Reinforcement Learning(1)

(How agents can learn to decide?)

04/19/2021

Announcement

No lab tomorrow (rejuvenation day)

What we learned last week?

How an agent can make decisions under the assumption that there is no uncertainty involved?

What we will learn this ee?

Big topic
We only go through examples, and some concepts

How an agent can **learn** to make decisions under **unc**

Today:

Reward

Policy

Wednesday:

RL, q-learning, Deep RL

Intuition

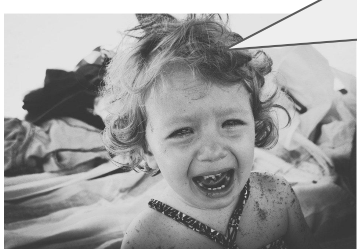
How to learn to improve based on direct experience?

Poor performance

Good performance

Intuition

Negative feedback (pain,losing money, crying, ...)



We usually try to avoid decisions that lead to receiving negative feedback. Instead, we try to

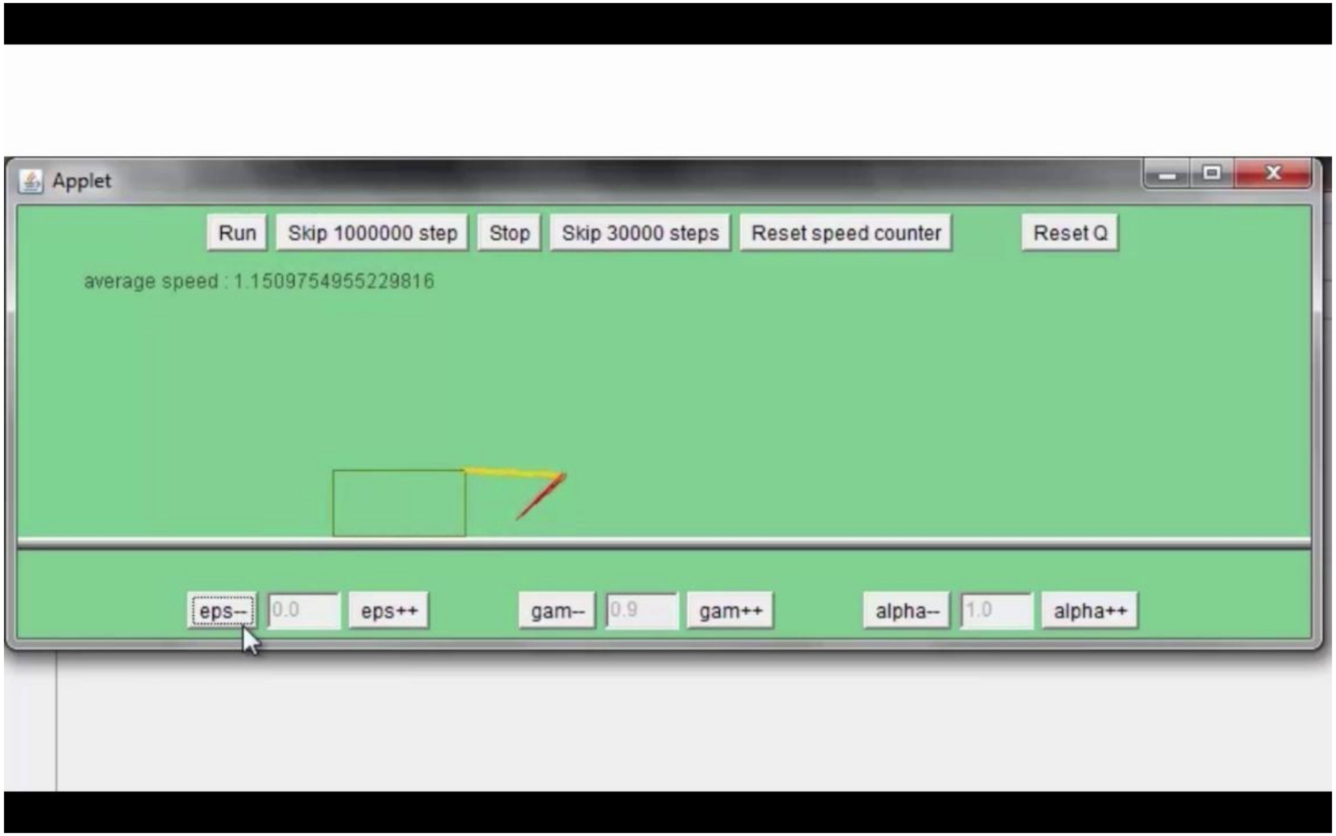
Positive feedback (pleasure, earn money, ...)

Motivation

The robot interacts with the environment
And tries to take actions so that it receives positive feedback

[Kohl & Stone, 2004]

Motivation: Crawler Bot



What we have learned so far?

Think:

Reasoning and
knowledge representation

Learn:

Supervised learning:

Reinforcement learning

Act:

Classical planning

Probabilistic planning

Reinforcement Learning

What if?

What if we are not **certain** about action execution outcome?

In addition, what if the agent doesn't have that much information about the **environment** it is operating in?

How it can take actions under the above conditions?

Planning under uncertainty

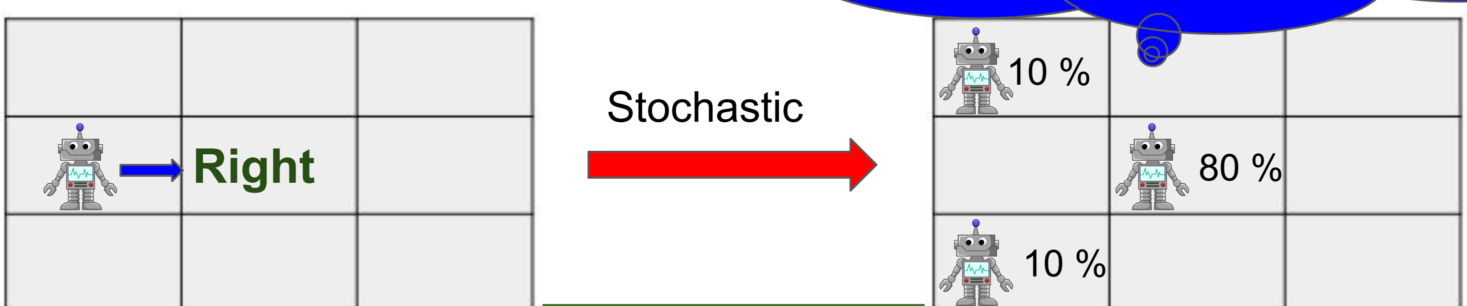
Act in the presence of uncertainty toward maximizing long-term utility

Reinforcement Learning

Learn to act under uncertainty toward maximizing long-term utility

Recap: Deterministic vs. Stochastic:

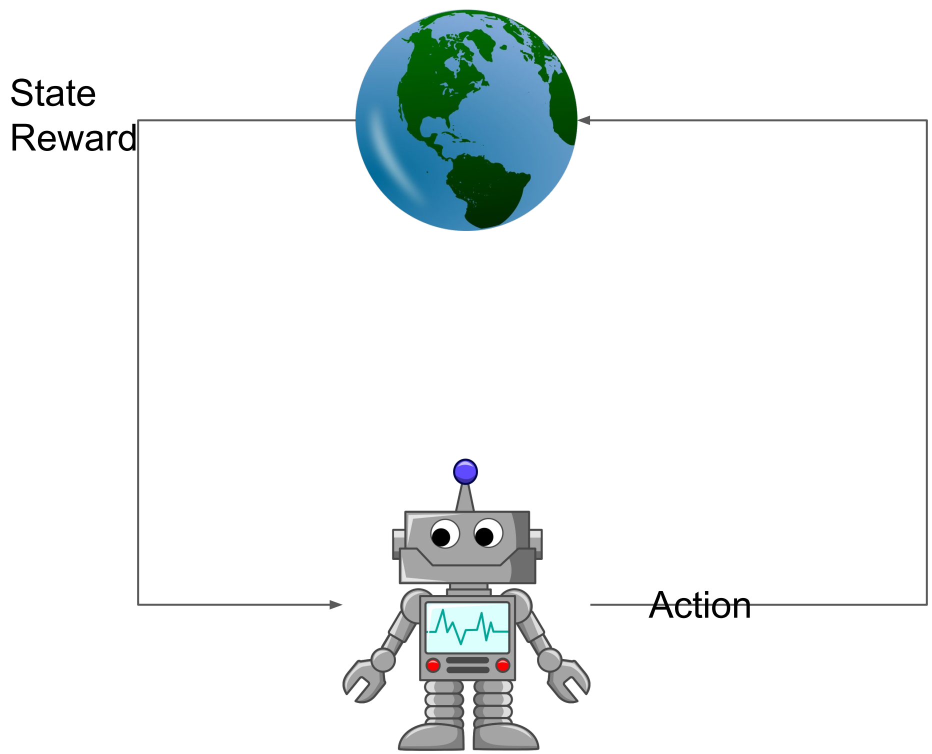
Need probabilistic planning to model the uncertainty



Reinforcement Learning

A computational approach to learning from interaction

Maximize
long-term
reward



Important concepts

Reward (R): The signal the environment sends to the agent

- a single number (could be negative).
- The objective is to maximize the total reward it receives over the long run.
- It defines what are the good and bad events for the agent.
- In a biological system, we might think of rewards as analogous to the experiences of pleasure or pain.

Important concepts

Value V(s): Total amount of reward an agent can expect to accumulate over the future, starting from state **s**.

Policy (π): a policy is a mapping from perceived states of the environment to actions to be taken

Policy could be optimal, or suboptimal

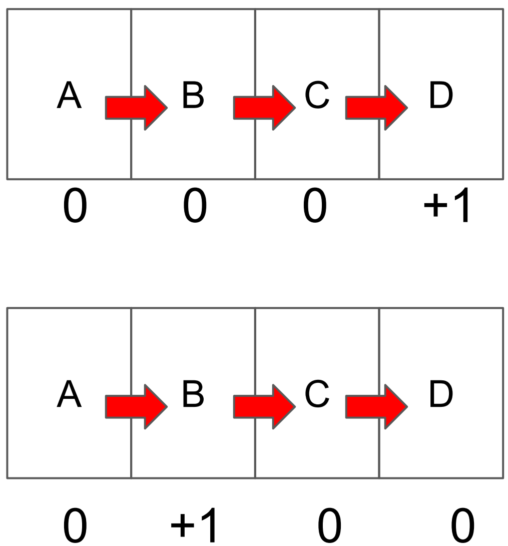
Early vs. late rewards

Earlier rewards are preferable

e.g, receiving a prize now vs. a year from now.

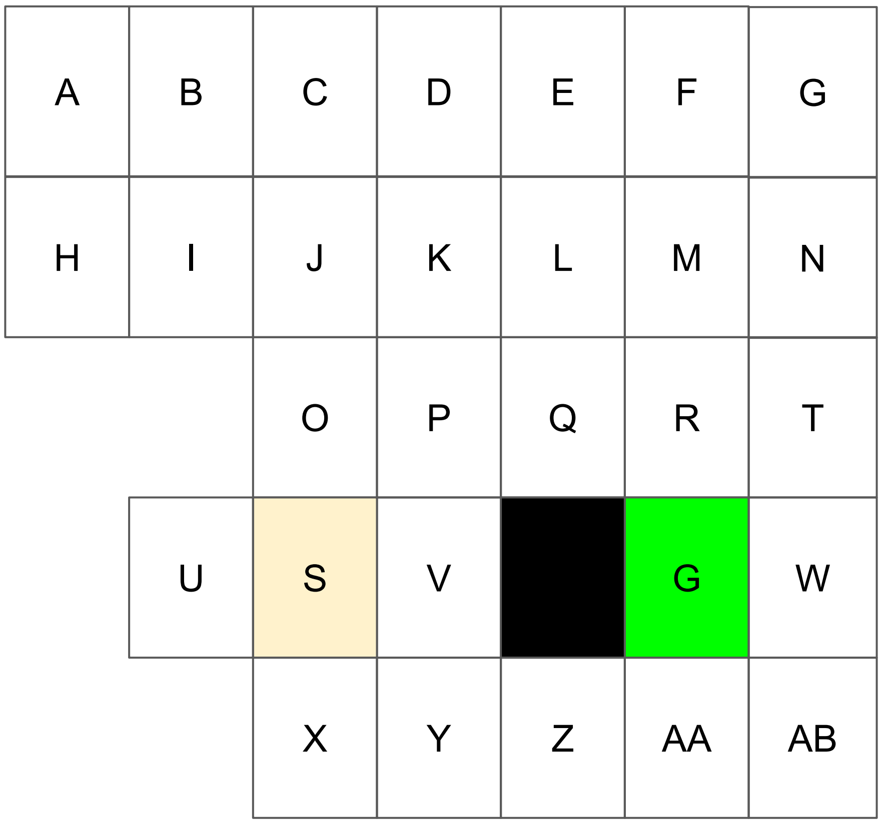
(A)

(B)



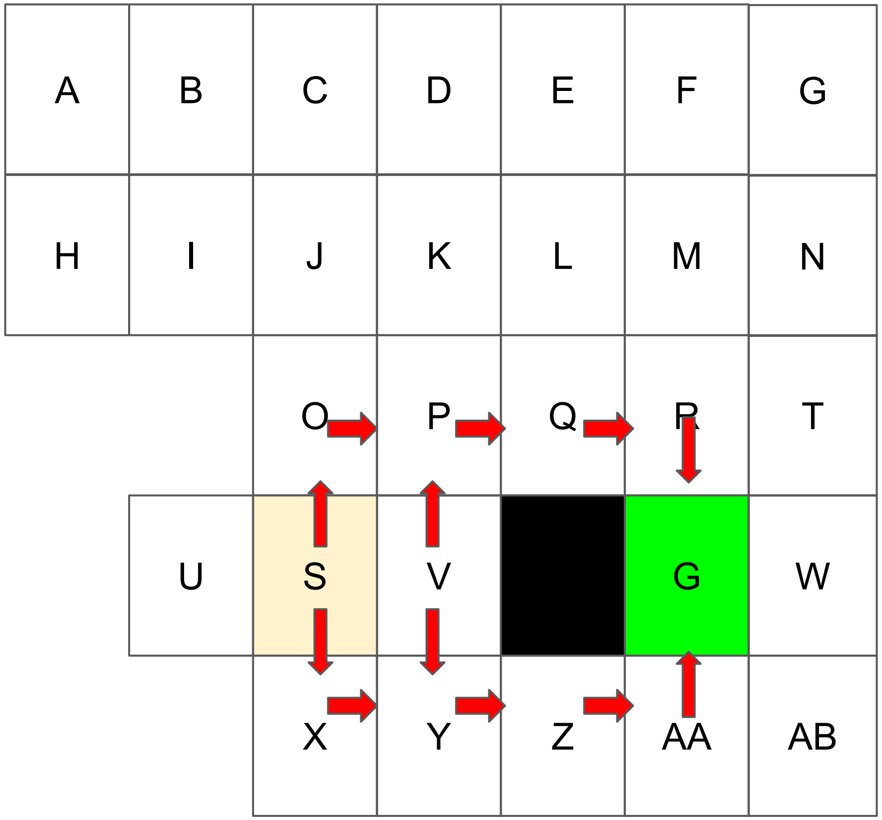
A or B?
Keep the term "discount factor γ" in mind

Policy/reward illustration

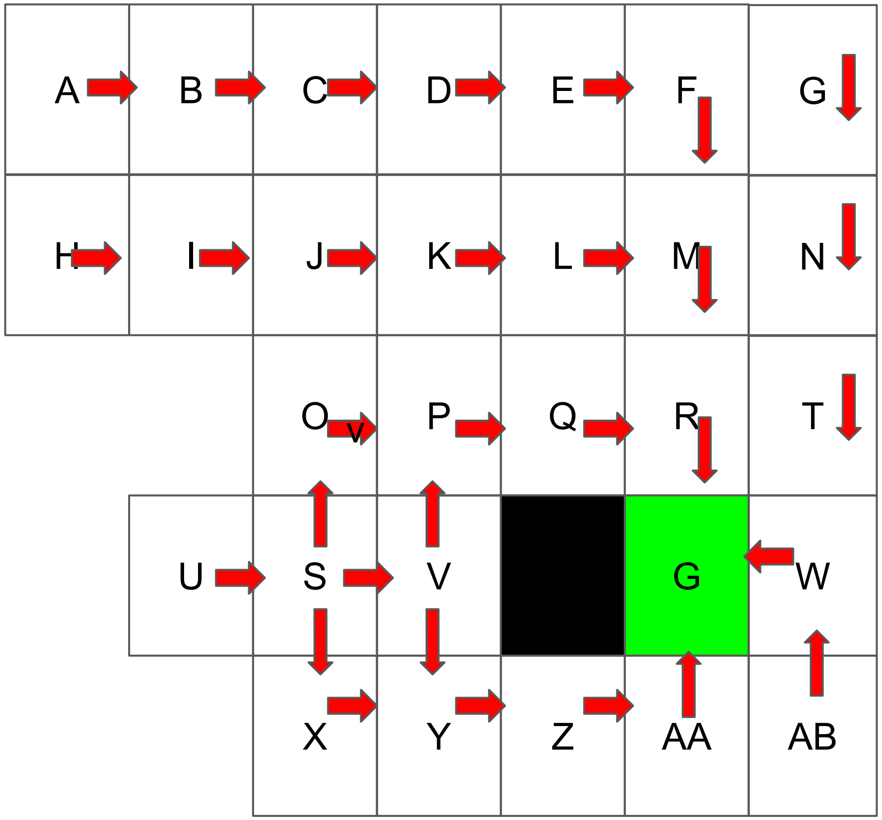


- By going to G, agent receives +100 reward.
- By visiting any other state, agent receives -1.
- Let’s assume there is no noise in the action
- Black: obstacle

Policy/reward illustration

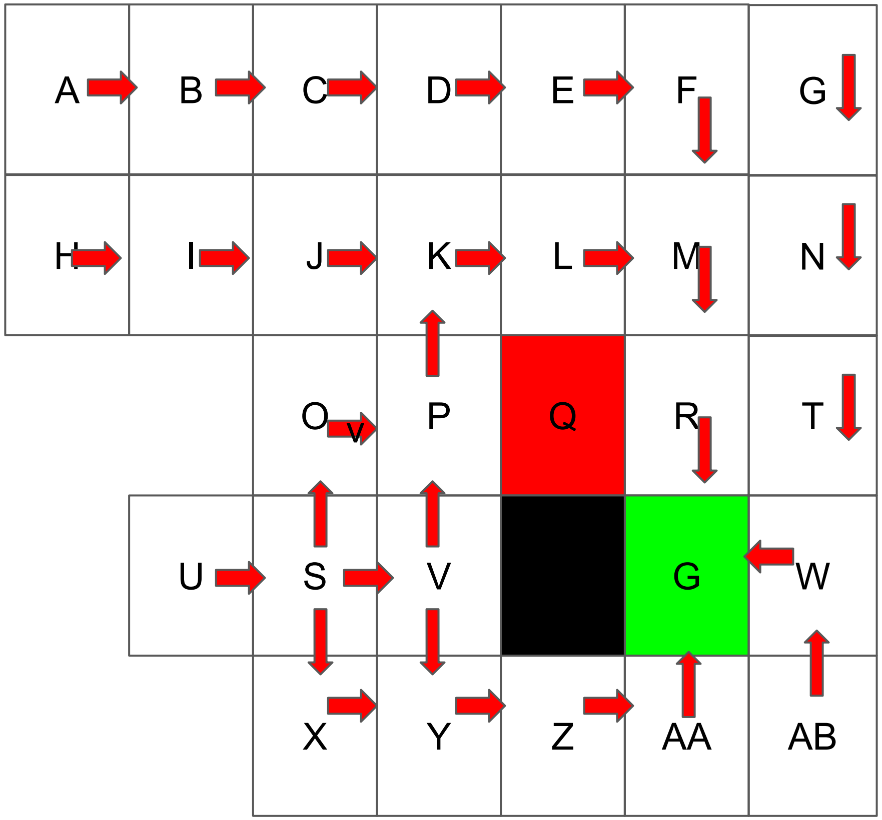


Policy/reward illustration

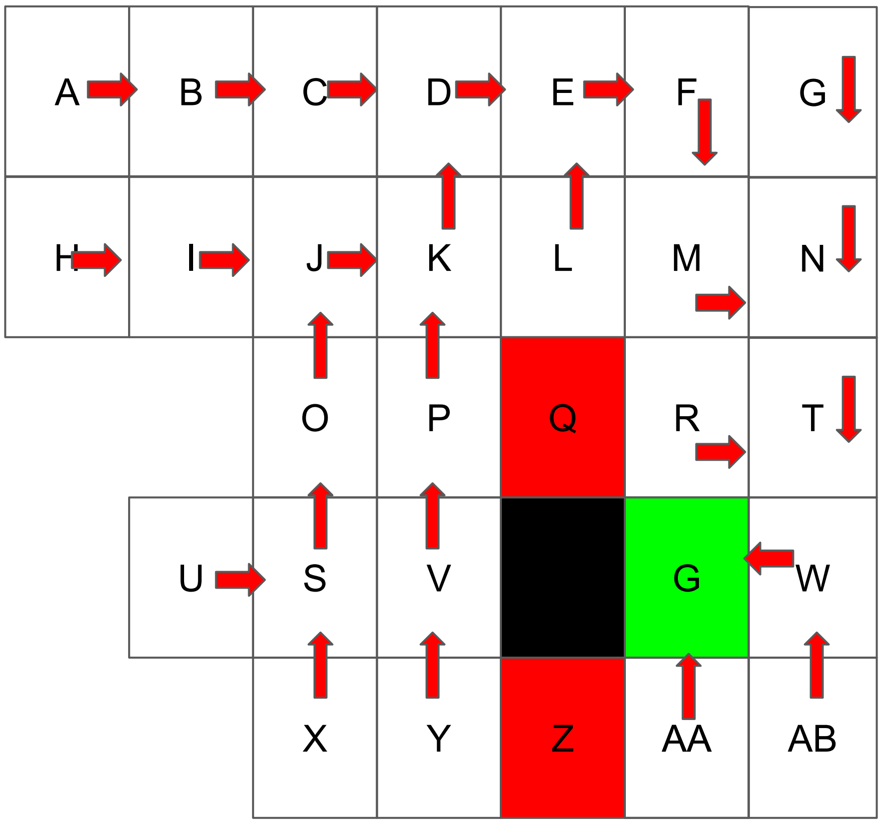


Policy/reward illustration

Red cells have a reward of -100



Policy/reward illustration



Red cells have a reward of -100

Actions are highly stochastic

Markov Decision Process

The underlying model for RL problems:

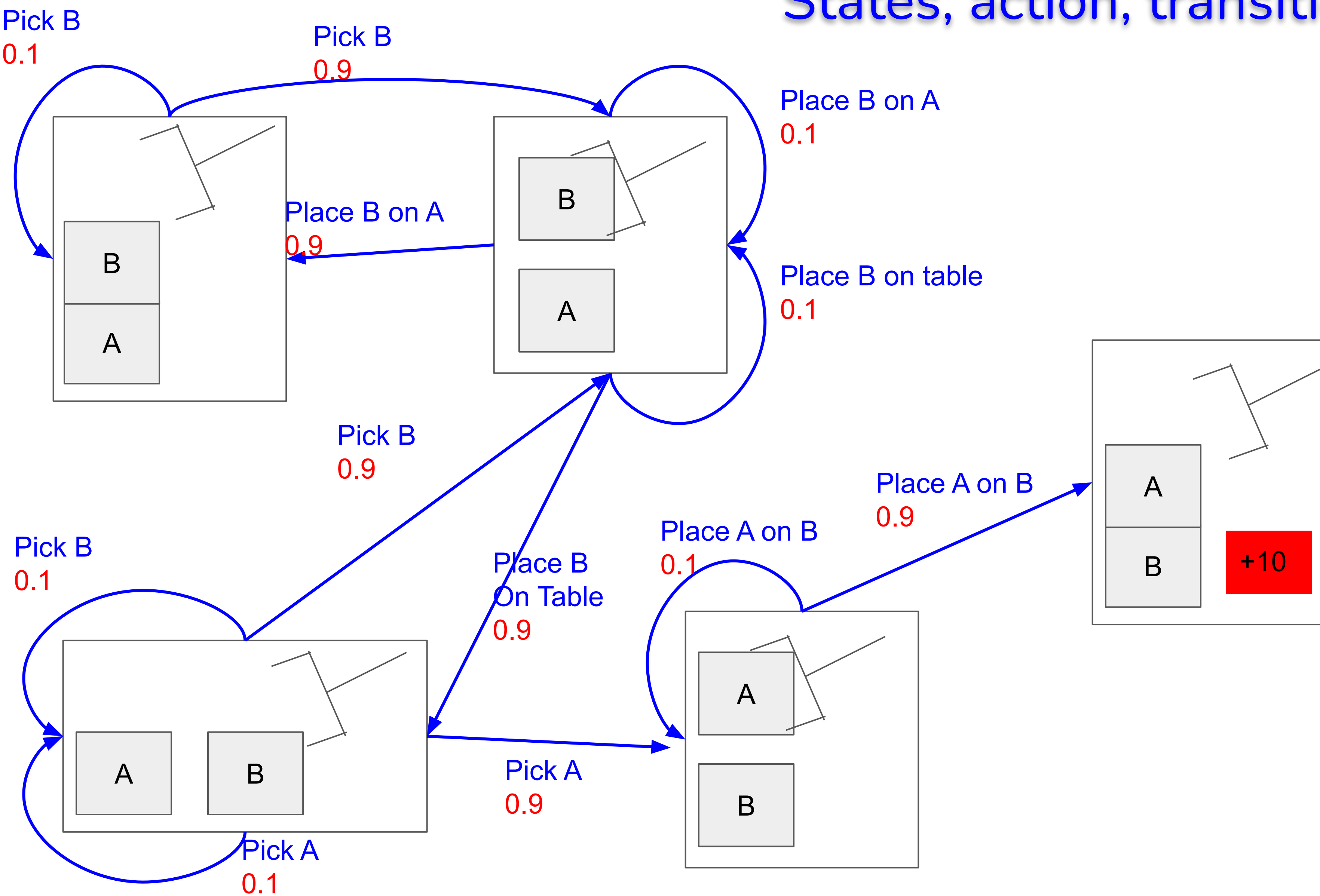
Markov decision processes:

- Set of states S
- Set of actions A
- Transitions P(s'|s,a) (or T(s,a,s'))
- Rewards R(s,a)
- discount γ

Revisiting the stacking problem

(with a different approach)

States, action, transitions



Markov assumption

Each state only relies to the previous state

Important note

In a lot of problems, transition function is not known.

Policy, values are also unknown is also unknown in a lot of problems, that's what the RL agent tries to learn

Utility (value)

State value V(s): The overall expected value that the agent can collect from state s

V(s) is usually defined recursively:

The utility of a state equals its own reward plus the expected utility of its successor

$V(s) = R(s) + \text{gamma} * \text{sum}\ (P(s'|s,a) * V(s'))$

Extra material

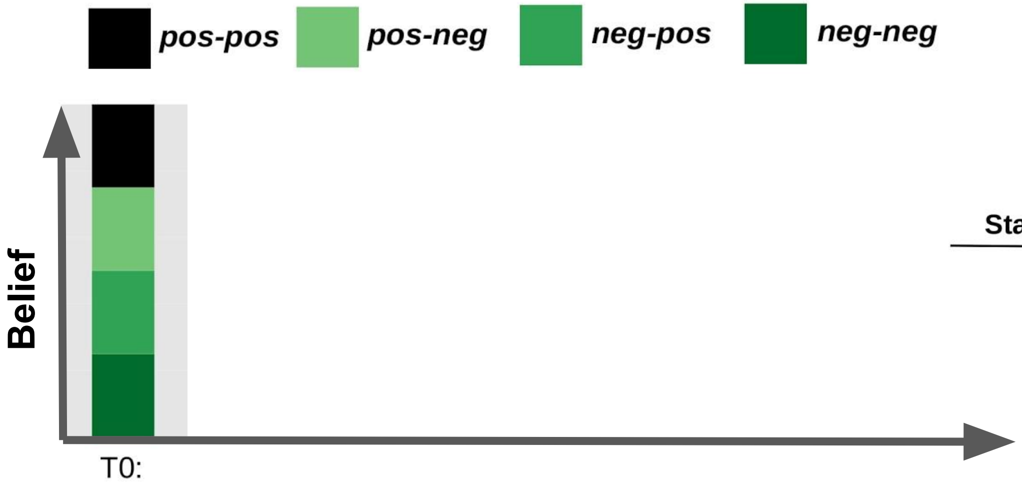
Partially Observable MDP

Object selected:

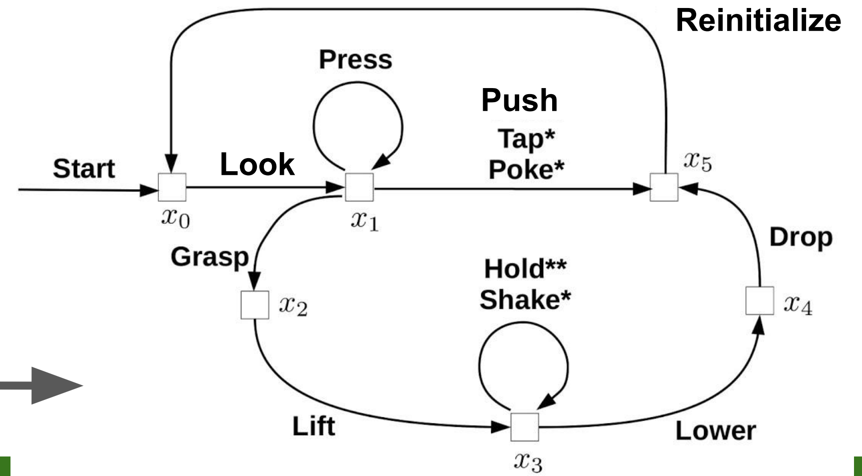
- A red and blue bottle full of water

Query:

- Yellow and metallic?



Timestep



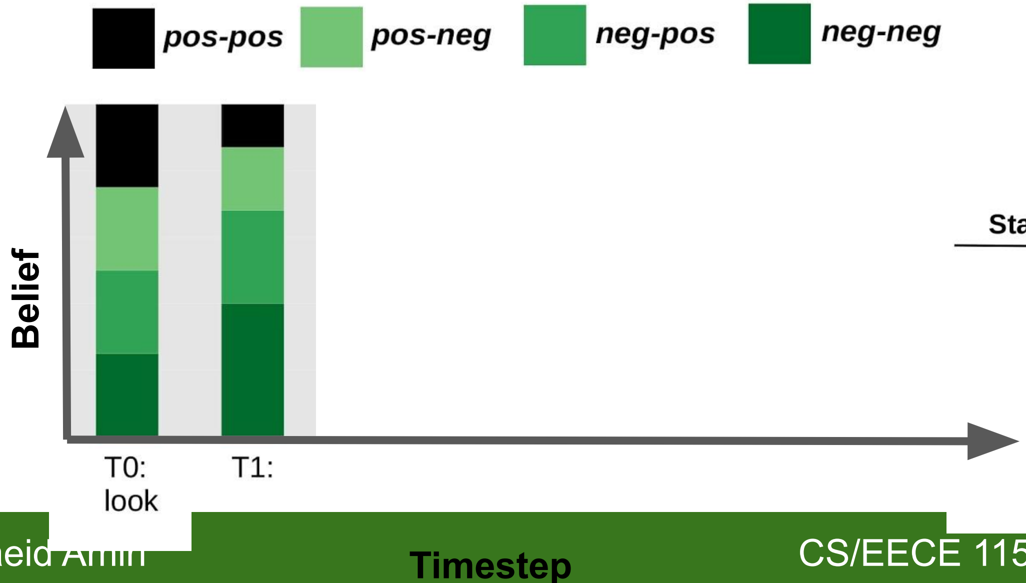
Partially Observable MDP

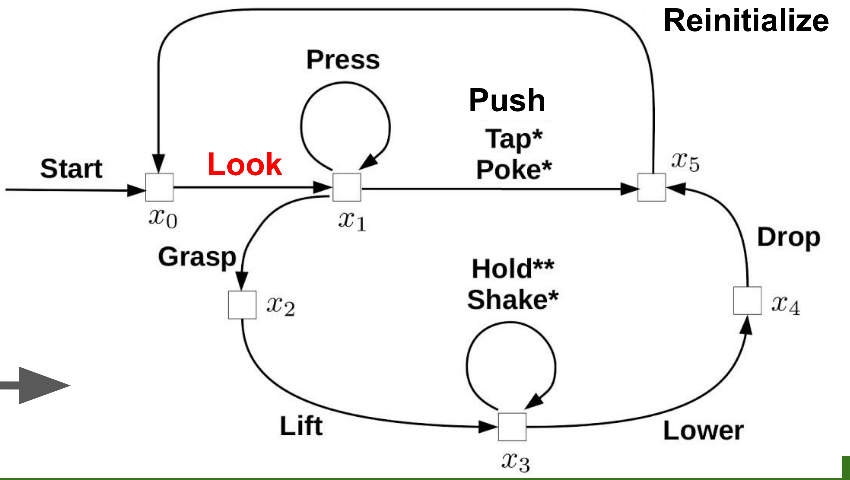
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





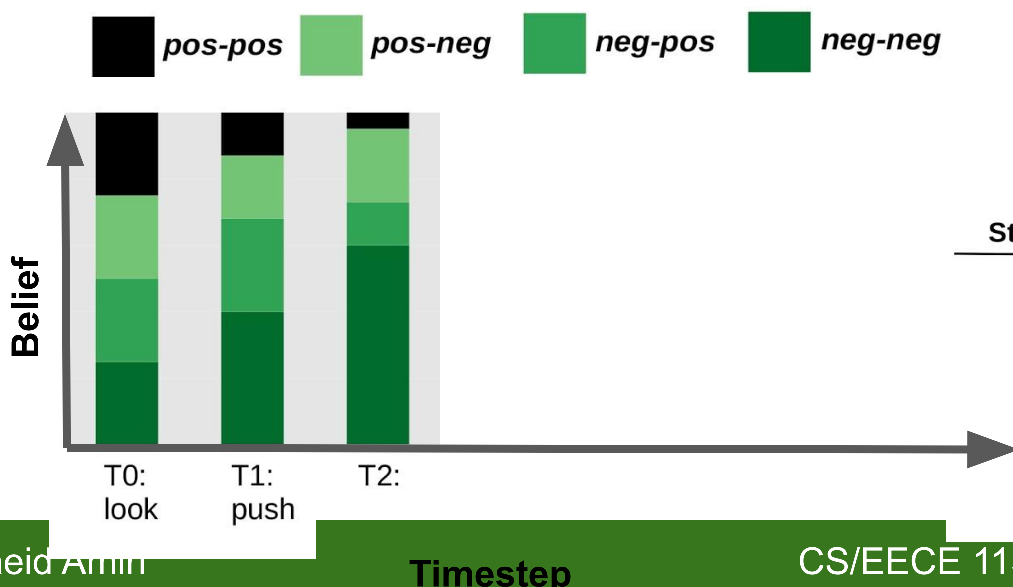
Partially Observable MDP

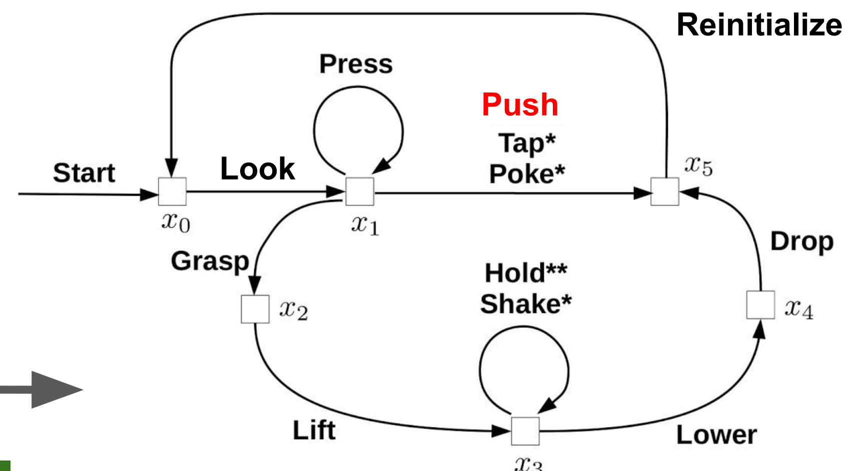
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





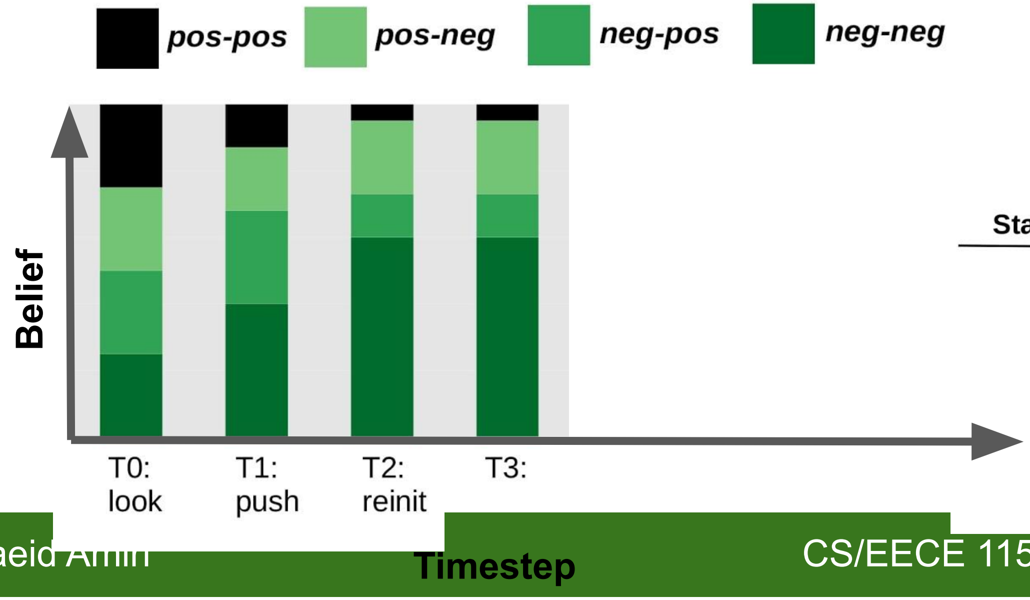
Partially Observable MDP

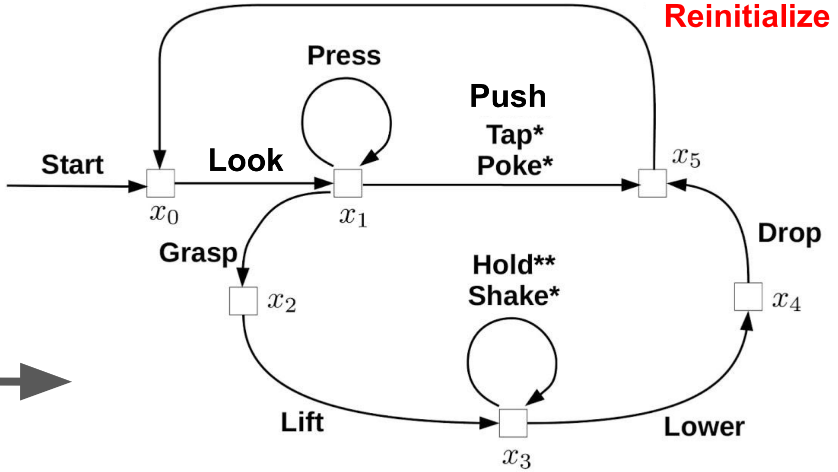
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





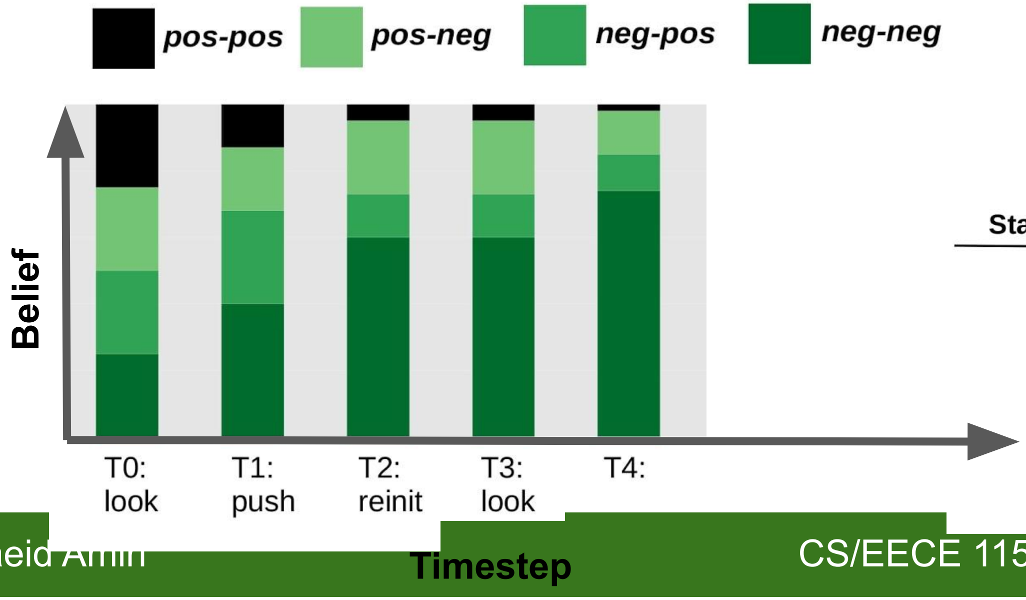
Partially Observable MDP

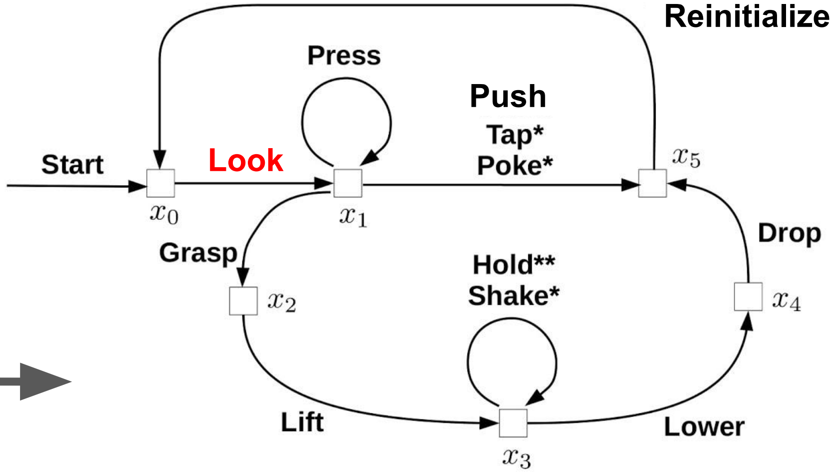
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





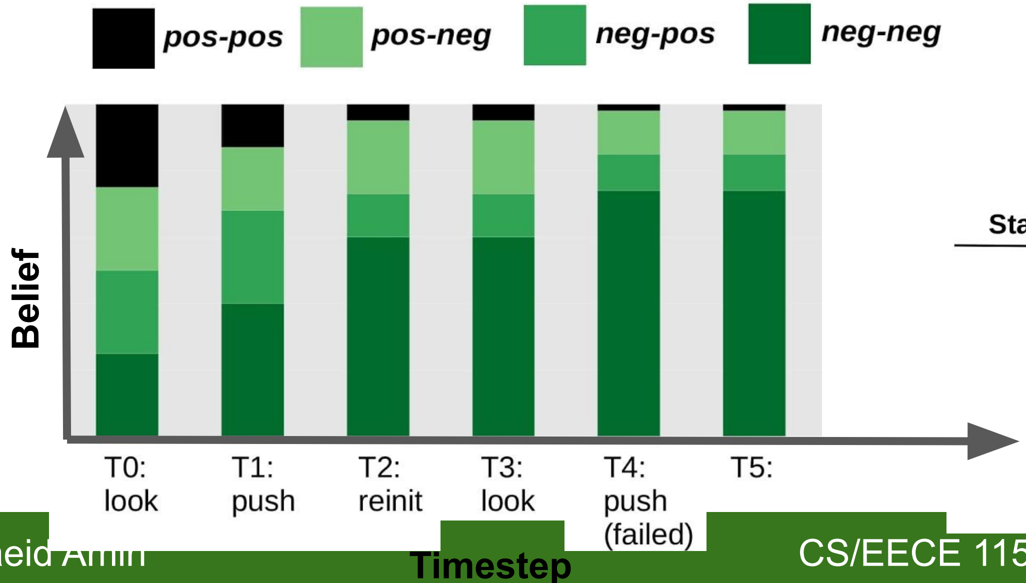
Partially Observable MDP

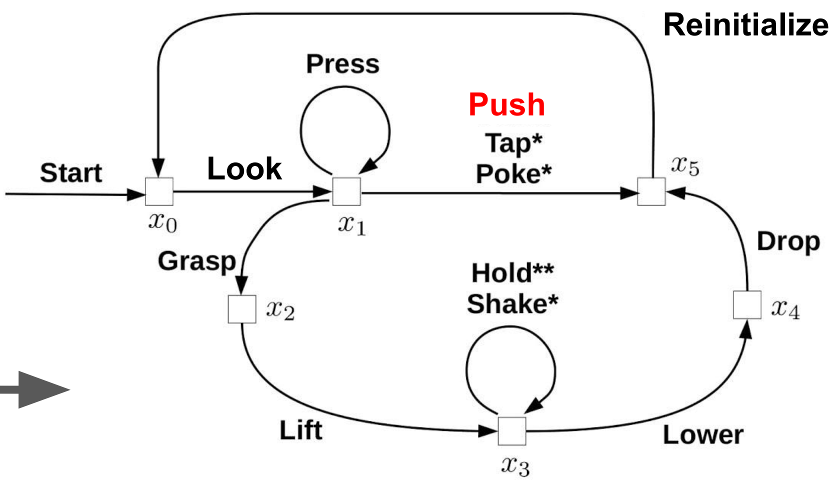
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





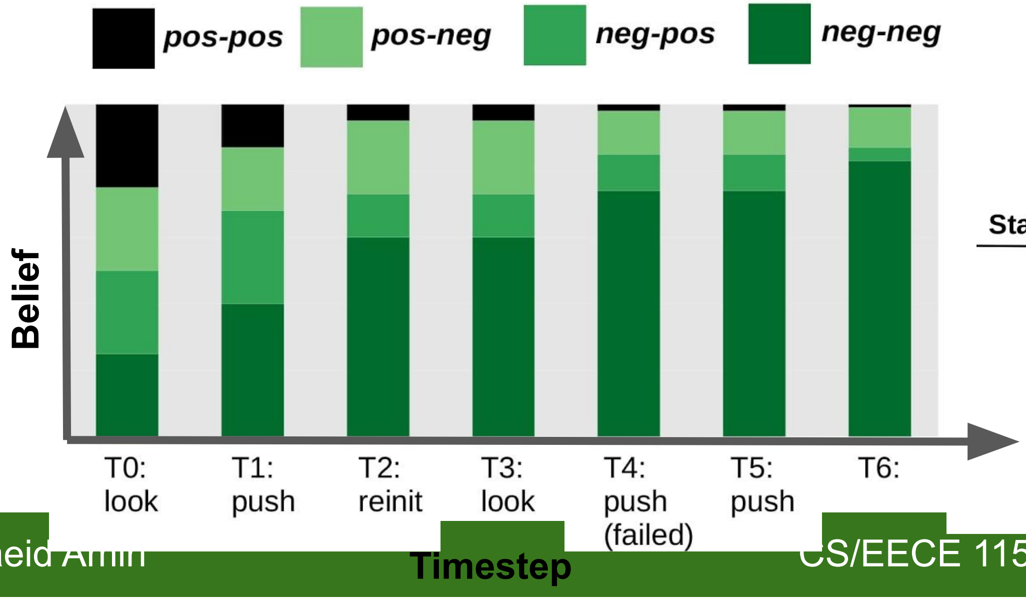
Partially Observable MDP

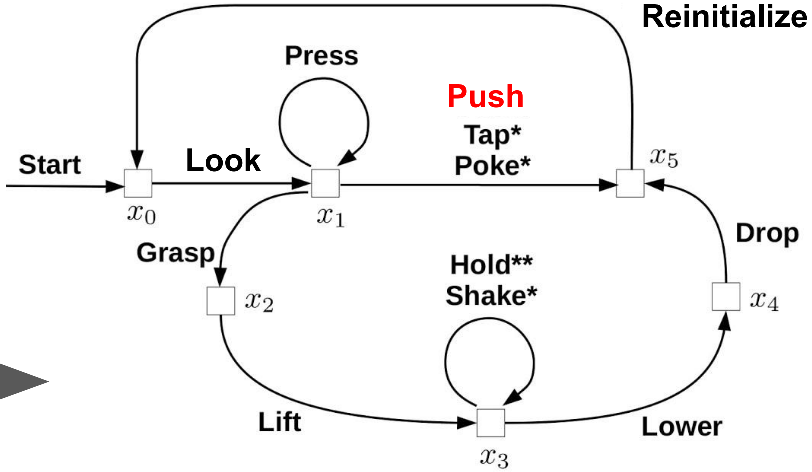
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





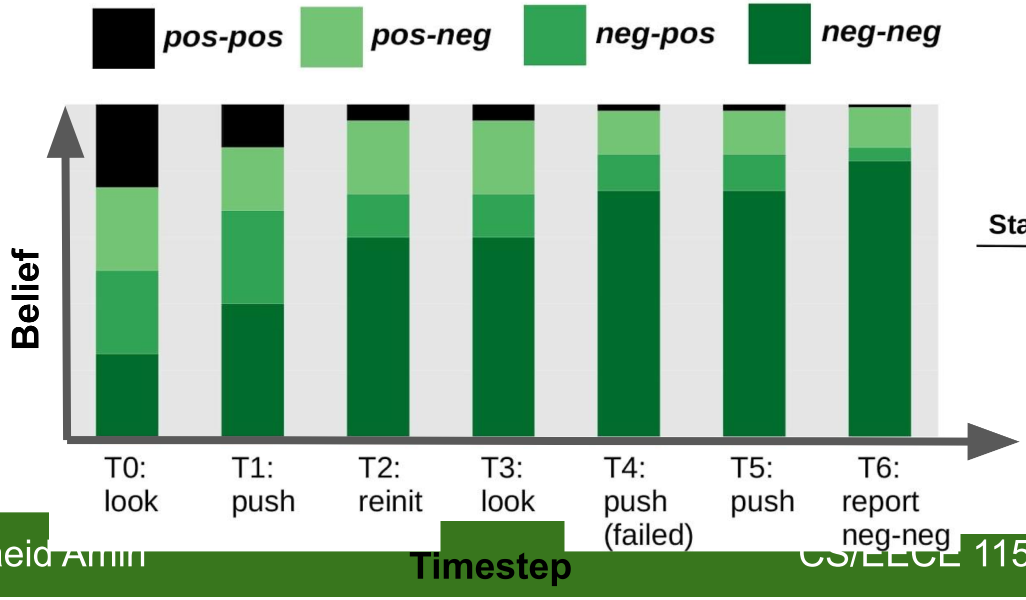
Partially Observable MDP

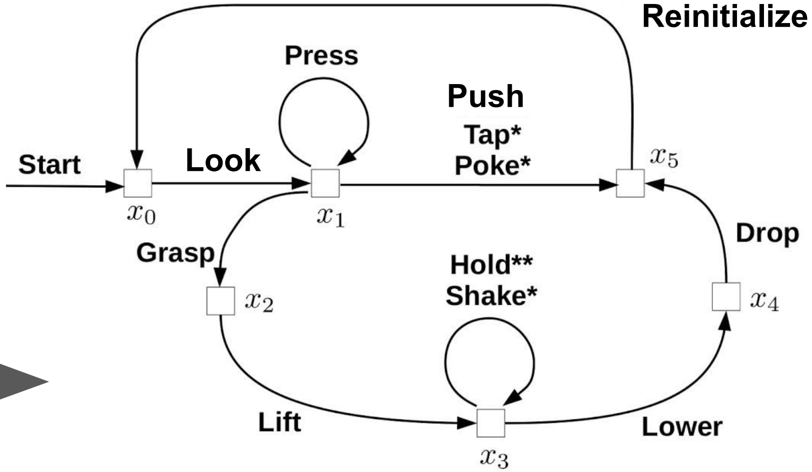
Object selected:

- A red and blue bottle full of water

Query:

- Yellow and metallic?





References

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

Nate Kohl and Peter Stone. Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion. In Proceedings of the IEEE International Conference on Robotics and Automation, May 2004.