to act in order to maximize expected reward
    - With no interaction with real environment

Key Challenges in Learning to Make Sequences of Good Decisions

- Planning (Agent's internal computation)
  - Given model of how the world works
    - Dynamics and reward model
  - Algorithm computes how to act in order to maximize expected reward
    - With no interaction with real environment
- Reinforcement learning
  - Agent doesn't know how world works
  - Interacts with world to implicitly/explicitly learn how world works
  - Agent improves policy (may involve planning)

Planning Example

- Solitaire: single player card game
- Know all rules of game / perfect model
- If take action a from state s
  - Can compute probability distribution over next state
  - Can compute potential score
- Can plan ahead to decide on optimal action
  - E.g. dynamic programming, tree search, ...

Reinforcement Learning Example

- Solitaire with no rule book
- Learn directly by taking actions and seeing what happens
- Try to find a good policy over time (that yields high reward)

Exploration and Exploitation

- Agent only experiences what happens for the actions it tries
  - Mars rover trying to drive left learns the reward and next state for trying to drive left, but not for trying to drive right
  - Obvious! But leads to a dilemma

Exploration and Exploitation

- Agent only experiences what happens for the actions it tries
- How should an RL agent balance its actions?
  - Exploration: trying new things that might enable the agent to make better decisions in the future
  - Exploitation: choosing actions that are expected to yield good reward given past experience
- Often there may be an exploration-exploitation tradeoff
  - May have to sacrifice reward in order to explore & learn about potentially better policy

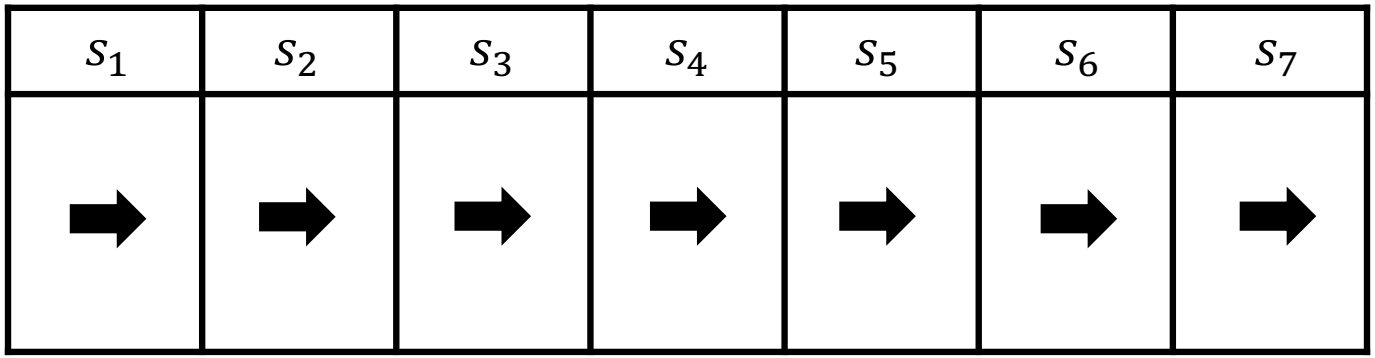
Exploration and Exploitation Examples

- Movies
  - Exploitation: Watch a favorite movie you've seen before
  - Exploration: Watch a new movie
- Advertising
  - Exploitation: Show most effective ad so far
  - Exploration: Show a different ad
- Driving
  - Exploitation: Try fastest route given prior experience
  - Exploration: Try a different route

Evaluation and Control

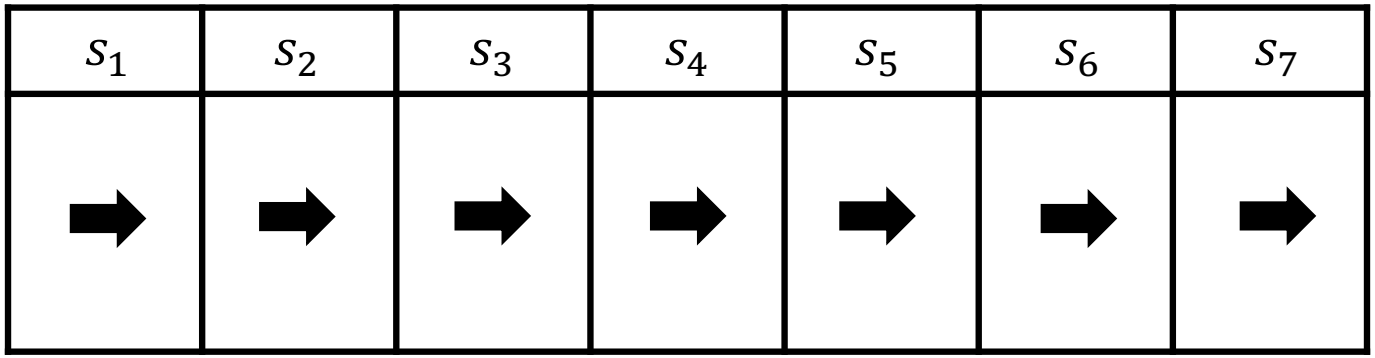
- Evaluation
  - Estimate/predict the expected rewards from following a given policy
- Control
  - Optimization: find the best policy

Example: Mars Rover Policy Evaluation



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

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

See website for more details

Summary

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

Types of RL Agents: What the Agent (Algorithm) Learns

- Model-based
  - Explicit: Model
  - Implicit: Policy and/or value function (can use model to plan: compute a policy and/or value function)
- Valued-based
  - Explicit: Value function
  - Implicit: Policy (can derive a policy from value function)
- Policy-based
  - Explicit: Policy
  - No value function
- Actor-Critic
  - Explicit: Policy
  - Explicit: Value function
- Algorithms can also have an explicit model, value and policy.