Lab

04/27/2021

Agenda

Stacking problem:

- Classical Planning (one demo, one problem)
- Probabilistic Planning (one demo)

Q-learning (RL)

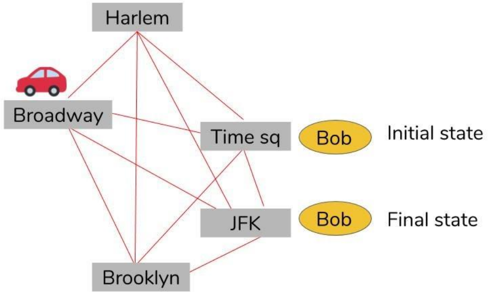
- One problem

Robotics:

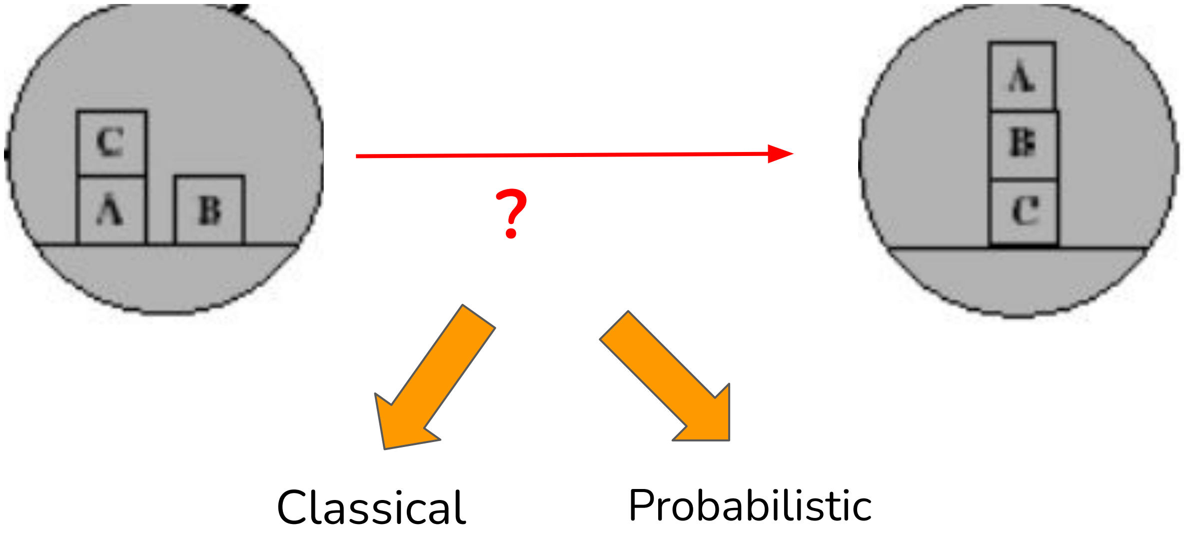
- Mapping
- Localization

What we discussed previously

We used Answer Set Programming (ASP) to solve these problems.



Stacking problem

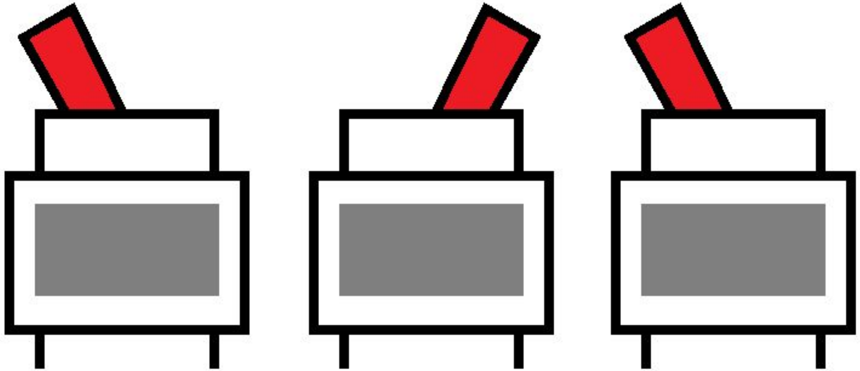


Stacking problem

(classical planning)

Switch problem: Classical planning

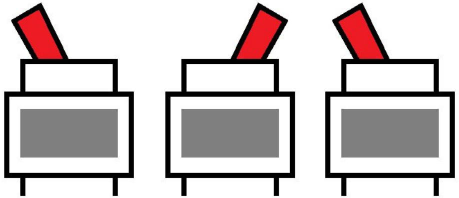
Before, solving the stacking problem, let's look at a demonstration example



Switch problem: Classical planning

The main components of a planning problem are:

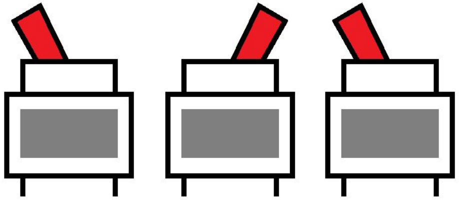
- A set of things that make up the world (the objects)
- The properties that are used to describe the state of those things (the predicates)
- A description of how the world behaves and the capabilities of the agent (the actions)
- A description of the initial situation (the initial state)
- A description of the desired situation (the goal)



Switch problem: Classical planning

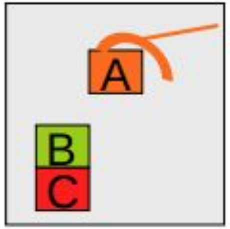
Click on the link below:

http://editor.planning.domains/#read_session=jfespcjFc3



Recap: Representing states

World states are represented as sets of facts. We will also refer to facts as propositions.

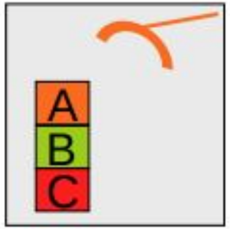


holding(A)
clear(B)
on(B,C)
onTable(C)

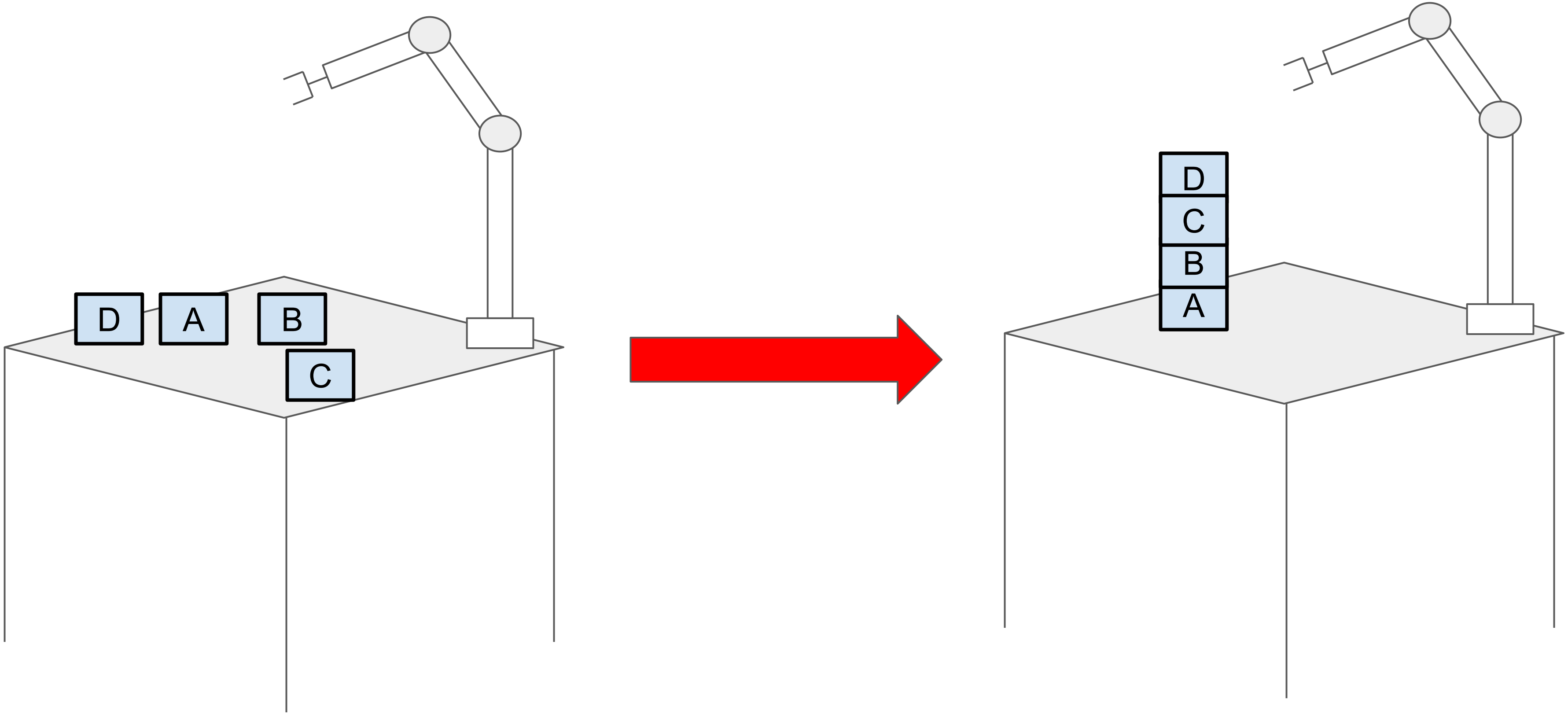
State 1

handEmpty
clear(A)
on(A,B)
on(B,C)
onTable(C)

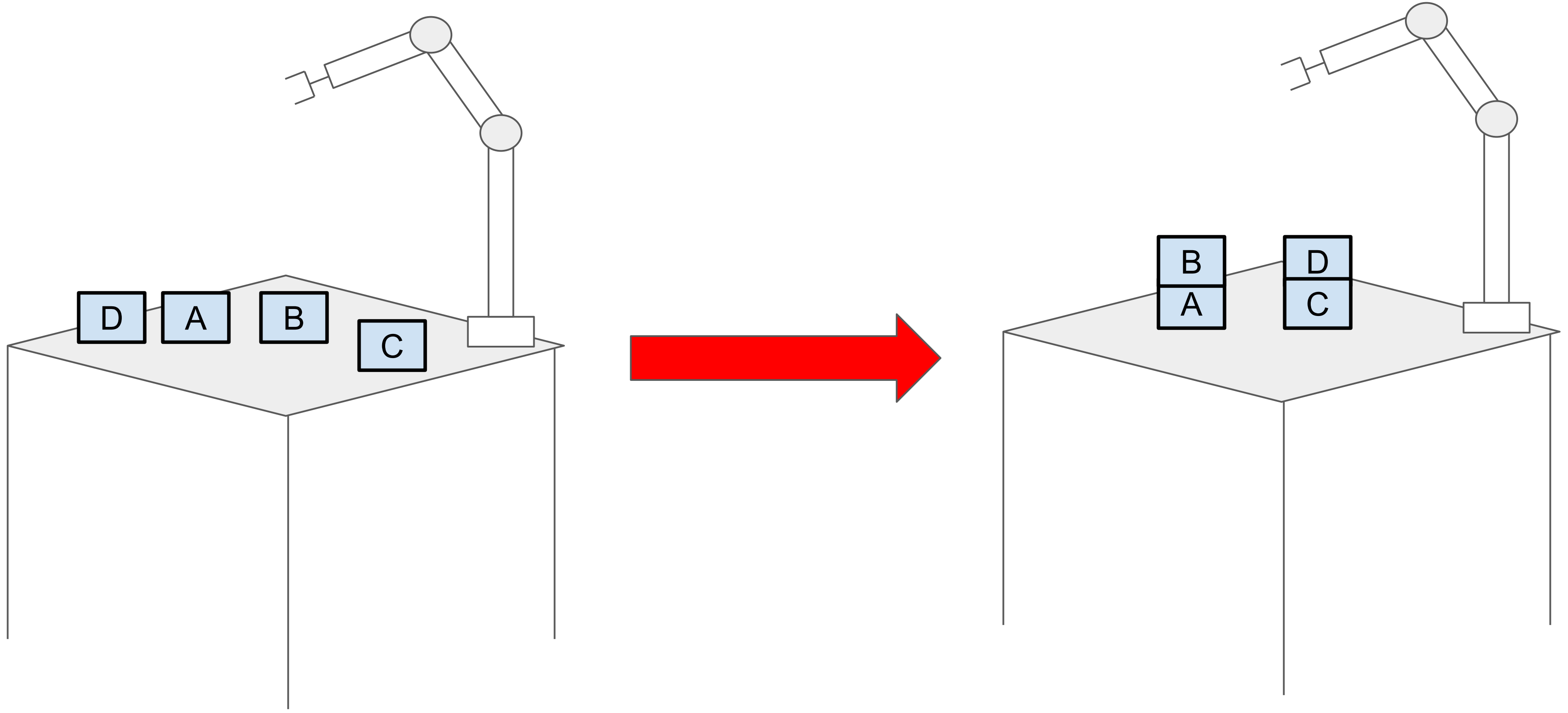
State 2



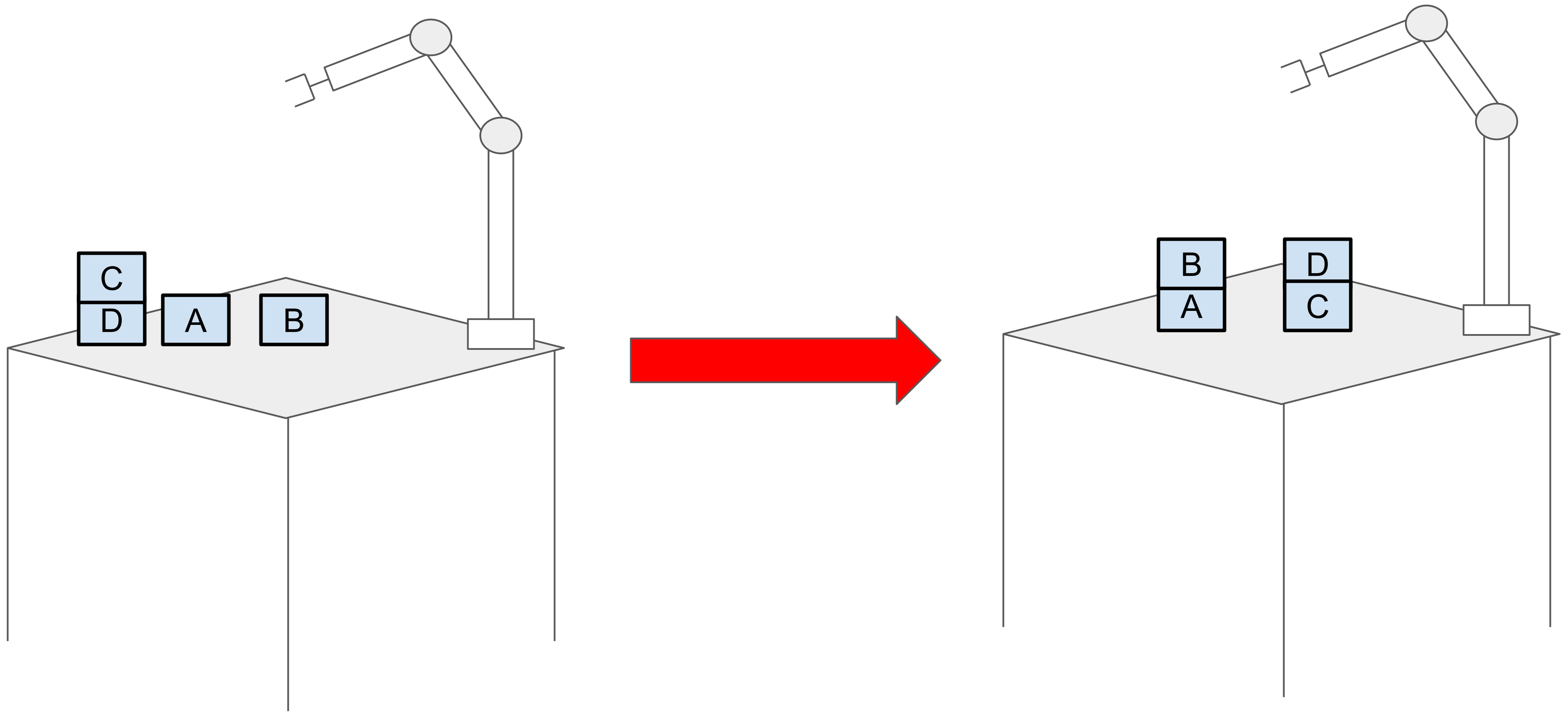
Stacking problem (1): Classical planning



Stacking problem (2): Classical planning



Stacking problem (3): Classical planning



Stacking problem: Classical planning

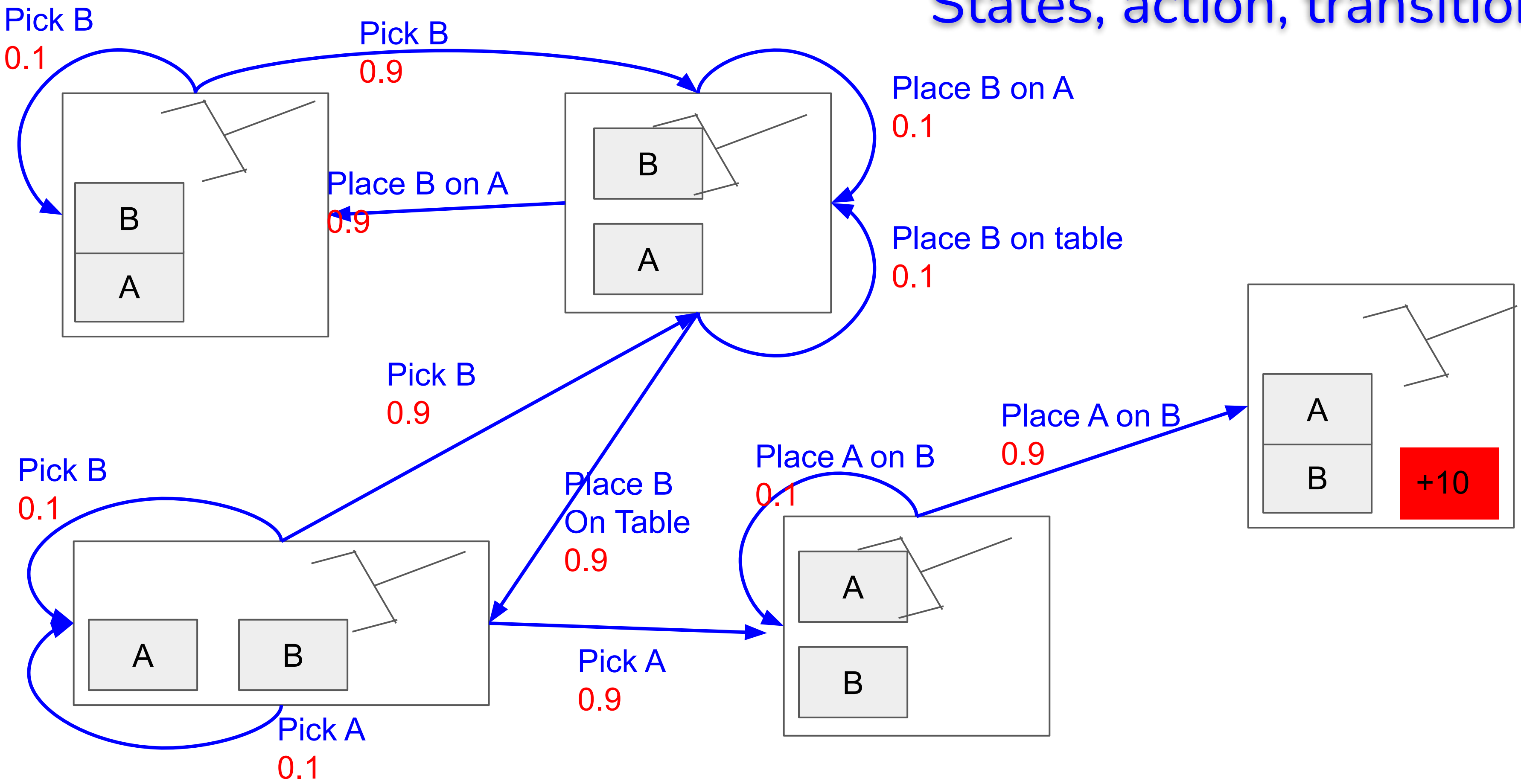
Solver link: http://editor.planning.domains/

Incomplete code link:
https://github.com/FRI-IASA/teaching/tree/master/week12/classical_planning

Stacking problem

(Probabilistic planning)

States, action, transitions



Stacking problem: Probabilistic planning

Demo code link:

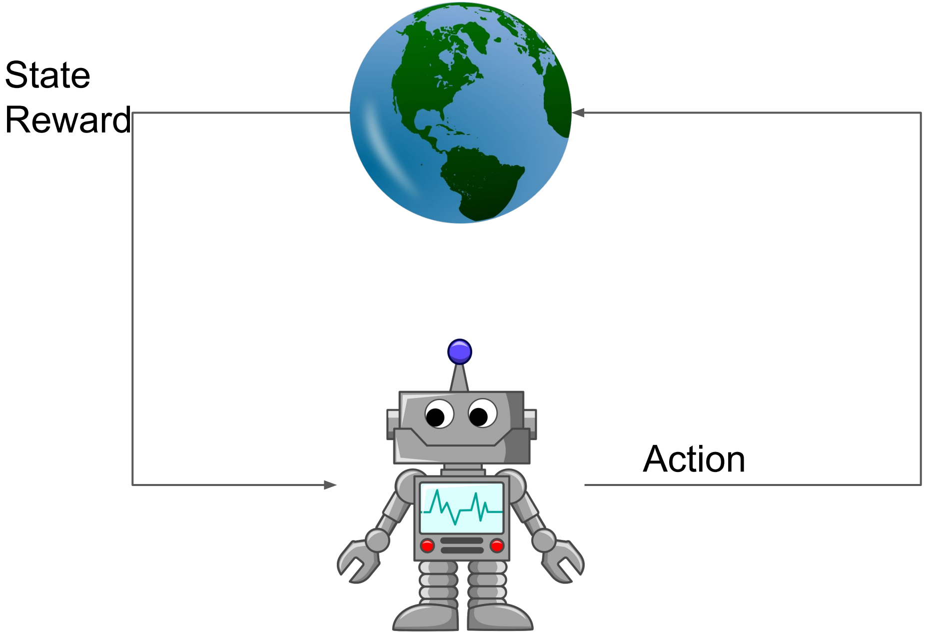
https://github.com/FRI-IASA/teaching/tree/master/week12/value_iteration

Reinforcement Learning (q-learning)

Recap: Reinforcement Learning

A computational approach to learning from interaction

Maximize
long-term
reward



Multi-armed bandit

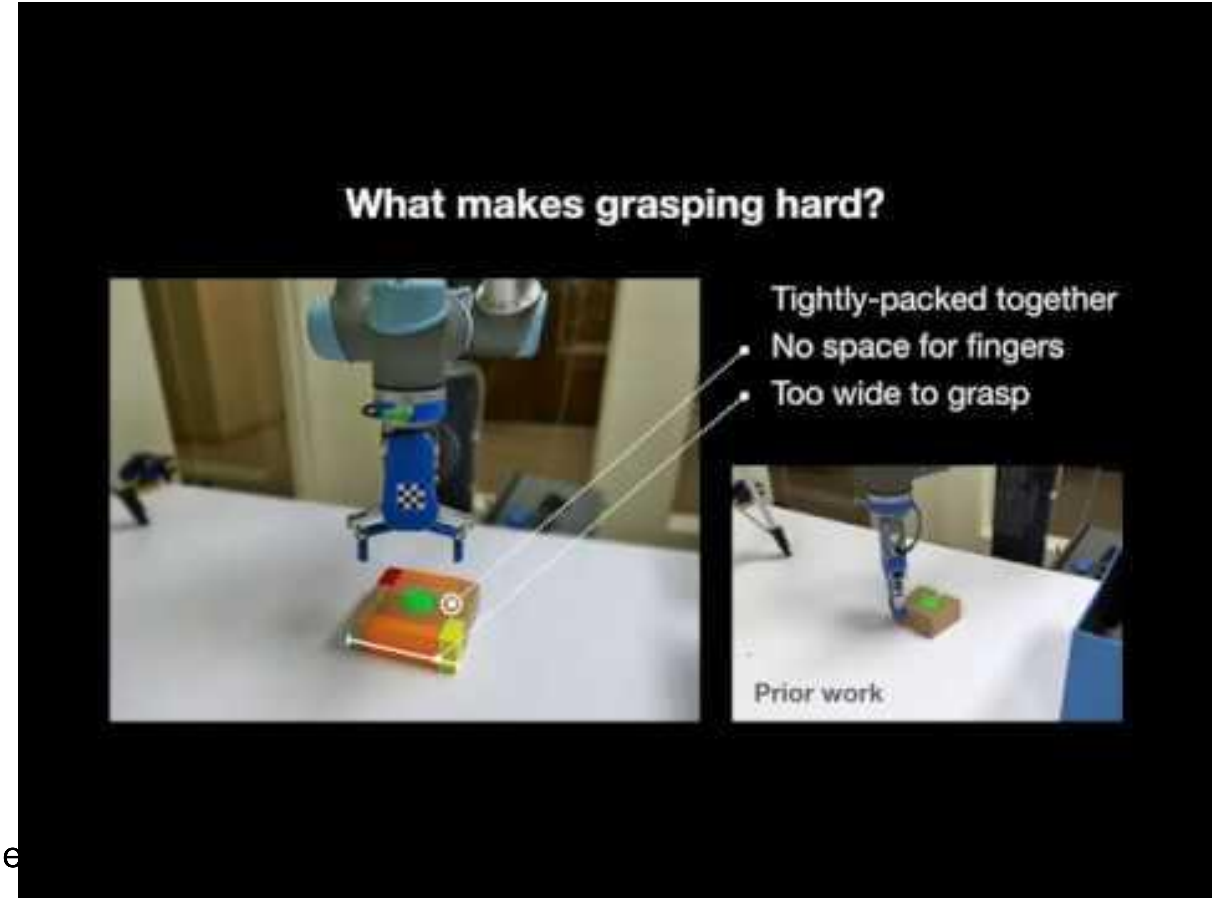
Consider the following learning problem. You are faced repeatedly with a choice among **k** different options, or actions.

After each choice you receive a numerical reward.

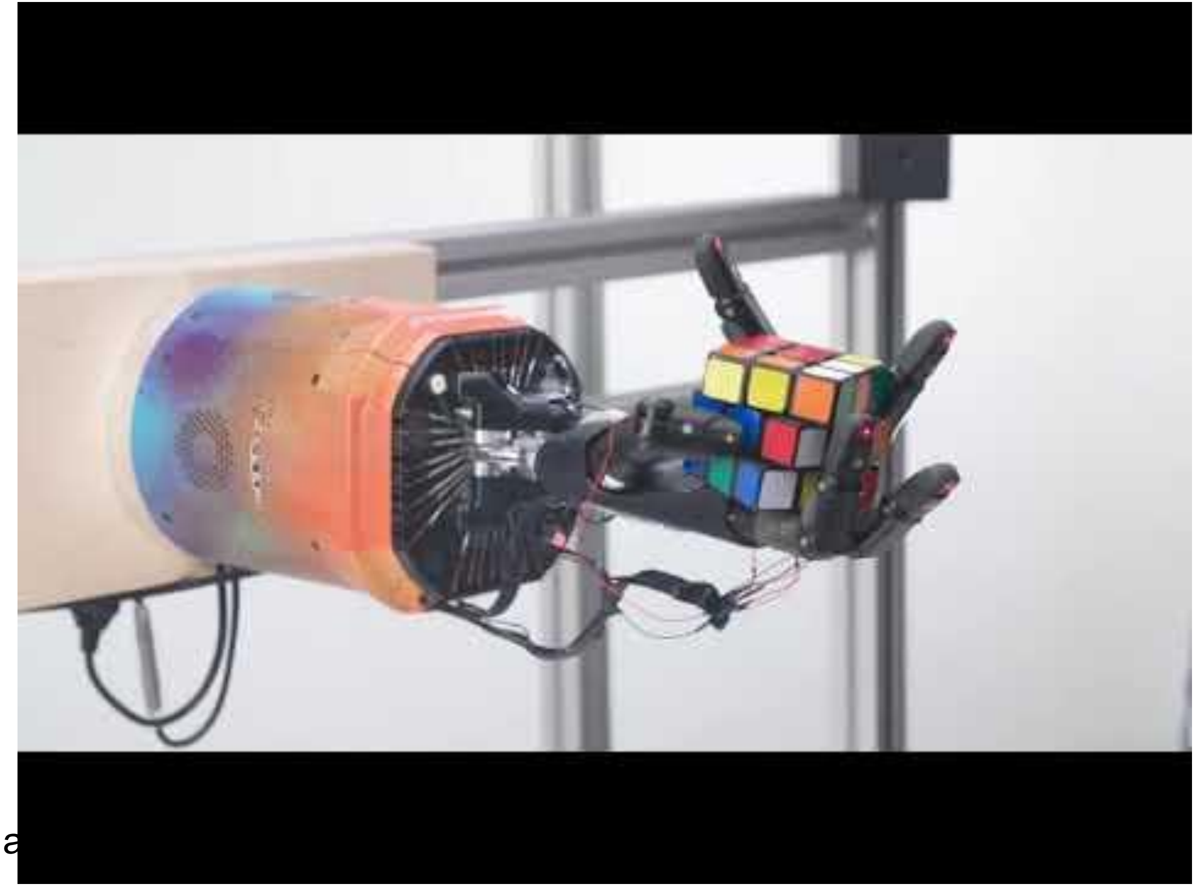
Objective is to maximize the expected total reward over some time period:

Demo: http://iosband.github.io/2015/07/28/Beat-the-bandit.html

[Mnih et


What makes grasping hard?
Tightly-packed together
No space for fingers
Too wide to grasp
Prior work

[Zeng e



[Akkaya et a

Recap: Important concepts

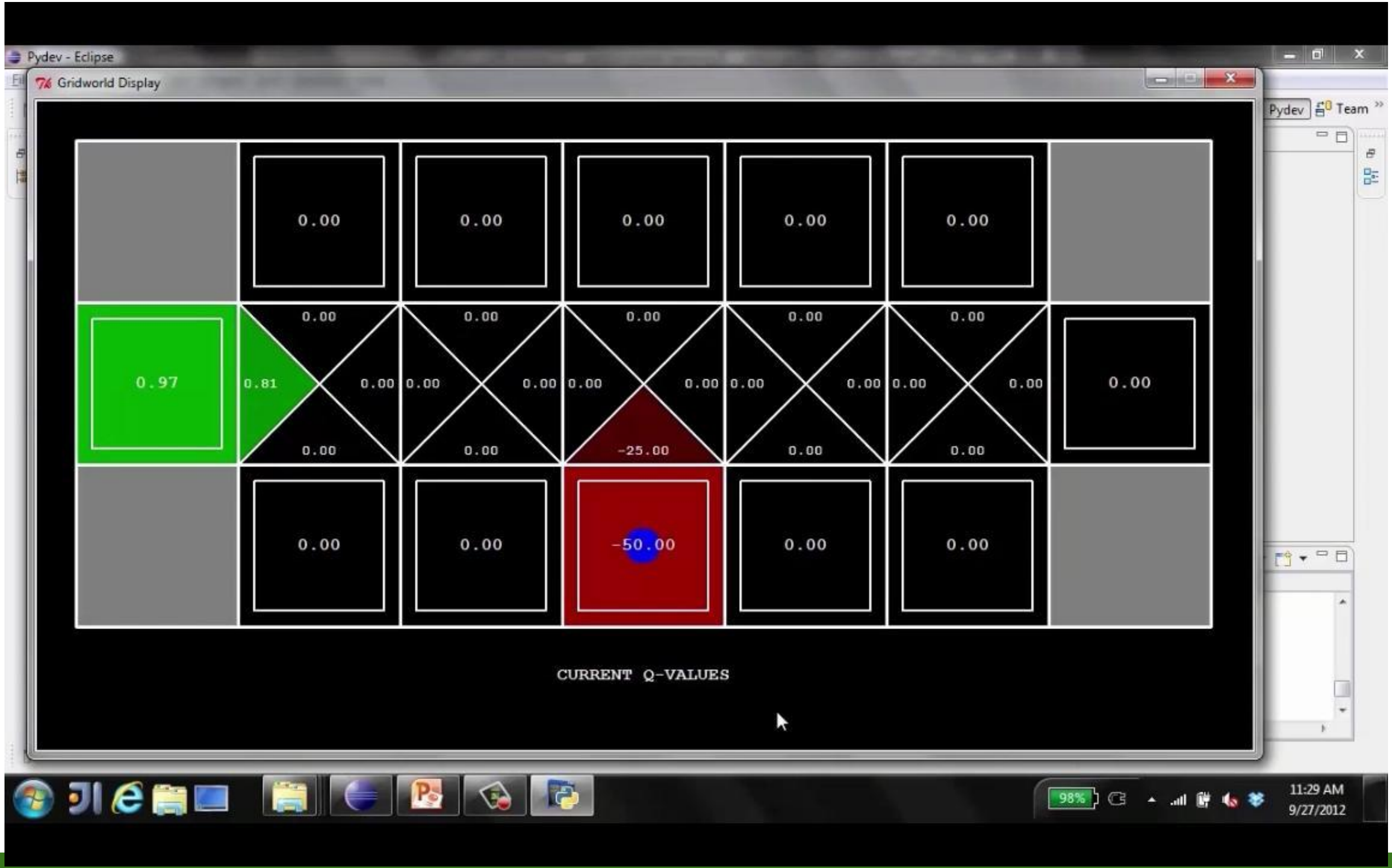
Value V(s): Total amount of reward an agent can expect to accumulate over the future, starting from state **s**.

Q-Value Q(s,a): Total amount of reward an agent can expect to accumulate over the future, starting from state **s** and taking action **a**.

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

Policy could be optimal, or suboptimal

Demo video



Pydev - Eclipse
Gridworld Display
0.00
0.00
0.00
0.00
0.00
0.97
0.81
0.00
0.00
-25.00
-50.00
CURRENT Q-VALUES
Pydev
Team
98%
11:29 AM
9/27/2012

q-learning

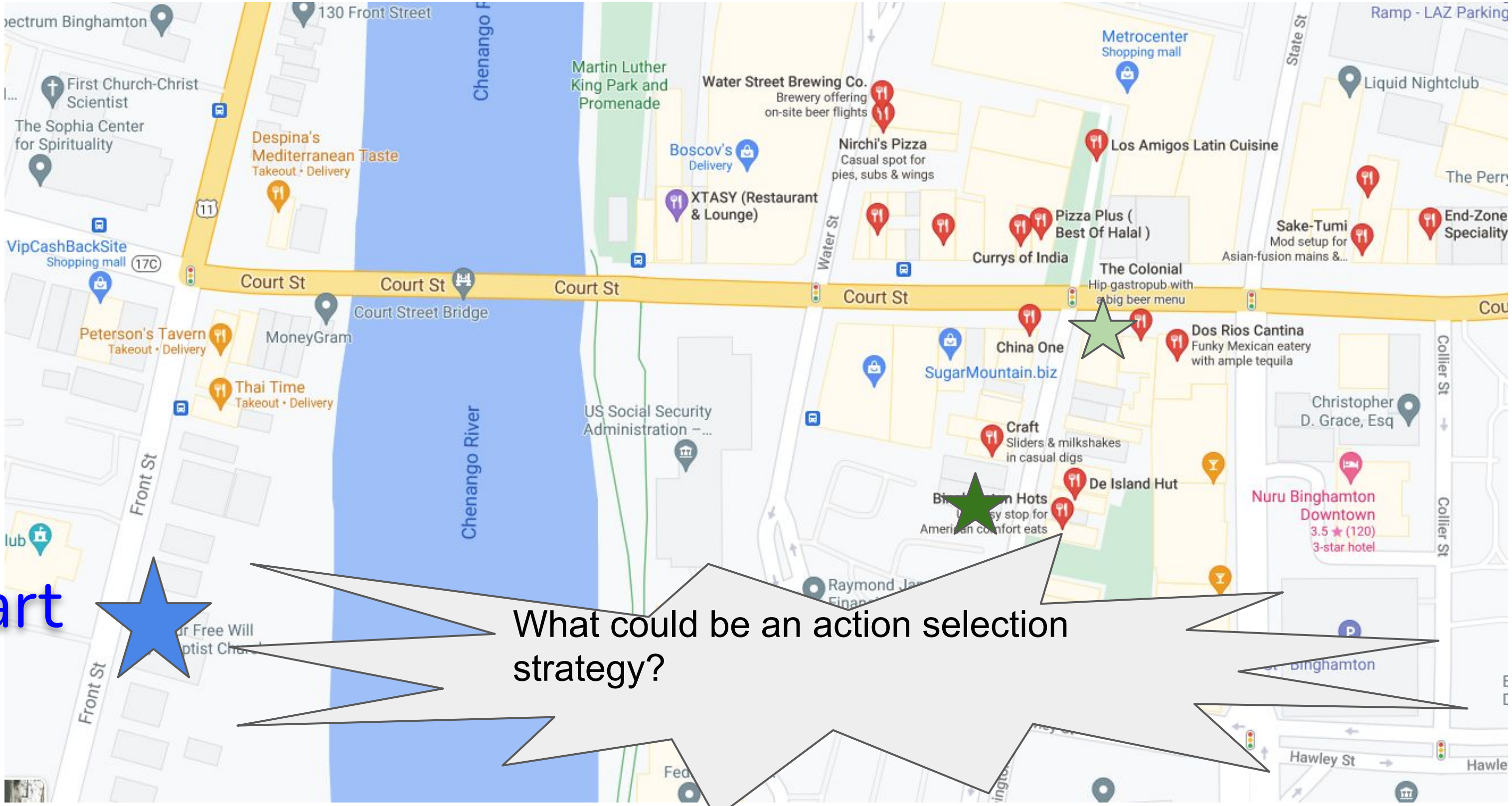
Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using policy derived from Q (e.g., ε-greedy)
        Take action A, observe R, S'
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
        $S \leftarrow S'$
    until S is terminal

Exploration vs. exploitation

Start

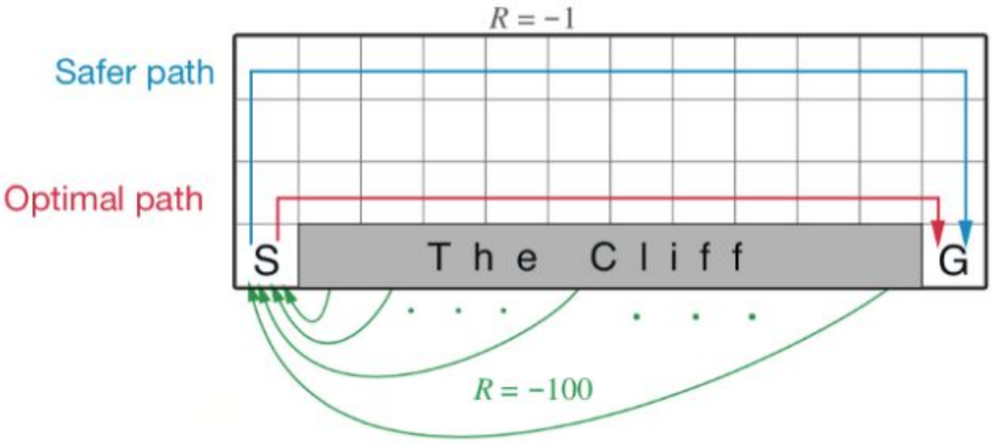


What could be an action selection strategy?

Q-learning

Cliff Environment

48 states

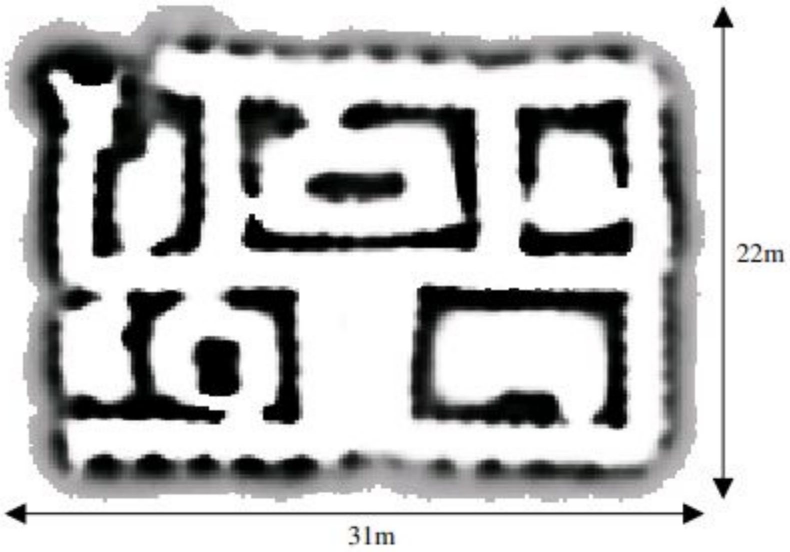


Cliff Walking

Q-learning Problem

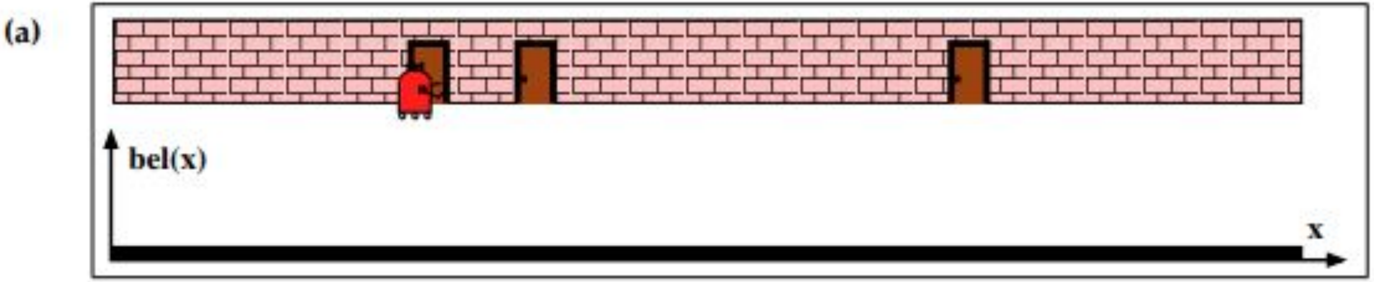
https://colab.research.google.com/drive/1u2zNNDi8-45QhlMVHRPFDKcTBVgC8EWP?usp=sharing

Mapping demo



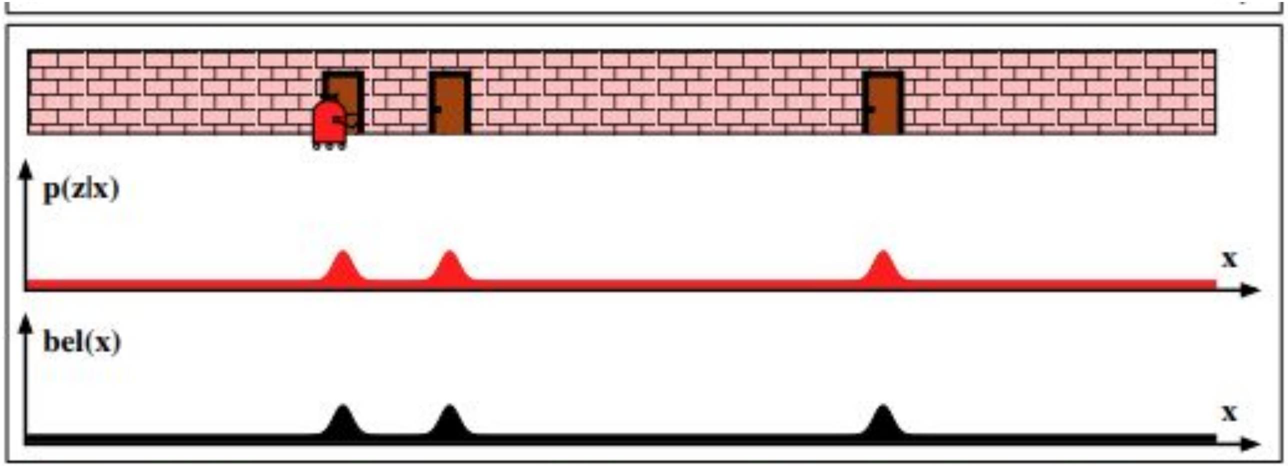
Localization Demo

Sense, Act and predict

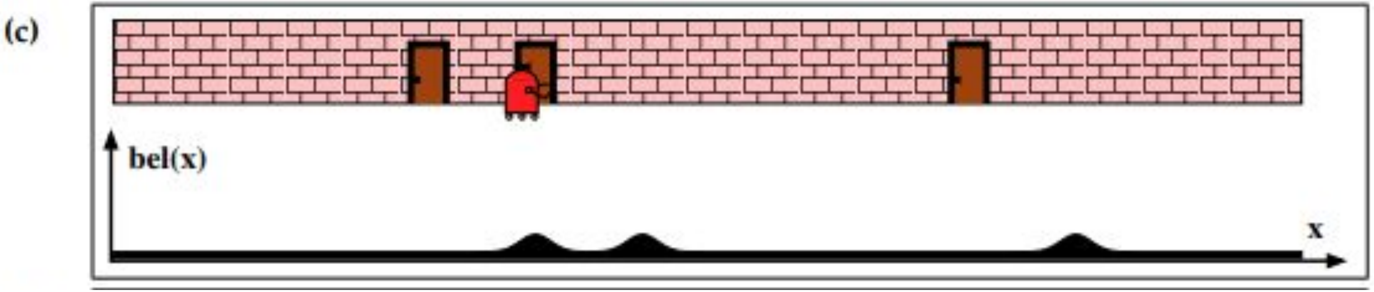


Sense, Act and predict

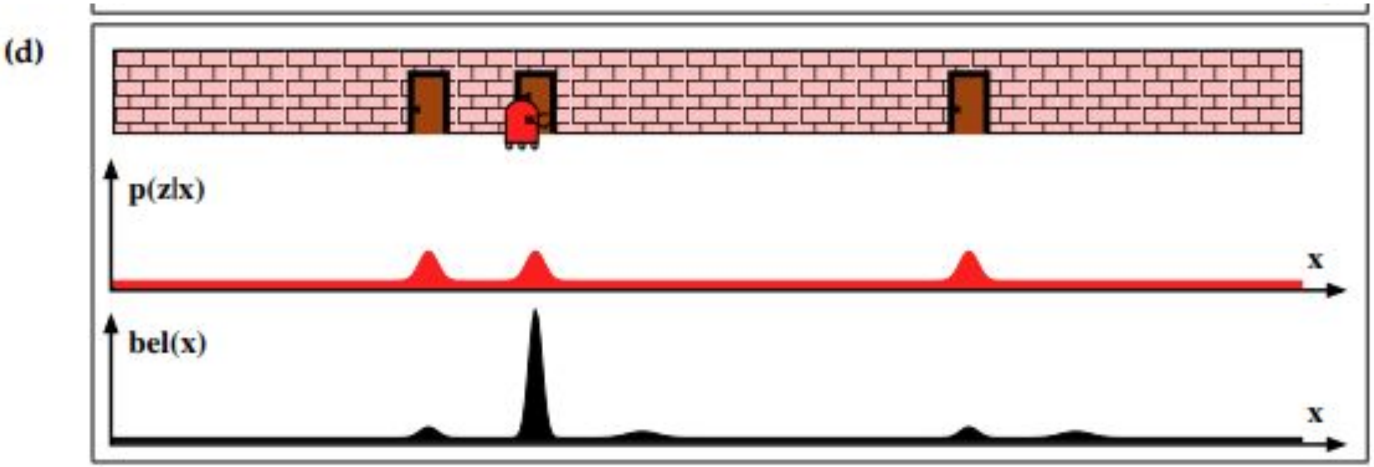
(b)



Sense, Act and predict



Sense, Act and predict



Sense, Act and predict