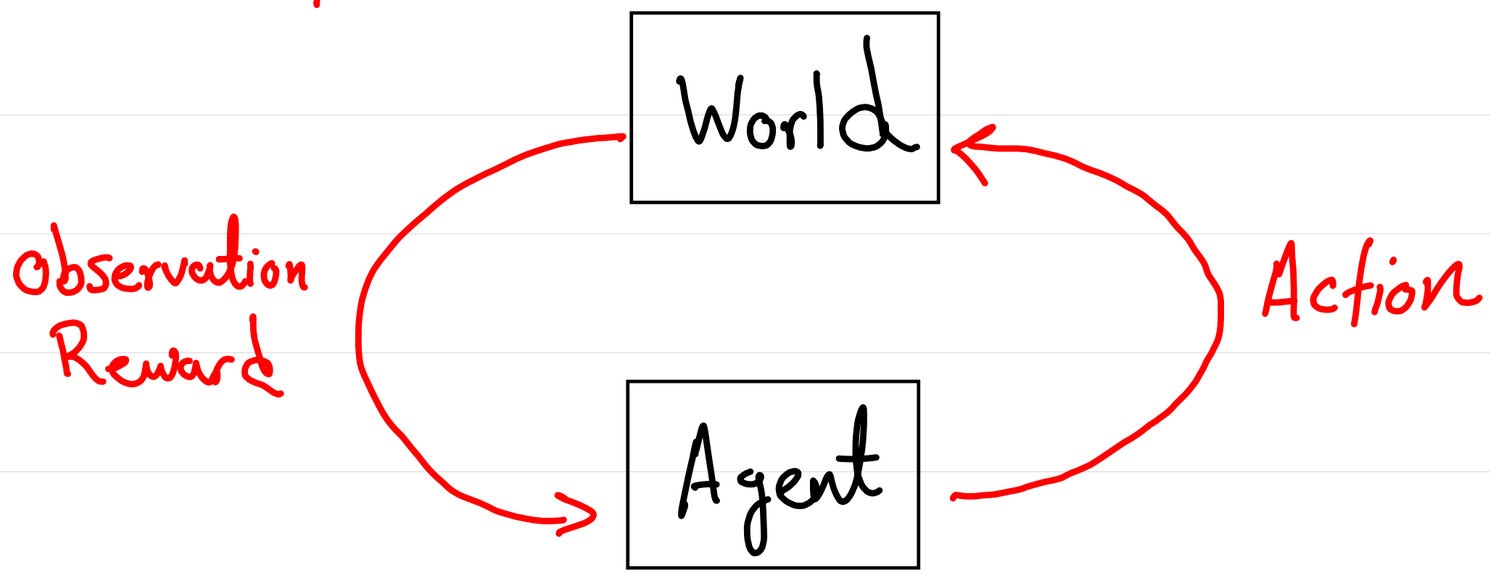
Week 11 - Part 2

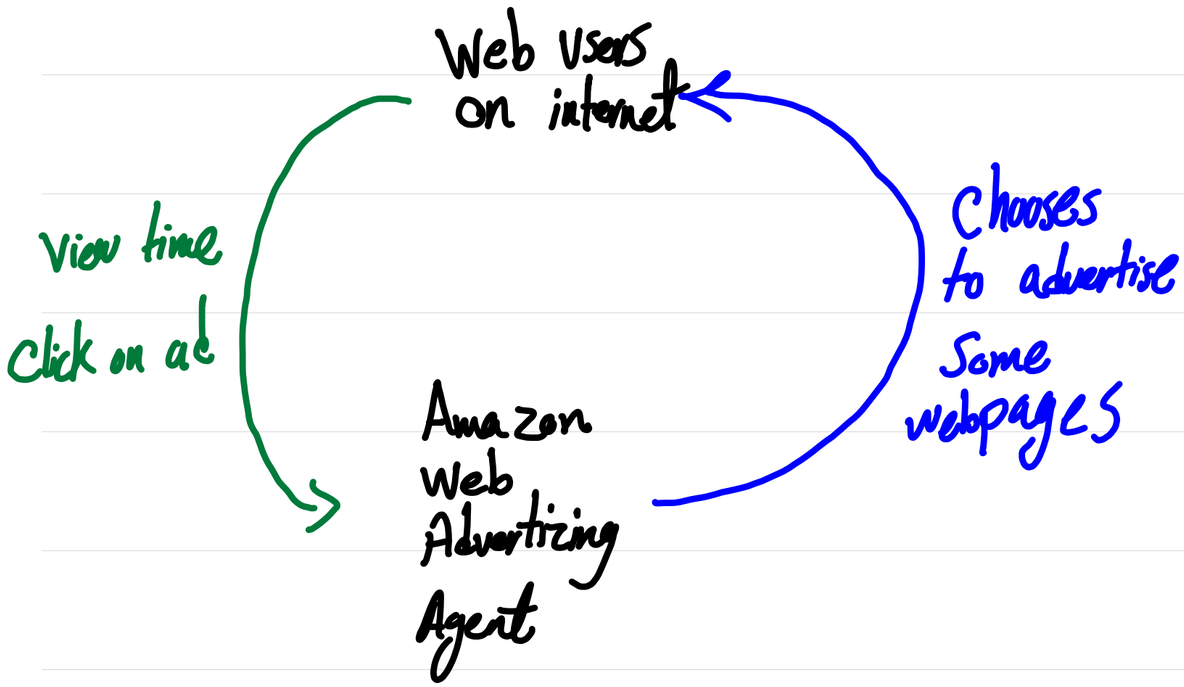
- Outcome of this part
  - Sequential Decision process
  - Observation, history, and state
  - Markov Decision Process (MDP)
    - What is Markov about MDP? Why Markov assumption is common?
  - Dynamics Model & Reward Model
    - Transition Graph
  - Return / Utility
  - Policies
  - Finding the "best" policy (i.e., solving the MDP) ← Next part

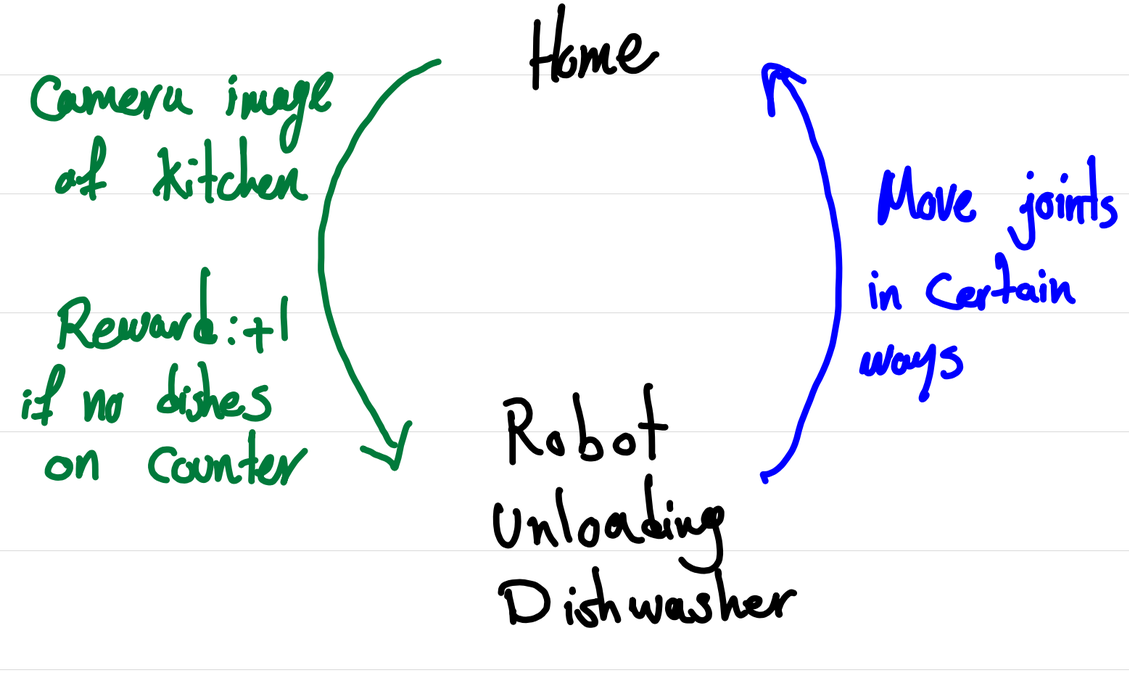
Sequential Decision Making



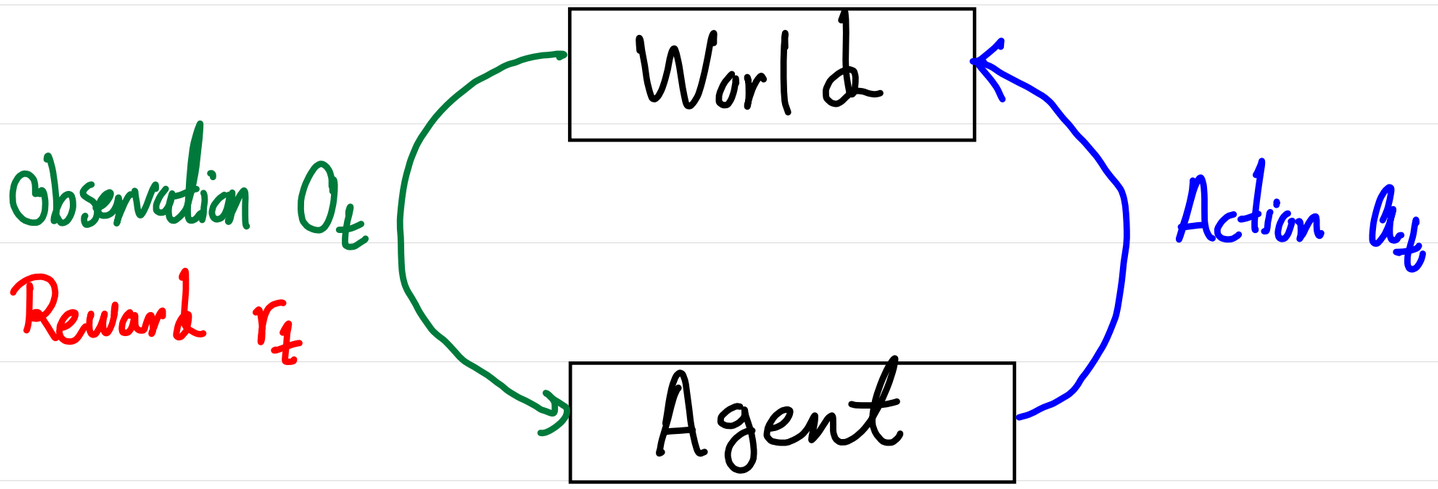
- **Goal:** Select actions to maximize total expected future reward
  - ↳ May require balancing immediate & long-term rewards

Some Examples



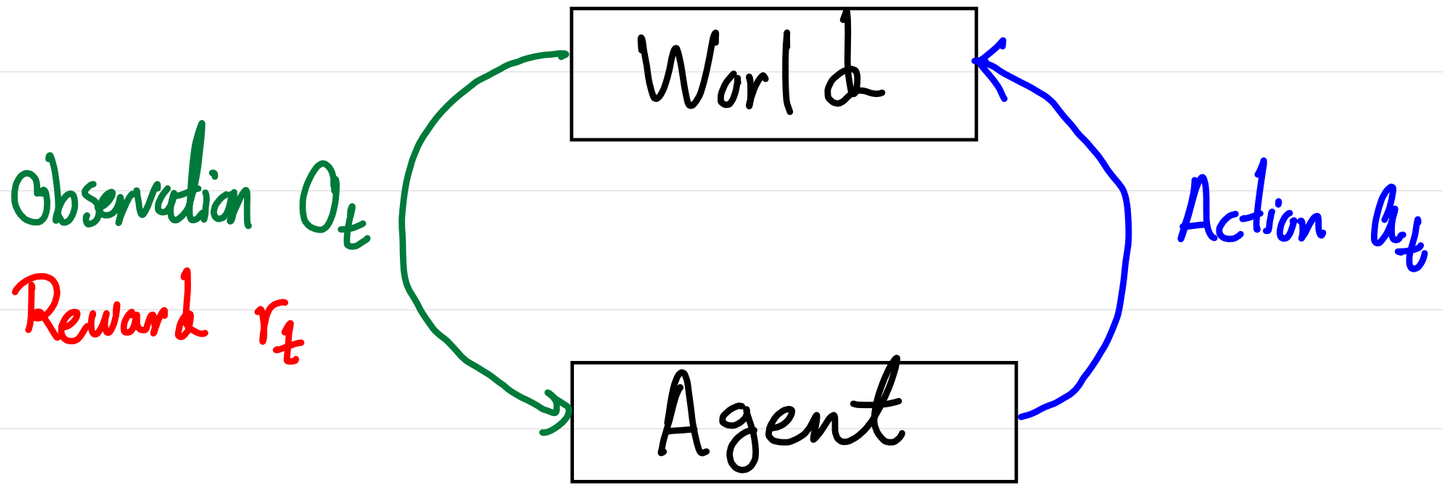


Sequential Decision Process: Agent & the World (Discrete Time)



- Each time step t:
  - Agent take an action a_t
  - Given action a_t, the world updates and
  -

History: Sequence of Past Observations, Actions, & Rewards



- History: $h_t = (a_0, O_1, r_1, a_1, O_2, r_2, \ldots, a_{t-1}, O_t, r_t)$
- Agent chooses action based on history
- State: information assumed to determine what happens next
  - State is a Function of history: $S_t = \psi(h_t)$

Observation V.S. History V.S State

- Can anyone tell me what is the difference between observation, history, and state?

- Consider the Atari game. How would you design/choose obeservation, history, and state?

Markov Assumption

- Often, to make problems tracktable, and because it is not a terrible assumption in reality, we will make Markov Assumption.

$$P(S_{t+1} \mid S_t, a_t) = P(S_{t+1} \mid S_t, a_t, S_{t-1}, a_{t-1}, S_{t-2}, a_{t-2}, \ldots, S_1, a_1)$$

- Future (S_{t+1}) is independent of past ($S_{t-1}, a_{t-1}, \ldots, S_1, a_1$) given the present ($S_t, a_t$)

Why is Markov Assumption Popular?

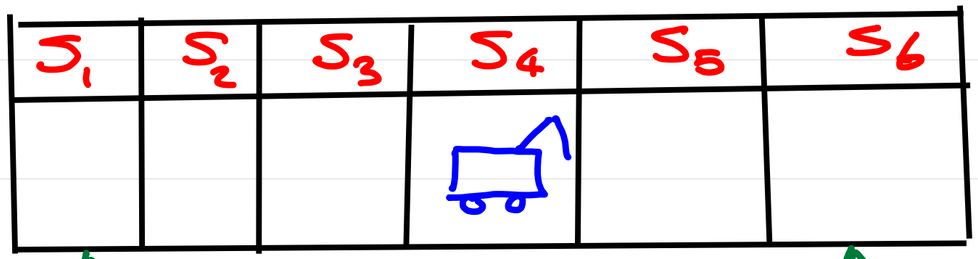
- It is simple and often can be satisfied if includes some previous observations as part of state
- There are many problems that we can satisfy Markov assumption by simply setting $S_t = O_t$.

- Markov assumption significantly simplifies working with the problem.

- Just like all ML problems we studied so far, there is this trade-off between the expressive power of state representation and how long it takes to train your model.

MDP Example: Mars Rover

locations on Mars →



s_1 and s_6 are the terminal locations, you obtain 1 and 5 samples from Soil in locations s_1 and s_6, respectively, and then the robot will shut down

- Mars Rover is the robot that you designed to collect samples on Mars.
- Your robot has to choose from going to right or left, to move between locations.
- Mars is strangely small. Only 6 locations on Mars.

State set:

Action Set:

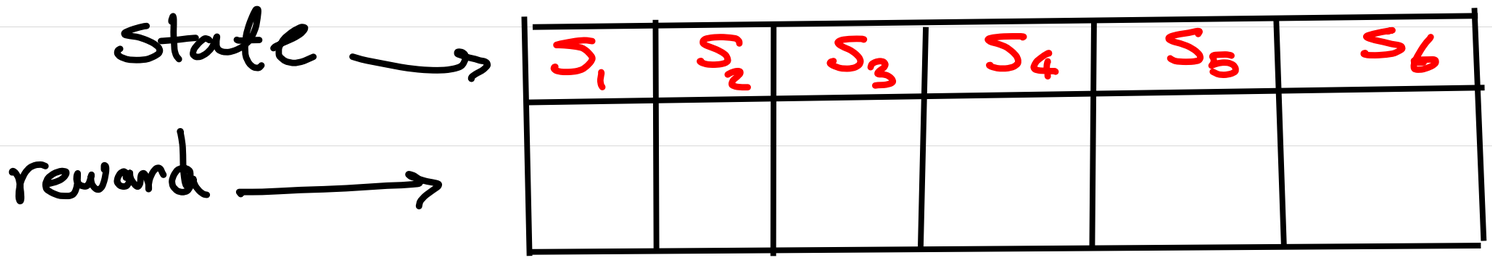
Rewards:

Dynamics Model & Reward Model

- Transition/dynamics model: predicts next state
- In other words, dynamics model specifies the distribution of outcomes when the agent makes a decision

- Reward model: predicts immediate reward:
  - Sometimes just , , or , depending on the model
  - Note that rewards can be zero, negative, or positive.

Example: Mars Rover Dynamics Model (Reward Model)



States & Actions:

Rewards Model:

It's Mars we are talking about. It's different from our Earth. Our robot is not completely predictible there. When the robot decides to go toward a direction, with 0.2 probability it may go the opposite direction.

Example: Mars Rover Dynamics Model (Dynamics Model)

state →

reward →

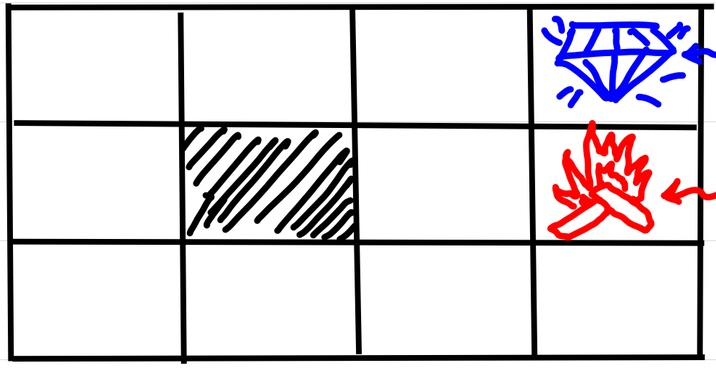
S_1	S_2	S_3	S_4	S_5	S_6

Transition Model:

It's Mars we are talking about. It's different from our Earth. Our robot is not Completely predictible there. When the robot decides to go toward a direction, with 0.2 probability it may go the opposite direction.

Example: Mars Rover In Grid World

- Consider the following Grid Model of Mars



Mars Rover gets a reward of (+1) if gets to this location

Mars Rover gets a reward of (-1) if it gets to this location

$R(s, a, s') = -0.03$ for all non-terminal states s'

- Noisy Movements on Mars: actions do not always go as planned
  - with prob. 0.8, the action takes the agent to the desired direction
  - with prob. 0.1, the actions takes the agent to a direction prependicular to the desired direction.
  - If there's a wall in the direction would have been taken to, it stays put

Homework: Mars Rover In Grid World

- formulate this MDP.

$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{1,0}$		$S_{1,2}$	$2_{1,3}$
$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$

Example: The undergrad student's life at UofT

- Erfan wants to thrive in his undergrad
- He has three states: happy, tired, burnt-out
  - burnt-out is the terminal state.
- He has two actions: "Work hard", "party hard"
  - Working hard gets double reward: reward 2 for working hard

S	a	S'	T(S, a, S')	R(S, a)
	Work hard		0.5	2
	Work hard		0.5	2
	party hard		1	1
	Work hard		1	2
	party hard		0.5	1
	party hard		0.5	1
	end		1	0

We can summarize the dynamics and reward model with a table

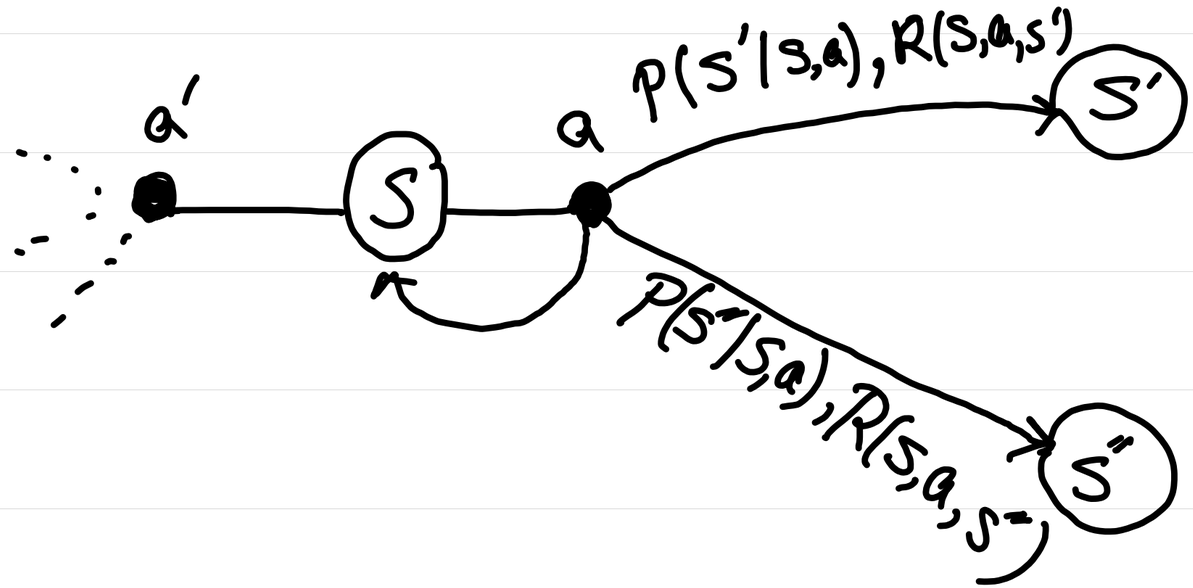
Transition Graph

- Transition Graph is a useful way of summarizing the dynamics of a finite MDP
- There are two types of nodes
  - state nodes: There is a state node for each possible state
    - Typically a large circle labeled by the name of the state, e.g., (s)
  - Action nodes (also called q-state node): There is an action node for each state-action pair.
    - Typically a small solid circle labeled by the name of action or (state, action) pair and connected by a line to the state node, e.g., $\bullet^{a}$ or $\bullet^{(s,a)}$

Transition Graph

- Starting in state "s" and taking action "a" move you along the line from state nods "S" to action node (s, a).

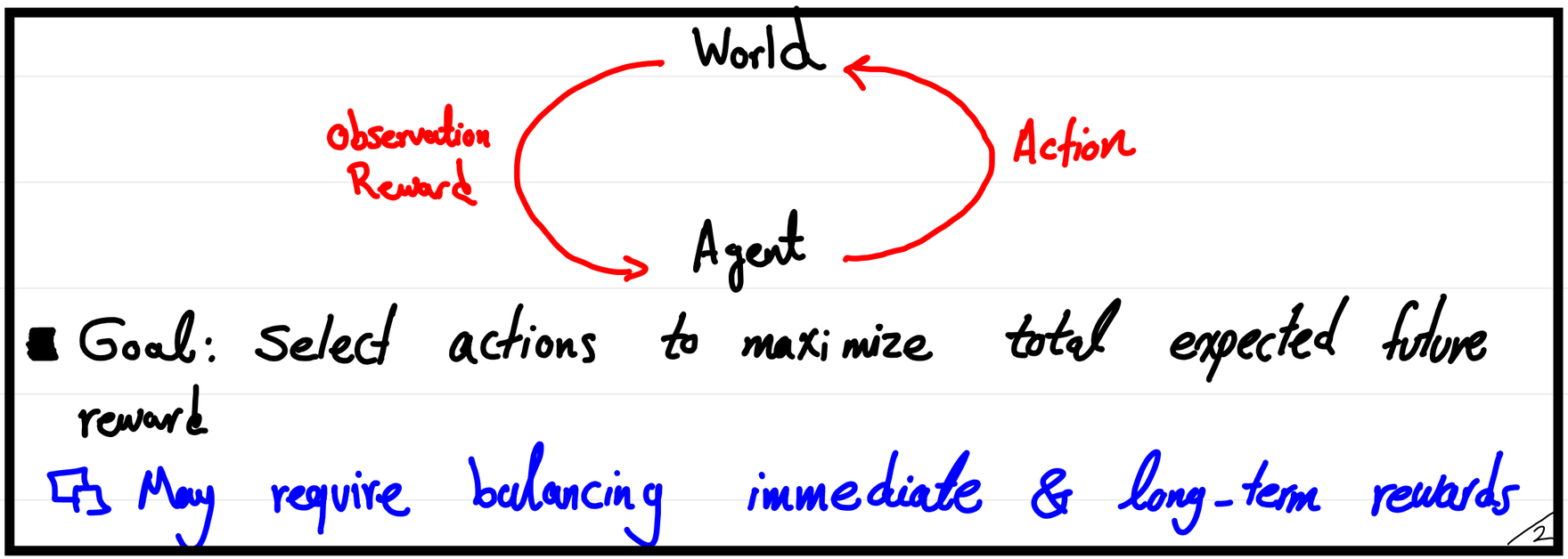
- Then the environment responds with a transition to the next state's node.



Activity: Draw the transition Graph for Erfan the Undergrad

Returns & Episodes

- Recall:



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

- The agent's goal is to: maximize the expected return
  - ↳ Return is a function of the reward sequence

Return

- In simplest case, the return is the sum of the rewards
  - i.e., $r_0 + r_1 + \cdots + r_T$

- Such return definition makes sense in applications with finite time steps.

- However, in applications with possibly infinite time step, i.e. $T = \infty$, the above definition of return is problematic
  - as the return can be ∞.

Return as Sum of Discounted Rewards

- That's why we use a more complex definition for return

$$U([r_0, r_1, r_2, \cdots]) = r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots = \sum_{k=0}^{\infty} \gamma^k r_k$$

This return is known a "sum of discounted rewards"

$0 \leq \gamma \leq 1$ is the discount factor

- The discount parameter determines the present value of the feature rewards:
  - A reward received K time step in feature is worth only γ^K times what would be worth if it were received immediately.
- As γ approaches 1, the return objective takes into account the feature rewards more strongly, i.e., the agent becomes more farsighted
- As γ approaches 0, the return objective take into account the immediate reward more strongly, i.e., the agent is more myopic (shortsighted)

Why Discounted Return is a good Model

i) As mentioned earlier, with discount factor of $\gamma < 1$, return is always bounded (i.e., finite) even when we have infinite horizon. That's because $\sum_{k=0}^{\infty} \gamma^k r_k$ is bounded if the sequence $\{r_k\}$ is finite.

Why Discounted Return is a good Model

ii) Intuitively speaking, discount return model conforms both human and animal behaviour.

☐ We care about the reward we will get in future, but we value the immediate reward more

iii) We used return function to model agent's preference over possible outcomes, i.e., reward sequence.

A common preference assumption is stationarity, i.e.,

$$[a_1, a_2, \dots] \succ [b_1, b_2, \dots] \iff [r, a_1, a_2, \dots] \succ [r, b_1, b_2, \dots]$$

There is only one way to model a stationary preference:

$$U([r_0, r_1, \dots]) = r_0 + \gamma r_1 + \gamma^2 r_2 + \dots, \text{ where } 0 \leq \gamma \leq 1.$$

Policies

- A policy π, gives an action for each state
  - $\pi : S \rightarrow A$
- Now that we defined return/utility as the sum of discounted rewards, we can compare policies by comparing the expected utilities obtained when they are followed

- For MDPs, we want to find an optimal policy $\pi^* : S \rightarrow A$
  - An optimal policy π^*, is the policy that maximizes the expected return if followed.
- Note: The above was the definition of deterministic policies.

Stochastic Polices : $\pi(a|s) = \mathbb{P}[a_t = a | s_t = s]$

Example: Erfan the undergrad student

- Recall the example "Erfan the undergrad".
  - $A = \{\underbrace{\text{"party hard"}}_{a_p}, \underbrace{\text{"work hard"}}_{a_w}, \underbrace{\text{"end"}}_{a_e}\}$
  - $S = \{\underbrace{\text{"happy"}}_{s_h}, \underbrace{\text{"tired"}}_{s_t}, \underbrace{\text{"burnt-out"}}_{s_b}\}$
- List all possible policies for the above MDP.
  - Note that the only possible action at s_b is a_e.

Expected Return

- Following a policy yields a random episode
- The utility of a policy is the sum of discounted rewards of the episode
  - This utility is a random variable

- Let's study a simple example:

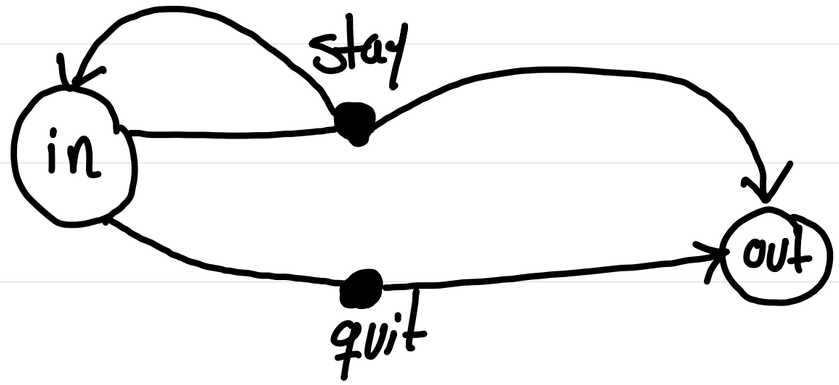
Erfan the gambler

(See the next page for the description)

Example: Erfan the Gambler

- For each round $r = 1, 2, \ldots$
  - You choose stay or quit
  - If quit, you get \$10 and we end the game
  - If stay, you get \$4 and I roll a 6-sided fair dice
    - If the dice results in 1 or 2, we end the game
    - Otherwise, continue to the next round.

- Let's draw the transition graph.



- Let the discount factor be $\gamma = 1$.
- Here are some possible episodes if Erfan's policy is to stay:

Episode	Return
[(in, stay, 4), out]	4
[(in, stay, 4), (in, stay, 4), (in, stay, 4), out]	12
[(in, stay, 4), (in, stay, 4), end]	8
⋮	⋮

- Let's formalize what we saw in the previous example for general MDP
- The value (utility) of policy π at state s, denoted by $V_\pi(s)$, is the expected return if starting at s and following policy π
- Starting at state "s" and following policy "π" can yield to different sequences of (state, action, reward), e.g.,
  - $(s, \pi(s), R(s,\pi(s))), (s', \pi(s'), R(s',\pi(s'))), (s'', \pi(s''), R(s'',\pi(s'')), \ldots$
  - $(s, \pi(s), R(s,\pi(s))), (s', \pi(s'), R(s',\pi(s'))), (s''', \pi(s'''), R(s''',\pi(s''')), \ldots$
  - $(s, \pi(s), R(s,\pi(s))), (s''', \pi(s'''), R(s''',\pi(s''))), (s', \pi(s'), R(s',\pi(s'))), \ldots$
  - $(s, \pi(s), R(s,\pi(s))), (s'', \pi(s''), R(s'',\pi(s''))), (s', \pi(s'), R(s',\pi(s'))), \ldots$
  - ⋮
- Each one of the possible episodes happens with a probability

Note: in the above MDP, we assumed a reward model of type $R(s,a)$ for simplicity

* Finding the Expected Return

- How can we formulate $V_\pi(s)$?

$$V_\pi(s) = \mathop{\mathbb{E}}_{(s_1,s_2,\dots)\,|\,s_0=s}\left[R(s_0,\pi(s_0)) + \gamma R(s_1,\pi(s_1)) + \gamma^2 R(s_2,\pi(s_2)) + \dots \,\middle|\, (s_0=s,\pi(s))\right]$$

where $s_1, s_2, \dots$ is a random sequence of states yield by following policy π. So $s_i \in S$ for any $i \geq 1$.

- What is the probability of observing the sequence $s_1, s_2, s_3, \dots$ Conditioned on $s_0 = s$? $P(s_1 \mid s, \pi(s)) \times P(s_2 \mid s_1, \pi(s_1)) \times \dots$

- We can rewrite $V_\pi(s)$ as

$$V_\pi(s) = \mathop{\mathbb{E}}_{(s_1,s_2,\dots)\,|\,s_0=s}\left[R(s,\pi(s)) + \gamma\left(R(s_1,\pi(s_1)) + \gamma R(s_2,\pi(s_2)) + \dots\right) \,\middle|\, (s_0=s,\pi(s))\right]$$

$$= R(s,\pi(s)) + \gamma \mathop{\mathbb{E}}_{(s_1,s_2,\dots)\,|\,s_0=s}\left[R(s_1,\pi(s_1)) + \gamma R(s_2,\pi(s_2)) + \dots \,\middle|\, (s_0=s,\pi(s))\right]$$

* Finding the Expected Return

- We can rewrite $V_\pi(s)$ as

$$V_\pi(s) = R(s,\pi(s)) + \gamma \mathop{\mathbb{E}}_{(s_1,s_2,\dots)|s_0=s} \left[R(s_1,\pi(s_1)) + \gamma R(s_2,\pi(s_2)) + \dots \,\middle|\, (s_0=s,\pi(s)) \right]$$

- **Recall**: Law of total probability $\mathop{\mathbb{E}}_{x}(X) = \sum_i P(A_i) \mathop{\mathbb{E}}_{x|A_i}(X|A_i)$ where $\{A_i\}$ is a finite partition of the sample space.

- **Recall**: Let us rewrite the above recall with an additional condition, so that it is more suitable for us.

$$\mathbb{E}(X|B) = \sum_i P(A_i|B) \mathop{\mathbb{E}}_{x|B,A_i}(X|B,A_i)$$

- Thus, we can rewrite $V_\pi(s)$ as

$$V_\pi(s) = R(s,\pi(s)) + \gamma \sum_{s' \in S} P(s_1=s' \mid s_0=s, \pi(s_0)) \mathop{\mathbb{E}}_{(s_1,s_2,s_3,\dots)|s_1=s', s_0=s} \left[R(s_1,\pi(s_1)) + \gamma R(s_2,\pi(s_2)) + \dots \,\middle|\, (s_0=s,\pi(s_0)), (s_1=s',\pi(s')) \right]$$

* Finding the Expected Return

■ Thus, we can rewrite $V_\pi(s)$ as

$$V_\pi(s) = R(s,\pi(s)) + \gamma \sum_{s' \in S} P(s_1 = s' \mid s_0 = s, \pi(s_0)) \underset{(s_2, s_3, \ldots) \mid s_1 = s'}{\mathbb{E}} \left[R(s_1, \pi(s_1)) + \gamma R(s_2, \pi(s_2)) + \ldots \mid (s_0 = s, \pi(s_0)), (s_1 = s', \pi(s')) \right]$$

$$\Rightarrow V_\pi(s) = R(s,\pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') \underset{(s_2, s_3, \ldots) \mid s_1 = s'}{\mathbb{E}} \left[R(s', \pi(s')) + \gamma R(s_2, \pi(s_2)) + \ldots \mid (s_1 = s', \pi(s')) \right]$$

$$\Rightarrow V_\pi(s) = R(s,\pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') \, V_\pi(s')$$

Finding the Expected Return

- We found a recursion for $V_\pi(s)$ under the dynamics model $T(s,a,s')$ and reward model $R(s,a)$

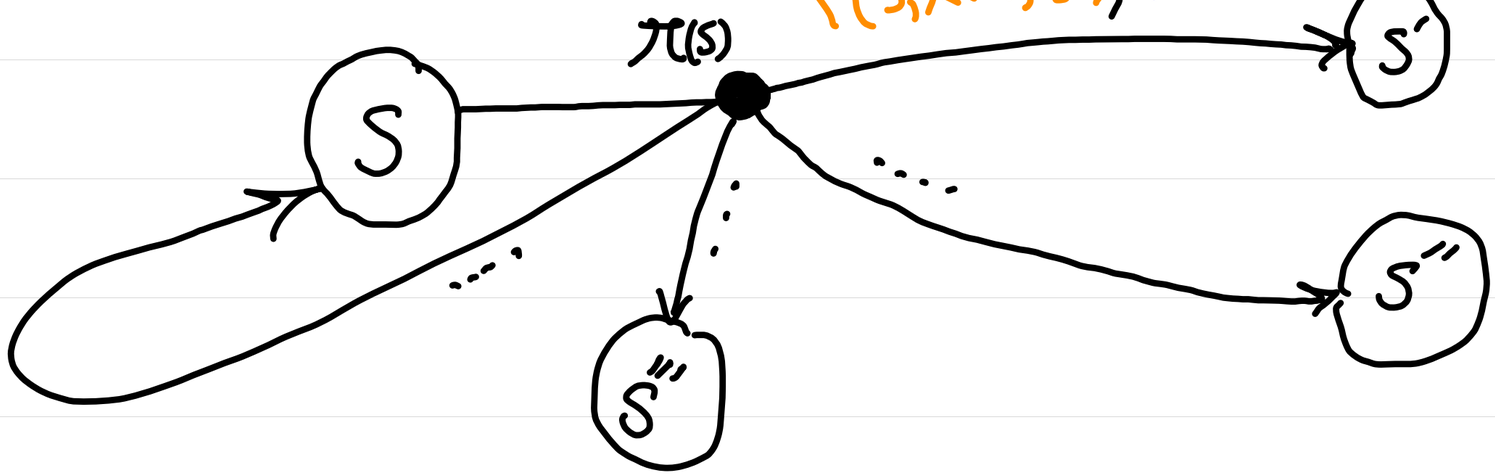
$$V_\pi(s) = R(s,\pi(s)) + \gamma \sum_{s'} T(s,\pi(s),s')\, V_\pi(s')$$

- More generally, we can show that for dynamics model $T(s,a,s')$ and Reward model $R(s,a,s')$

$$V_\pi(s) = \sum_{s' \in S} T(s,\pi(s),s') \left[R(s,\pi(s),s') + \gamma V_\pi(s') \right]$$

Finding the Expected Return

- An intuitive explanation



$$V_\pi(s) = T(\ ,\ ,\ )[R(\ ,\ ,\ ) + \gamma V_\pi(\ )]$$
$$+ T(\ ,\ ,\ )[R(\ ,\ ,\ ) + \gamma V_\pi(\ )]$$
$$+ T(\ ,\ ,\ )[R(\ ,\ ,\ ) + \gamma V_\pi(\ )]$$

with probability $T(s,\pi(s),s')$, you will receive the immediate reward $R(s,\pi(s),s')$ and go to the next state s'. From state s', your expected utility that you would receive is $V_\pi(s')$. This is the reward you receive in the next step. Hence, you should discount it by γ.

Value and Q-value of a Policy

- We use $V_\pi(s)$ to denote the expected utility received by following policy "π" from state "s".

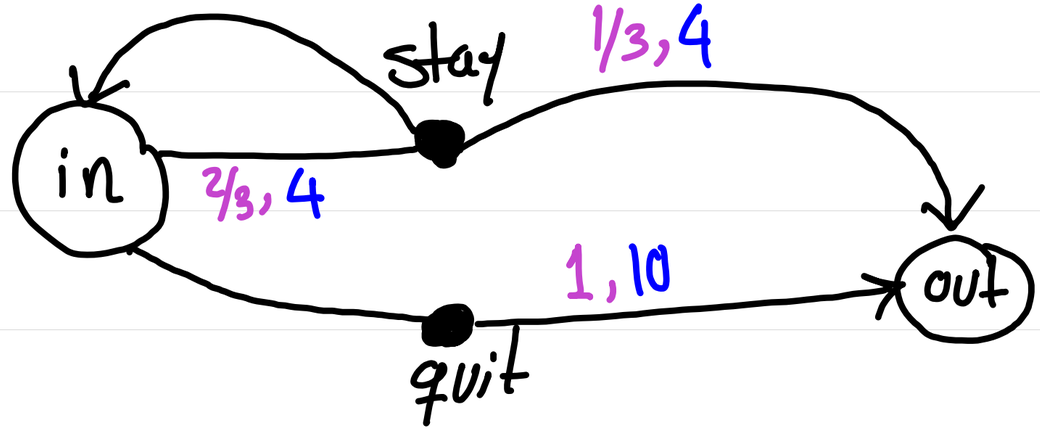
- We use $Q_\pi(s,a)$ to denote the expected utility of taking action "a" from state "s", and then following policy "π".

$V_\pi(s)$ ← value of a policy

$Q_\pi(s,a)$ ← Q-value of a policy

Example: Erfan the gambler

- Let's revisit that example and find the value of "stay" policy.



- Let π_1 denote the "stay" policy, i.e., $\pi_1(\text{"in"}) = \text{"stay"}$
  - Note that "out" is a terminal state and usually the value of terminal states is already known.
  - In this problem, when landing in "out" state, the game ends and we receive no more reward.
    - Thus, $V_\pi(\text{"out"}) = 0$, for any policy π.

- Observe that

$$V_{\pi_1}(\text{"in"}) = \sum_{s'} T(\text{"in"}, \pi_1(\text{in}), s') \left[R(\text{in}, \pi_1(\text{in}), s') + \gamma V_{\pi_1}(s') \right]$$

$$=$$

- Let's find the valve of "quit" policy.
- Let π_2 denote the "quit" policy, i.e., $\pi_2(\text{"in"}) = \text{"quit"}$.

$$V_{\pi_2}(\text{in}) = \sum_{s'} T(\text{in}, \pi_2(\text{in}), s') \left[R(\text{in}, \pi_2(\text{in}), s') + \gamma V_{\pi_2}(s') \right]$$

$$=$$

- Which policy do you think is better?

Next Lecture: Policy evaluation

- We could find the value of the states in our previous example easily.
  - we just had one unknown state. Easy Peasy!

- What about larger problems? $S = \{s, s', s'', s''', \dots\}$

$$V_\pi(s) = T(s,\pi(s),s)\left[R(s,\pi(s),s)+\gamma V_\pi(s)\right] + T(s,\pi(s),s')\left[R(s,\pi(s),s')+\gamma V_\pi(s')\right]+\dots$$

$$V_\pi(s') = T(s',\pi(s'),s)\left[R(s',\pi(s'),s)+\gamma V_\pi(s)\right] + T(s',\pi(s'),s')\left[R(s',\pi(s'),s')+\gamma V_\pi(s')\right]+\dots$$

$$V_\pi(s'') = T(s'',\pi(s'),s)\left[R(s'',\pi(s''),s)+\gamma V_\pi(s)\right] + T(s'',\pi(s''),s')\left[R(s'',\pi(s''),s')+\gamma V_\pi(s')\right]+\dots$$

Next Lecture: Policy evaluation

- What about larger problems? $S = \{s, s', s'', s''', \dots\}$

$$V_\pi(s) = T(s,\pi(s),s)\left[R(s,\pi(s),s)+\gamma V_\pi(s)\right] + T(s,\pi(s),s')\left[R(s,\pi(s),s')+\gamma V_\pi(s')\right]+\dots$$

$$V_\pi(s') = T(s',\pi(s'),s)\left[R(s',\pi(s'),s)+\gamma V_\pi(s)\right] + T(s',\pi(s'),s')\left[R(s',\pi(s'),s')+\gamma V_\pi(s')\right]+\dots$$

$$V_\pi(s'') = T(s'',\pi(s'),s)\left[R(s'',\pi(s''),s)+\gamma V_\pi(s)\right] + T(s'',\pi(s''),s')\left[R(s'',\pi(s''),s')+\gamma V_\pi(s')\right]+\dots$$

$\vdots$

- Let $\underline{V} = \begin{bmatrix} V_\pi(s) \\ V_\pi(s') \\ \vdots \end{bmatrix}$, $T = \begin{bmatrix} T(s,\pi(s),s) & T(s,\pi(s),s') & \dots \\ T(s',\pi(s'),s) & T(s',\pi(s'),s') & \dots \\ \vdots & \ddots & \end{bmatrix}$, $R = \begin{bmatrix} R(s,\pi(s),s) & R(s,\pi(s),s') & \dots \\ R(s',\pi(s'),s) & R(s',\pi(s'),s') & \dots \\ \vdots & \ddots & \end{bmatrix}$

- So, $\underline{V} = (T \odot R)\underline{1} + \gamma T \underline{V}$
  - $T \odot R$ is the hadamard product (i.e., element-wise product) of T and R.
  - $(T \odot R)\underline{1} = \begin{bmatrix} T(s,\pi(s),s)R(s,\pi(s),s') + T(s,\pi(s),s')R(s,\pi(s),s') + \cdots \\ T(s',\pi(s'),s)R(s',\pi(s'),s) + T(s',\pi(s'),s')R(s',\pi(s'),s') + \cdots \\ \vdots \end{bmatrix}$
- Thus, $\underline{V} = (I - \gamma T)^{-1}\left((T \odot R)\underline{1}\right)$