Reinforcement Learning

Lecture 2: Foundations of Reinforcement Learning

Chengchun Shi

Lecture Outline

1. General Reinforcement Learning (RL) Problems
2. Markov Decision Processes (MDPs)
3. Time-Varying MDPs and Partially Observable MDPs
4. Policy, Return and Value
5. The Existence of the Optimal Policy

Lecture Outline (Cont'd)

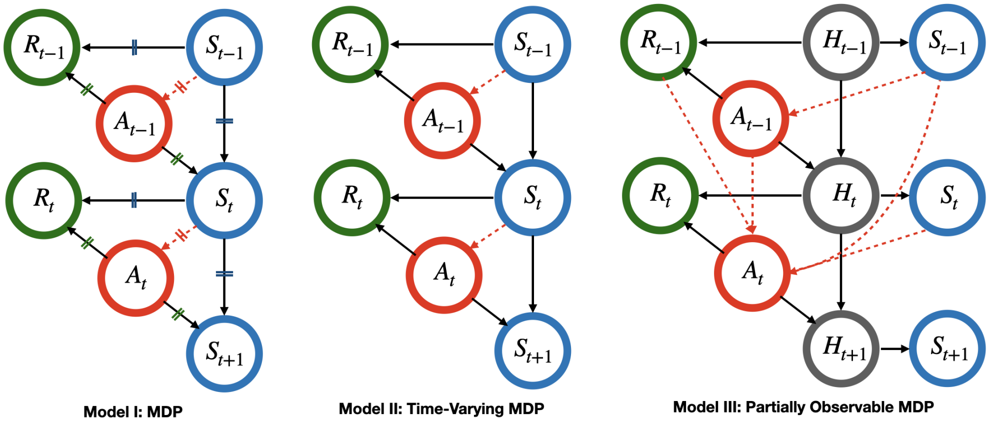
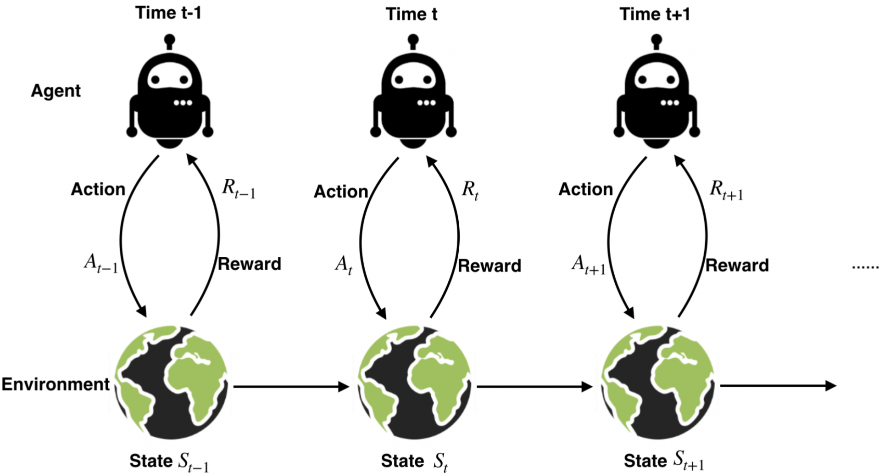


Figure: Causal diagrams for MDPs, TMDPs and POMDPs. Solid lines represent the causal relationships. Dashed lines indicate the information needed to implement the optimal policy. $\{H_t\}_t$ denotes latent variables. The parallel sign $\|$ indicates that the conditional probability function given parent nodes is equal.

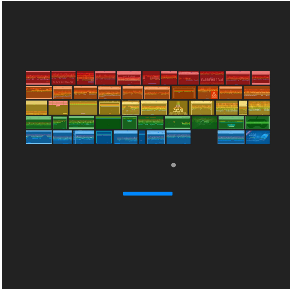
1. **General Reinforcement Learning (RL) Problems**
2. Markov Decision Processes (MDPs)
3. Time-Varying MDPs and Partially Observable MDPs
4. Policy, Return and Value
5. The Existence of the Optimal Policy

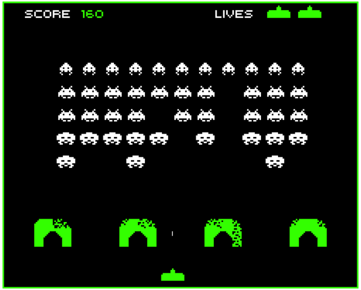
Sequential Decision Making



Objective: find an optimal policy that maximizes the cumulative reward

Atari Games

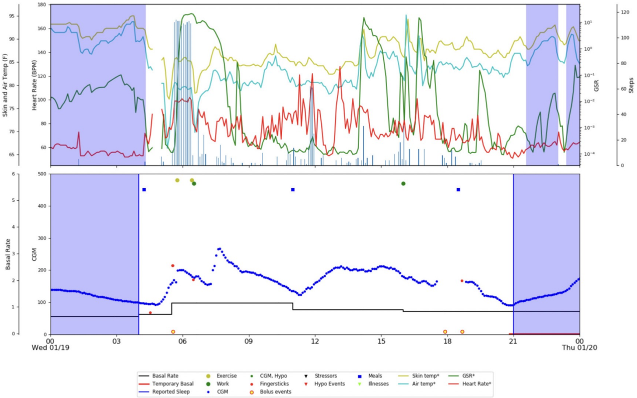




- S_t: **images**
- A_t: Legal **game actions**
- R_t: **Scores** & **lives**

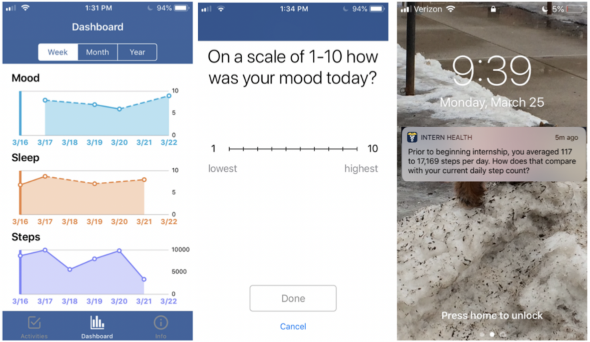
Diabetes

- Management of **Type-I diabetes**
- **Subject**: Patients with diabetes.
- **Objective**: Develop treatment policy to determine whether patients need to inject insulin at each time to improve their health
- S_t: Patient's **glucose levels**, **food intake**, **exercise intensity**
- A_t: **Insulin doses** injected
- R_t: **Index of Glycemic Control** (function of patient's glucose level)



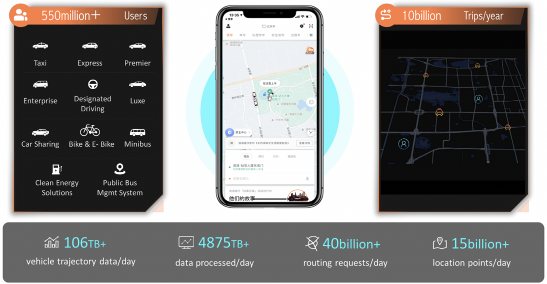
Intern Health Study

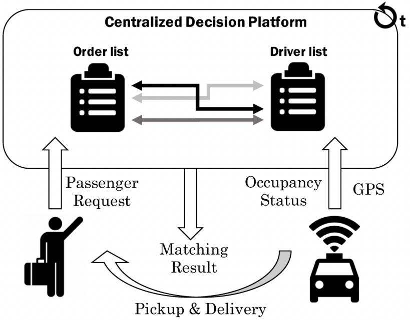
- **Physical & mental health** management
- **Subject**: First-year medical interns
- **Objective**: Develop treatment policy to determine whether to send certain text messages to interns to improve their health
- S_t: Interns' **mood scores**, **sleep hours** and **step counts**
- A_t: Send **text notifications** or not
- R_t: **Mood scores** or **step counts**



(i) App Dashboard (ii) Mood EMA (iii) Notifications

Ridesharing: Order-Dispatching





- S_t: **Supply** (drivers: availability, location) and **demand** (call orders: origin, destination)
- A_t: **Order-dispatching**: match a driver with an order
- R_t: **Answer rate/Completion rate/Drivers' income**

RL v.s. Supervised Learning

Supervised learning consider

- **Prediction** problems
- examples provided by a **supervisor**
- **Independent** data
- Applications:
  - Voice recognition
  - Image classification

RL is concerned with

- Sequential decision making
- No supervisor, only a **reward** signal
- **Time-dependent** data
- Applications:
  - Games
  - Robotics

Introduction to MDPs

- **Markov decision processes** formally describe an environment for reinforcement learning where the environment is **fully-observable**
- The current **state-action** pair completely characterizes the process (**Markov** property)
- Most RL problems can be formalised as MDPs, e.g.,
  - **Bandits** are MDPs with independent transitions
  - Many **non-Markov decision processes** (e.g., time-varying MDPs) can be converted into MDPs by
    - including time in the state
    - concatenating measurements over multiple times

(Time-Homogeneous) Markov Chains

Definition

$\{S_t\}_t$ forms a time-homogeneous **Markov chain** if and only if

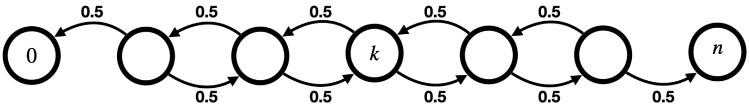
- $\Pr(S_{t+1}|S_t) = \Pr(S_{t+1}|S_1, \cdots, S_t)$ (Markov property)
- $\Pr(S_{t+1}|S_t = s) = \Pr(S_t|S_{t-1} = s)$ (time-homogeneity)

More on the **Markov property**:

- The future is independent of the past given the present
- The current **state** captures all relevant information from the history
- Once the state is known, the history may be thrown away
- The state can be viewed as a **sufficient statistic** of the history

Example: Random Walk on a Line

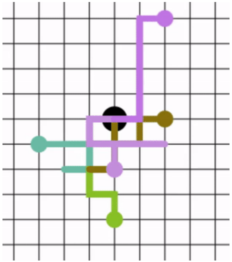
- You go into a casino with £k, and at each time step, you bet £1 on a fair game
- For each game, you win or lose with probability 0.5. The outcomes are **independent** across different games.
- You leave when you are broke or have £n



- A very popular model in finance to model stock price

Example: Two-Dimensional Random Walk

- The drunkard starts at a "home" vertex **0**
- Then **independently** chooses at **random** a neighbouring vertex (left, right, forward, backward) to walk next at each time



Example: High-Dimensional Random Walk

- A drunk man will find his way home, but a drunk bird may get lost forever
- In a two-dimensional space, the drunkard will return home **infinitely many** times

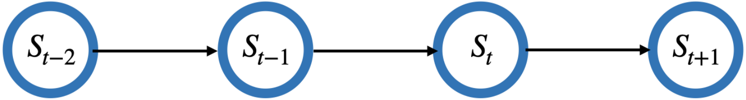
$$\sum_{t \geq 0} \mathbb{I}(S_t = S_0) = \infty$$

- In a three-dimensional space, the bird can only return home some **finite** number of times. After its last return home the bird then flies off never to return again

$$\sum_{t \geq 0} \mathbb{I}(S_t = S_0) < \infty$$

Causal Diagram

- **Markov chain**



- $\boldsymbol{X} \rightarrow \boldsymbol{Y}$ if and only if $\boldsymbol{X}$ **directly** impacts $\boldsymbol{Y}$
- $\boldsymbol{X}$ and $\boldsymbol{Y}$ are **independent** if and only if (iff) $\boldsymbol{X}$ and $\boldsymbol{Y}$ are d-separated i.e., there does not exist a connecting path between $\boldsymbol{X}$ and $\boldsymbol{Y}$
- $\boldsymbol{X}$ and $\boldsymbol{Y}$ are **conditionally independent** given $\boldsymbol{Z}$ iff $\boldsymbol{X}$ and $\boldsymbol{Y}$ are d-separated by $\boldsymbol{Z}$. In our examples, it requires $\boldsymbol{Z}$ to block every path between $\boldsymbol{X}$ and $\boldsymbol{Y}$.

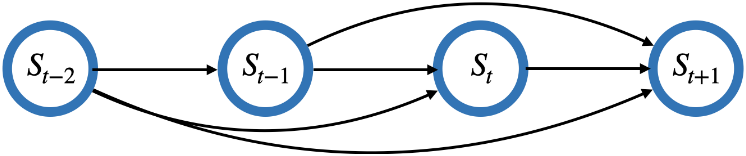
Causal Diagram

- **Markov chain**

- $X \to Y$ if and only if X **directly** impacts Y
- X and Y are **independent** if and only if (iff) X and Y are d-separated i.e., there does not exist a connecting path between X and Y
- X and Y are **conditionally independent** given Z iff X and Y are d-separated by Z. In our examples, it requires Z to block every path between X and Y.

Causal Diagram (Cont'd)

Without the Markov property



Markov Decision Processes

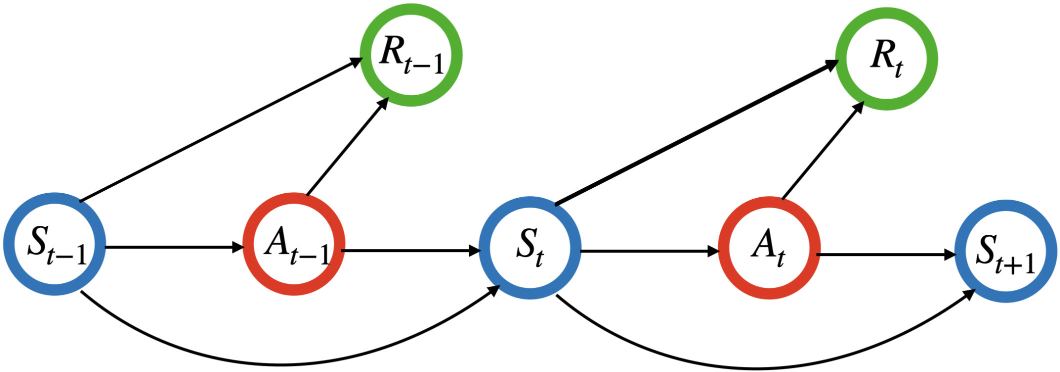
Definition

$\{S_t, A_t, R_t\}_t$ forms a Markov decision process if and only if

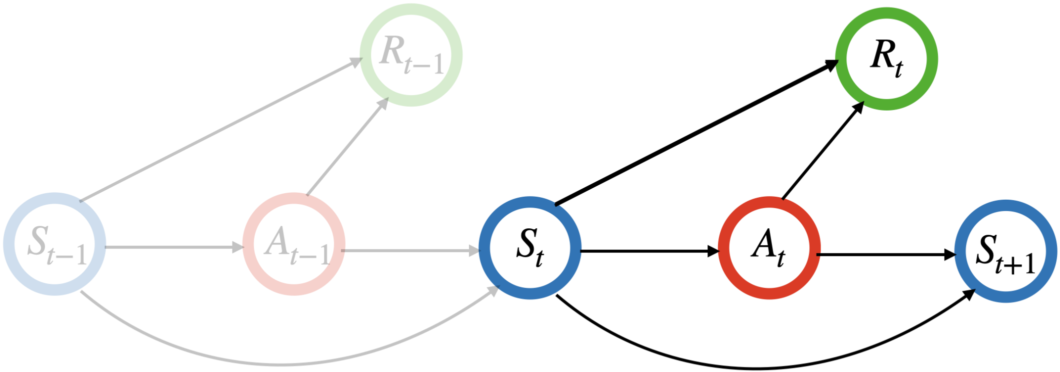
- $\Pr(S_{t+1}, R_t | A_t, S_t) = \Pr(S_{t+1}, R_t | A_t, S_t, R_{t-1}, A_{t-1}, S_{t-1}, \cdots)$ (Markovianity)
- $\Pr(S_{t+1}, R_t | A_t = a, S_t = s) = \Pr(S_t, R_{t-1} | A_{t-1} = a, S_{t-1} = s)$ (time-homogeneity)

- The current **state-action** pair captures all relevant information from the history
- When A_t depends the history only through S_t, $\{S_t, A_t, R_t\}_t$ forms a Markov chain.

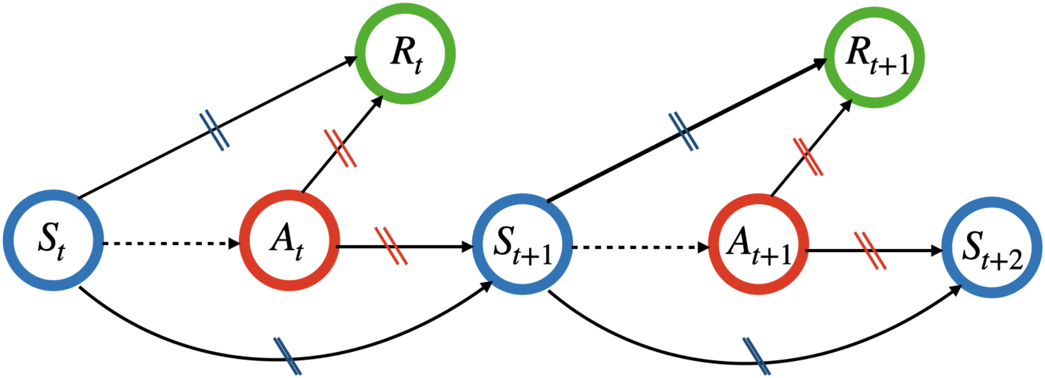
Markov Assumption



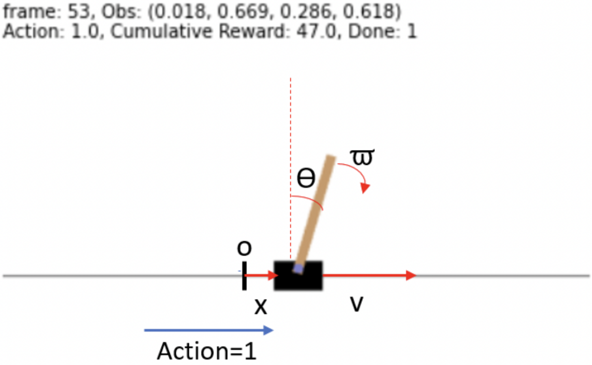
Markov Assumption



Stationarity Assumption



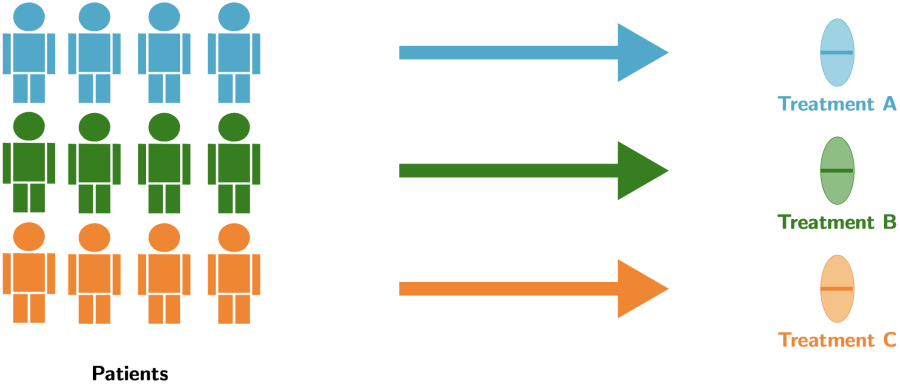
OpenAI Gym Example: CartPole



- S_t: x (**Position**); v (**velocity**); θ (**Angle**); ϖ (**Angular velocity**)
- A_t: Pushing to the **right** or **left**
- R_t: Binary, depending on whether $|\theta| > 15\,\text{deg}$ or not

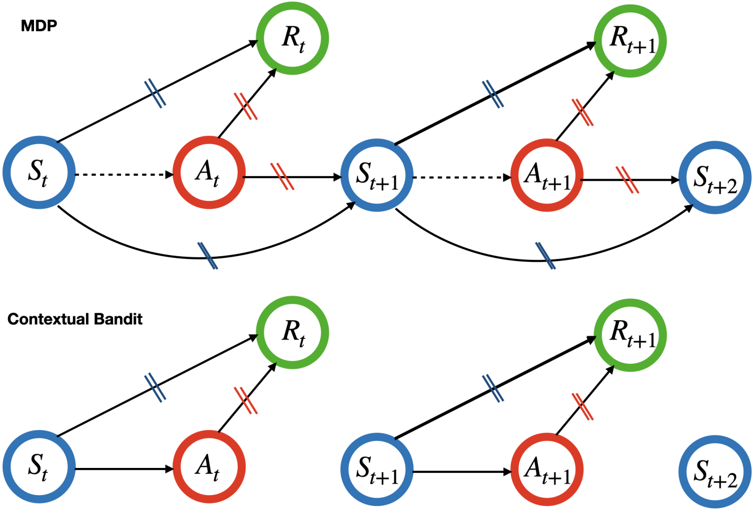
- R_t depends on the history only through θ_t
- (S_t, A_t) captures all relevant information (position, velocity, acceleration)
- The dependencies are **homogeneous** over time (according to laws of physics)
- Most OpenAI Gym Examples satisfy the MDP model assumption

Bandits Example: Precision Medicine

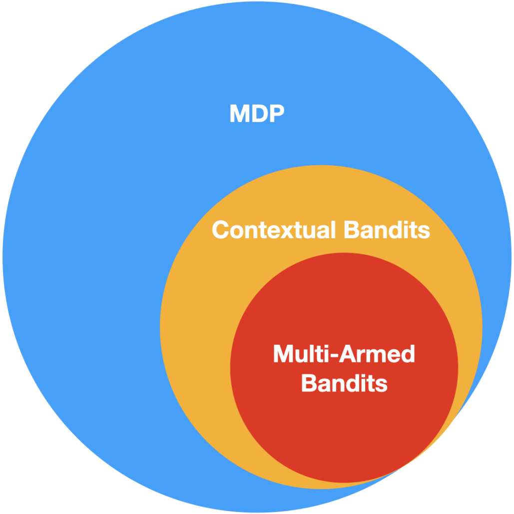


- Patients' **states** (baseline characteristics) are independent
- A patient's **reward** (outcome) depends only on their own state-treatment pair
- **State-treatment-reward** triples are identically distributed

MDP vs Contextual Bandits



MDP v.s. Contextual Bandits (Cont'd)



Time-Varying MDPs

- The **time-homogeneity** assumption is likely to be violated in real applications (e.g., mobile health, ridesharing)
- **Nonstationarity** *is the case most commonly encountered in reinforcement learning* [Sutton and Barto, 2018]

Definition

$\{S_t, A_t, R_t\}_t$ forms a time-varying Markov decision process iff

$$\Pr(S_{t+1}, R_t | A_t, S_t) = \Pr(S_{t+1}, R_t | A_t, S_t, R_{t-1}, A_{t-1}, S_{t-1}, \cdots) \qquad \text{(Markovianity)}$$

Causal Diagram: TMDP

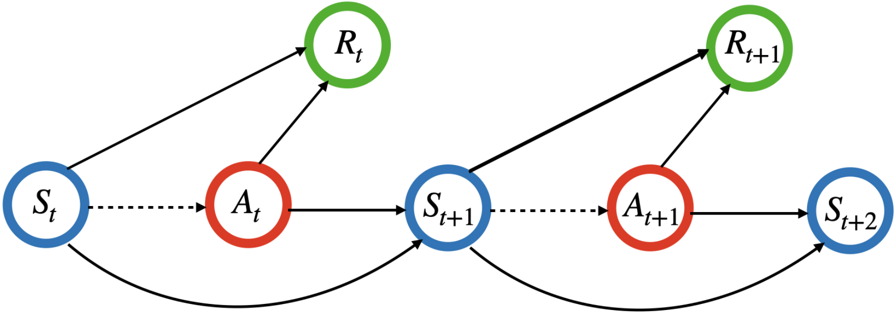
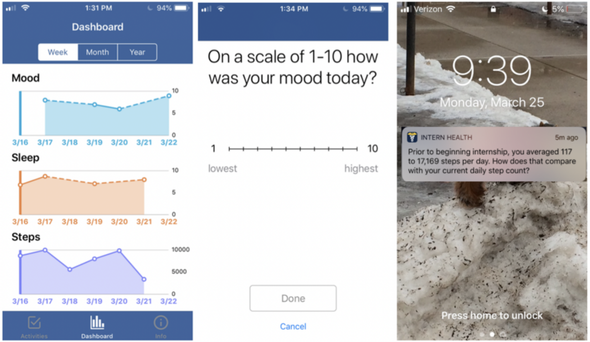


Figure: Causal diagrams for MDPs. Solid lines represent causal relationships. The parent nodes for the action is **not** specified in the model. $\boldsymbol{A_t}$ could either depend on $\boldsymbol{S_t}$ or the history.

Mobile Health Example: Intern Health Study

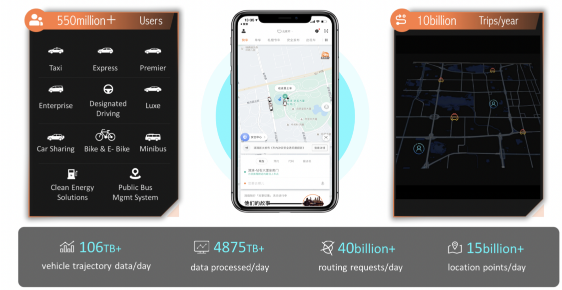
- **Physical** & **mental health** management
- **Subject**: First-year medical interns
- S_t: Interns' **mood scores**, **sleep hours** and **step counts**
- A_t: Send **text notifications** or not
- R_t: **Mood scores** or **step counts**

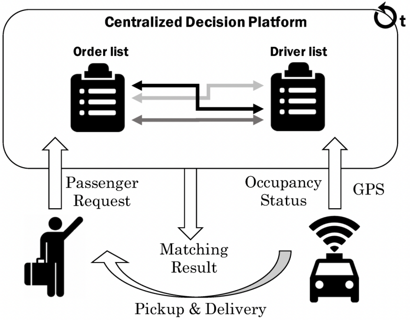


(i) App Dashboard (ii) Mood EMA (iii) Notifications

- The study lasts for half an year
- Treatment effects are usually **time-inhomogeneous** (decays over time)
- Leading to TMDPs

Ridesharing Example: Order-Dispatching





- S_t: **Supply** (drivers: availability, location) and **demand** (call orders: origin, destination)
- A_t: **Order-dispatching**: match a driver with an order
- R_t: **Answer rate/Completion rate/Drivers' income**
- Weekday-weekend differences, peak and off-peak differences lead to **time-inhomogeneity**

Partially Observable MDPs

- Difference between MDPs and POMDPs: states **fully-observable** or **partially-observable**
- The fully-observability assumption might be violated in practice
- In healthcare, patients' characteristics might not be fully recorded

Causal Diagram: POMDP

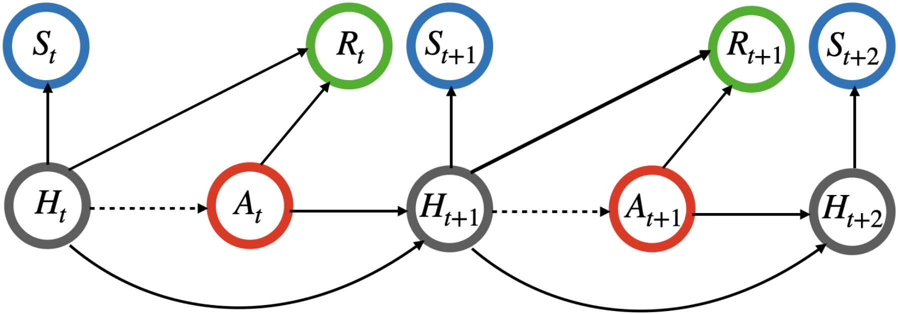
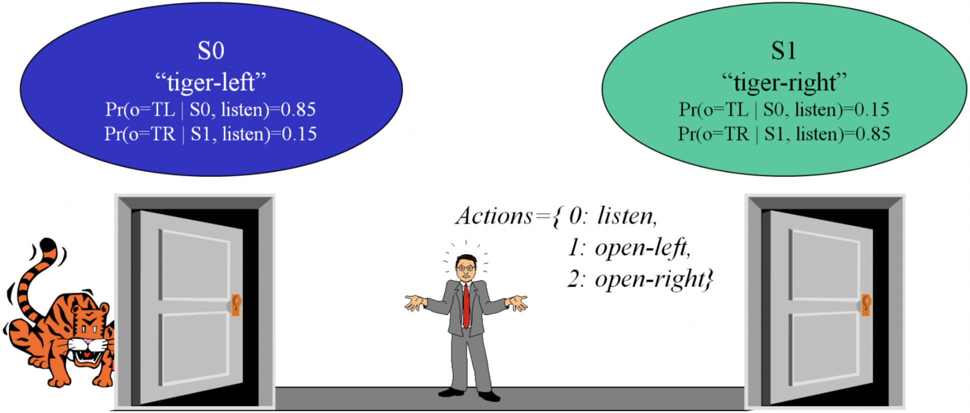


Figure: Causal diagrams for MDPs. Solid lines represent causal relationships. $\{\boldsymbol{H_t}\}_t$ denotes latent states. The parent nodes for the action is **not** specified in the model. $\boldsymbol{A_t}$ could either depend on $\boldsymbol{S_t}$ or the history.

Example: the Tiger Problem



Reward Function

- *Penalty for wrong opening: -100*
- *Reward for correct opening: +10*
- *Cost for listening action: -1*

Observations

- *to hear the tiger on the left (TL)*
- *to hear the tiger on the right(TR)*

Example: the Tiger Problem (Cont'd)

Suppose we choose to listen at each time

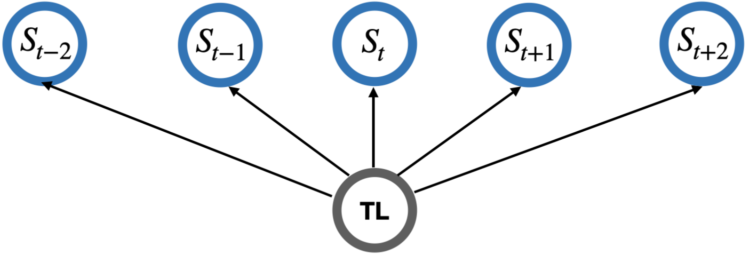


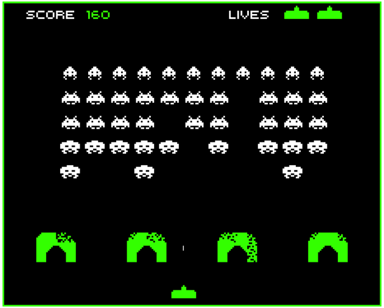
Figure: Causal diagram for the tiger problem. TL denotes the tiger location. S_t denotes the inferred location of the tiger at time t.

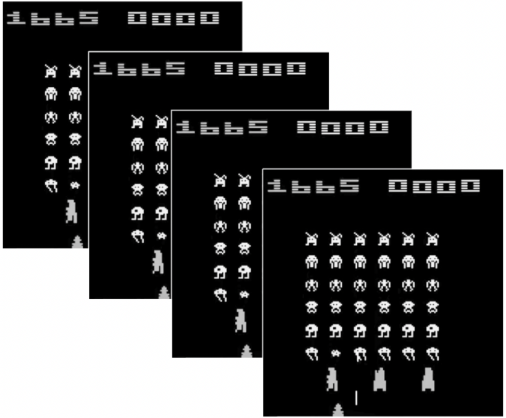
Converting non-MDPs into MDPs

- MDP assumptions: Markovianity & time-homogeneity
- To ensure **time-homogeneity**: include time variables in the state
- In ridesharing, include dummy variables weekdays/weekends & peak/off-peak hours
- In mobile health, use more recent observations
- To ensure **Markovianity**: concatenate measurements over multiple time steps

Stacking Frames in Atari Games

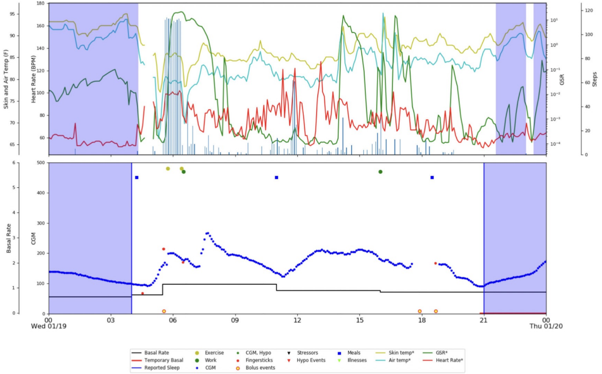
Input is a stack of 4 most recent frames [Mnih et al., 2015]





Concatenating Observations in Diabetes Study

- Management of **Type-I diabetes**
- **Subject**: Patients with diabetes.
- S_t: Patient's **glucose levels**, **food intake**, **exercise intensity**
- A_t: **Insulin doses** injected
- R_t: **Index of Glycemic Control** (function of patient's glucose level)



- Markovianity holds when concatenating 4 most recent observations [Shi et al., 2020]
- Concatenating observations also yield better policies

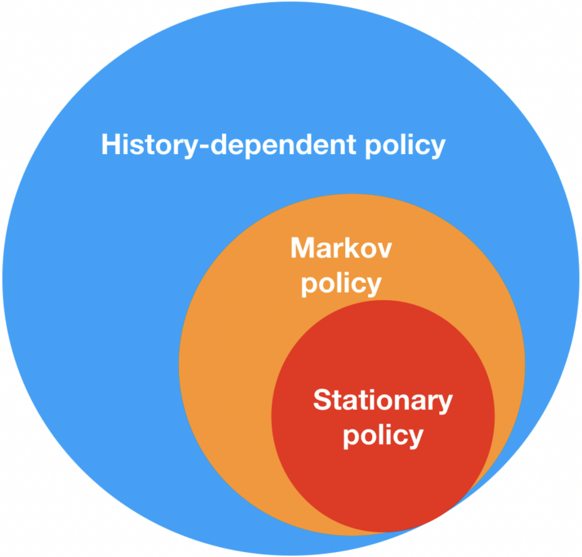
The Agent's Policy

- The agent implements a **mapping** π_t from the observed data to a probability distribution over actions at each time step
- The collection of these mappings $\pi = \{\pi_t\}_t$ is called **the agent's policy**:

$$\pi_t(a|\bar{s}) = \Pr(A_t = a|\bar{S}_t = \bar{s}),$$

  where $\bar{S}_t = (S_t, R_{t-1}, A_{t-1}, S_{t-1}, \cdots, R_0, A_0, S_0)$ is the set of **observed data history** up to time t.
- **History-Dependent** Policy: π_t depends on $\bar{S}_t$.
- **Markov** Policy: π_t depends on $\bar{S}_t$ only through S_t.
- **Stationary** Policy: π is Markov & π_t is **homogeneous** in t, i.e., $\pi_0 = \pi_1 = \cdots$.

The Agent's Policy (Cont'd)



The Agent's Policy (Cont'd)

- The collection of these mappings $\pi = \{\pi_t\}_t$ is called **the agent's policy**:

$$\pi_t(a|\bar{s}) = \Pr(A_t = a|\bar{S}_t = \bar{s}),$$

  where $\bar{S}_t = (S_t, R_{t-1}, A_{t-1}, S_{t-1}, \cdots, R_0, A_0, S_0)$.

- **Random** Policy: $\pi_t(\bullet|\bar{s})$ is a probability distribution over the action space
- **Deterministic** Policy: each probability distribution is degenerate
  - i.e., for any t and $\bar{s}$, $\pi_t(a|\bar{s}) = 1$ for some a and 0 for other actions
  - use $\pi_t(\bar{s})$ to denote the action that the agent selects

Goals, Objectives and the Return

The agent's goal: find a policy that maximizes the **expected return** received in long run

Definition (Return, Average Reward Setting)

The **return** G_t is the average reward from time-step t.

$$G_t = \lim_{T \to \infty} \frac{1}{T} \sum_{i=t}^{t+T-1} R_i.$$

Definition (Return, Discounted Reward Setting)

The **return** G_t is the cumulative discounted reward from time-step t.

$$G_t = \sum_{i=0}^{+\infty} \gamma^i R_{i+t}.$$

Discounted Reward Setting (Our Focus)

Definition (Return)

The **return** G_t is the cumulative discounted reward from time-step t.

$$G_t = \sum_{i=0}^{+\infty} \gamma^i R_{i+t}.$$

- The **discount factor** $0 \leq \gamma < 1$ represents the **trade-off** between **immediate** and **future** rewards.
- The value of receiving reward R after k time steps is $\gamma^k R$.
- $\gamma = 0$ leads to "**myopic**" evaluation
- γ close to 1 leads to "**far-sighted**" evaluation (close to the average reward)

Why Discount?

- **Mathematically convenient**: avoids infinite returns.
- **Computationally convenient**: easier to develop practical algorithms.
- In finance, immediate rewards earn more **interests** than delayed rewards
- Animal/human behaviour shows **preference** for immediate reward
  - Go to bed late and you'll be tired tomorrow
  - Eat heartily in winter and you'll need to trim fat in summer
- Possible to set $\gamma = \mathbf{1}$ in **finite horizon** settings (number of decision steps is finite; e.g., precision medicine applications where patients receive only a finite number of treatments)

(State) Value Function

Definition

The (state) value function $V^{\pi}(s)$ is expected return starting from s under π,

$$V^{\pi}(s) = \mathbb{E}^{\pi}(G_t | S_t = s) = \mathbb{E}^{\pi}\left(\sum_{i=0}^{+\infty} \gamma^i R_{i+t} | S_t = s\right).$$

- V^{π} is **independent** of the time t in its definition, under **time-homogeneity**
- $\mathbb{E}^{\pi}$ denotes the expectation assuming the system follows π

Bellman Equation

Definition

The Bellman equation for the state value function is given by

$$V^{\pi}(s) = \mathbb{E}^{\pi}\{R_t + \gamma V^{\pi}(S_{t+1}) | S_t = s\}.$$

- The value function can be **decomposed** into two parts:
  - Immediate reward R
  - discounted value of success state $\gamma V^{\pi}(S_{t+1})$
- Forms the basis for **value evaluation** (more in later lectures)

Bellman Equation (Proof)

$$
\begin{aligned}
V^\pi(s) &= \mathbb{E}^\pi(G_t | S_t = s) \\
&= \mathbb{E}^\pi(R_t + \gamma(R_{t+1} + \gamma R_{t+2} + \cdots) | S_t = s) \\
&= \mathbb{E}^\pi(R_t | S_t = s) + \gamma \mathbb{E}^\pi(G_{t+1} | S_t = s) \\
&= \mathbb{E}^\pi(R_t | S_t = s) + \gamma \mathbb{E}^\pi\{\mathbb{E}^\pi(G_{t+1} | S_{t+1}, S_t) | S_t = s\} \\
&= \mathbb{E}^\pi(R_t | S_t = s) + \gamma \mathbb{E}^\pi\{\mathbb{E}^\pi(G_{t+1} | S_{t+1}) | S_t = s\} \\
&= \mathbb{E}^\pi(R_t | S_t = s) + \gamma \mathbb{E}^\pi\{V^\pi(S_{t+1}) | S_t = s\},
\end{aligned}
$$

The second last equation holds due to the **Markov assumption**.

Bellman Optimality Equation

Definition

The Bellman optimality equation for the state-value function is given by

$$V^{\pi^{\text{opt}}}(s) = \max_{a} \mathbb{E}\{R_t + \gamma V^{\pi^{\text{opt}}}(S_{t+1}) | A_t = a, S_t = s\}.$$

- According to the Bellman equation,

$$V^{\pi^{\text{opt}}}(s) = \mathbb{E}^{\pi^{\text{opt}}}\{R_t + \gamma V^{\pi^{\text{opt}}}(S_{t+1}) | S_t = s\}.$$

- The optimal policy selects the action that maximizes the value: $\mathbb{E}^{\pi^{\text{opt}}} = \max_{a} \mathbb{E}$

1. General Reinforcement Learning (RL) Problems

2. Markov Decision Processes (MDPs)

3. Time-Varying MDPs and Partially Observable MDPs

4. Policy, Return and Value

5. The Existence of the Optimal Policy

Existence of Optimal Stationary Policy in MDPs

Theorem (See also Puterman [2014], Theorem 6.2.10)

Assume the state-action space is **discrete** *and the rewards are* **bounded***. Then there exists an* **optimal stationary policy** $\pi^{opt} = \{\pi_t^{opt}\}_t$ *such that*

- $\pi_1^{opt} = \pi_2^{opt} = \cdots = \pi_t^{opt} = \cdots$
- $\mathbb{E}^{\pi^{opt}} G_0 \geq \mathbb{E}^{\pi} G_0$ *for any* **history-dependent** *policy* π

- When the system dynamics satisfies the **Markov** and **time-homogeneity** assumption, so does the **optimal policy**.
- Lay the **foundation** for most existing RL algorithms
- Simplify the calculation since it suffices to focus on stationary policies

Existence of Optimal Markov Policy in TMDPs

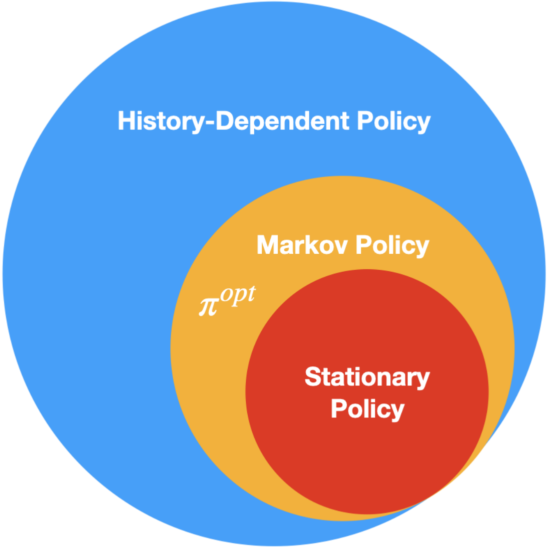
Theorem (See also Puterman [2014], Theorem 5.5.1)

Assume the state-action space is **discrete**. *Then there exists an* **optimal Markov policy** $\boldsymbol{\pi}^{opt} = \{\boldsymbol{\pi}_t^{opt}\}_t$ *such that*

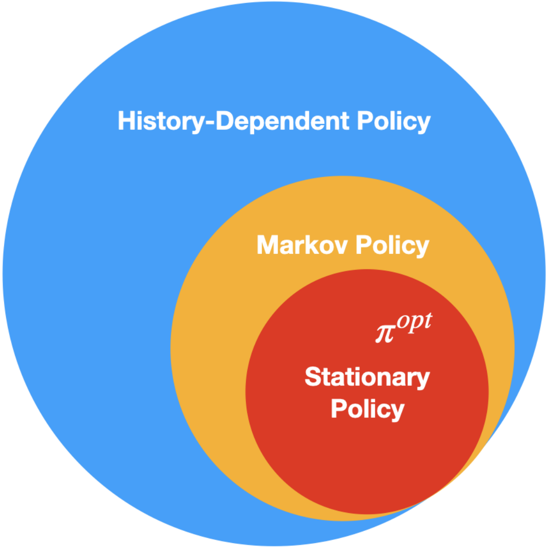
- *each* $\boldsymbol{\pi}_t^{opt}$ *depends on the data history only through* $\boldsymbol{S_t}$
- $\mathbb{E}^{\boldsymbol{\pi}^{opt}} \boldsymbol{G_0} \geq \mathbb{E}^{\boldsymbol{\pi}} \boldsymbol{G_0}$ *for any* **history-dependent** *policy* $\boldsymbol{\pi}$

When the system dynamics satisfies the **Markov** assumption, so does the **optimal policy**.

In TMDPs


History-Dependent Policy
Markov Policy
π^{opt}
Stationary Policy

In MDPs



History-Dependent Policy
Markov Policy
π^{opt}
Stationary Policy

Summary

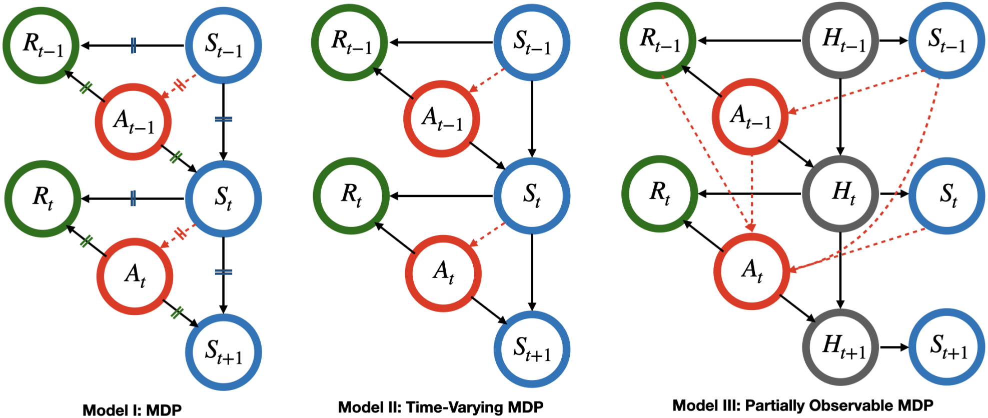
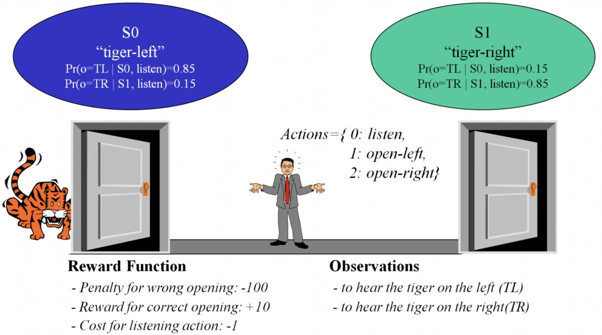


Figure: Causal diagrams for MDPs, TMDPs and POMDPs. Solid lines represent the causal relationships. Dashed lines indicate the information needed to implement the optimal policy. $\{H_t\}_t$ denotes latent variables. The parallel sign $\|$ indicates that the conditional probability function given parent nodes is equal.

Seminar

- Solution to HW1 (**Deadline**: Web 12pm)
- Demonstrating the difference between the form of optimal policy in MDPs and that in POMDPs using the Tiger problem



- A sketch of the proof of the **Existence of the Optimal Stationary Policy**

References I

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Chengchun Shi, Runzhe Wan, Rui Song, Wenbin Lu, and Ling Leng. Does the markov decision process fit the data: Testing for the markov property in sequential decision making. *arXiv preprint arXiv:2002.01751*, 2020.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Questions