Markov Decision Processes

CS 580
Intro to Artificial Intelligence

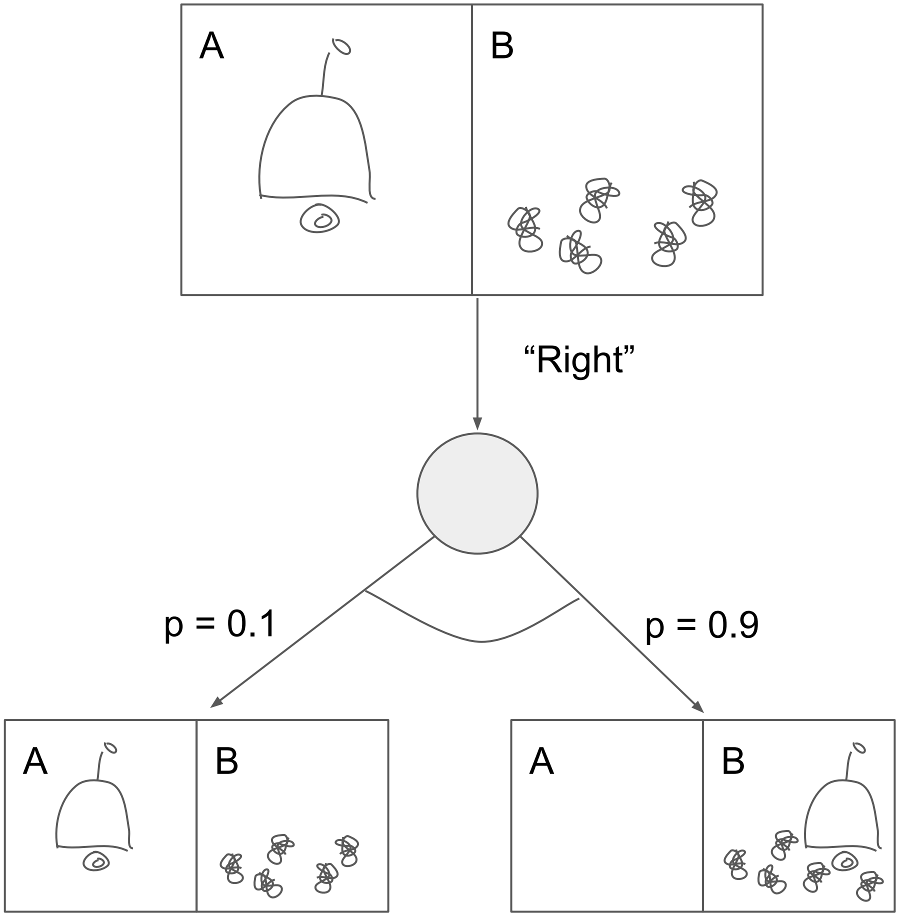
Stochastic actions

In search, we usually assumed that the problem was **deterministic**. Taking an action in a particular state always resulted in the **same** successor state.

In the real world, this usually isn't true, but we can often assign a **probability** that an action will have some result

In `And-Or` search, we developed a **contingency plan** (complex!)

Now we'll look at an alternative: **policy**



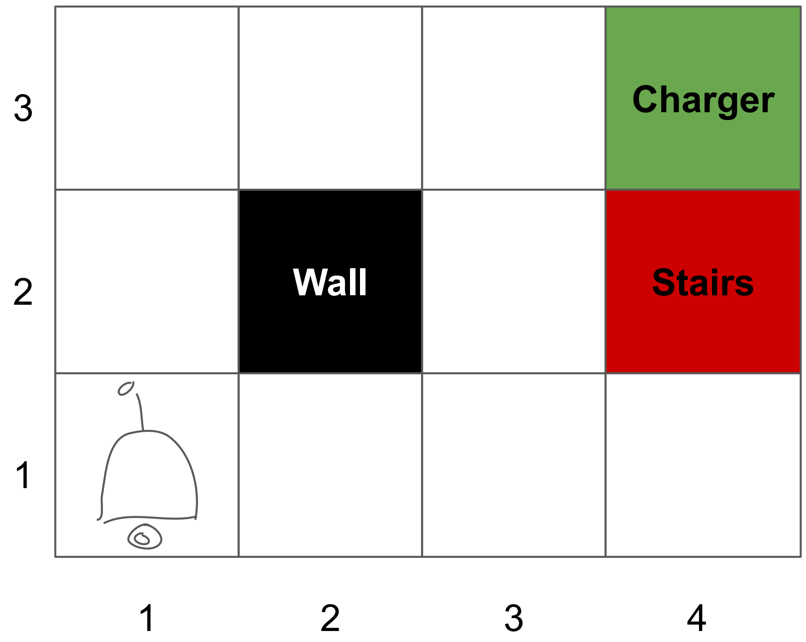
Example environment

Robot can be in any of the non-wall cells (x,y)

Actions
Up, Down, Left, Right. (moving into wall or out of bounds → stays put)

Goal
Low battery, get to the charger (4,3)! Avoid falling down the stairs (4,2)!

Non-deterministic wrinkle
Actions have a chance of moving the wrong way



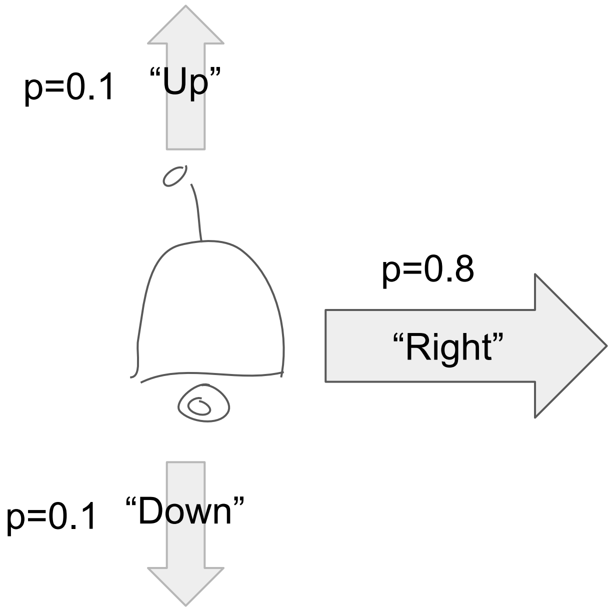
Transition Model

Let's be precise about "chance of moving the wrong way"

There is an 80% chance of moving in the intended direction. The remaining 20% is split evenly between the two orthogonal directions

Notation
Probability of transitioning from **s** to **s'** with action **a**

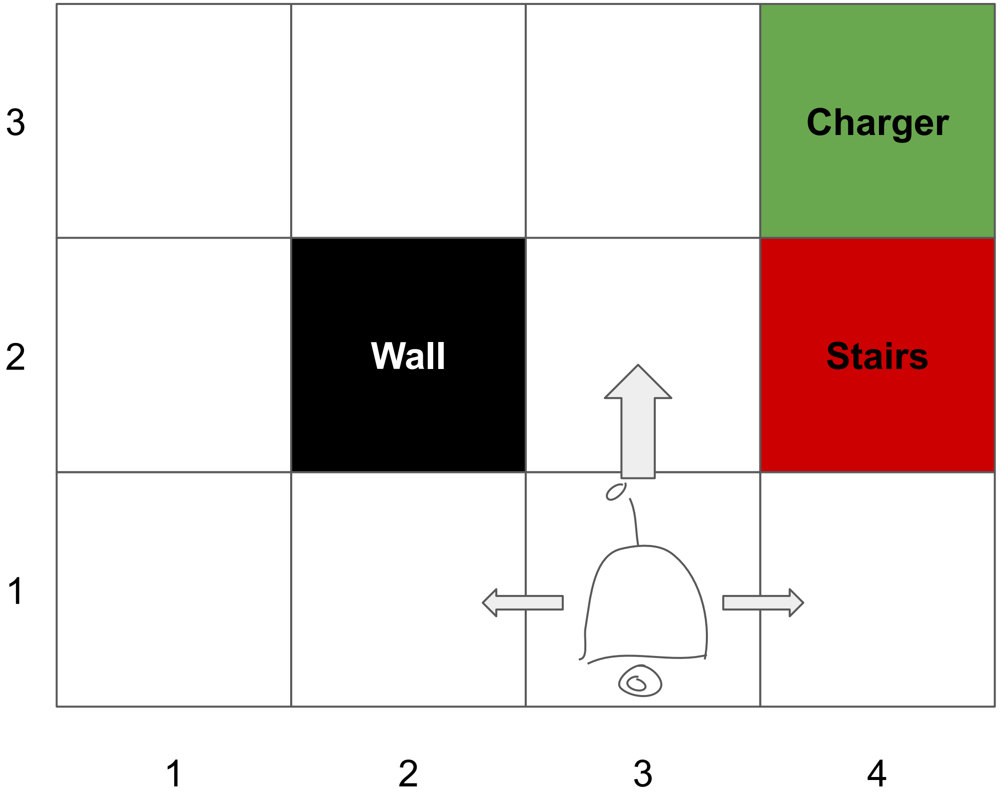
$$T(s, a, s') = p(s' \mid s, a)$$



Transition Model - Example (1)

"Up" in (3,1)→
(3,2) w/ p=0.8
(2,1) w/ p=0.1
(4,1) w/ p=0.1

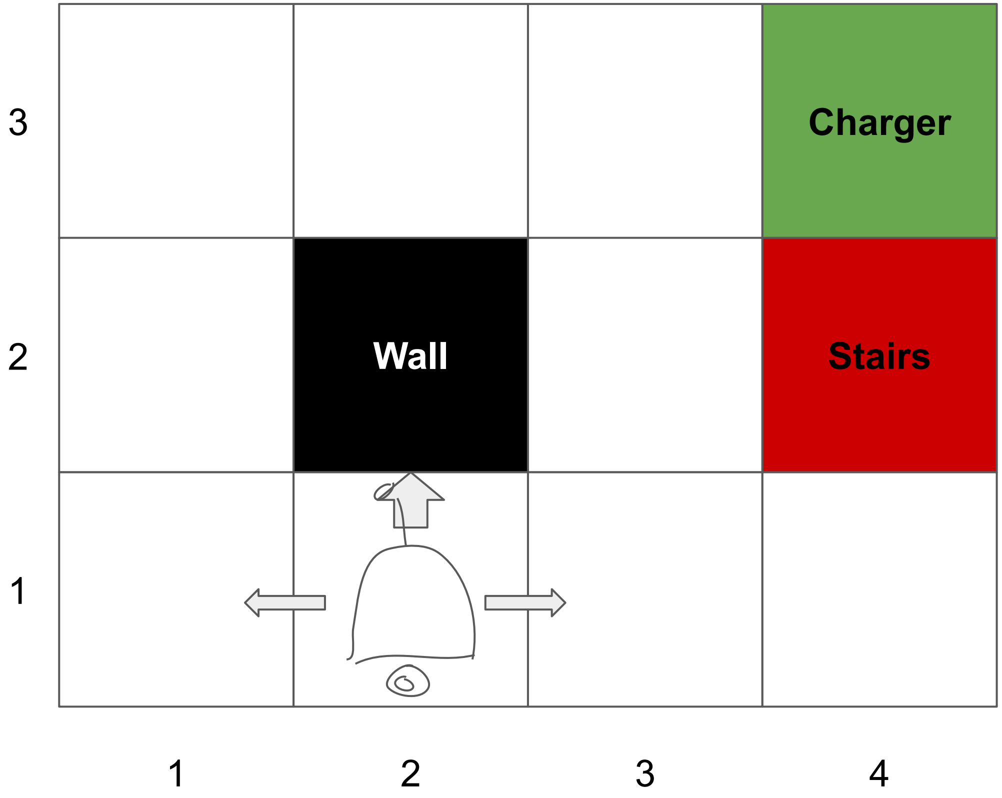
$$T((3,1), \mathrm{Up}, (3,2)) = 0.8$$
$$T((3,1), \mathrm{Up}, (2,1)) = 0.1$$
$$T((3,1), \mathrm{Up}, (4,1)) = 0.1$$



Transition Model - Example (2)

“Up” in (2,1)→
(2,1) w/ p=0.8
(1,1) w/ p=0.1
(3,1) w/ p=0.1

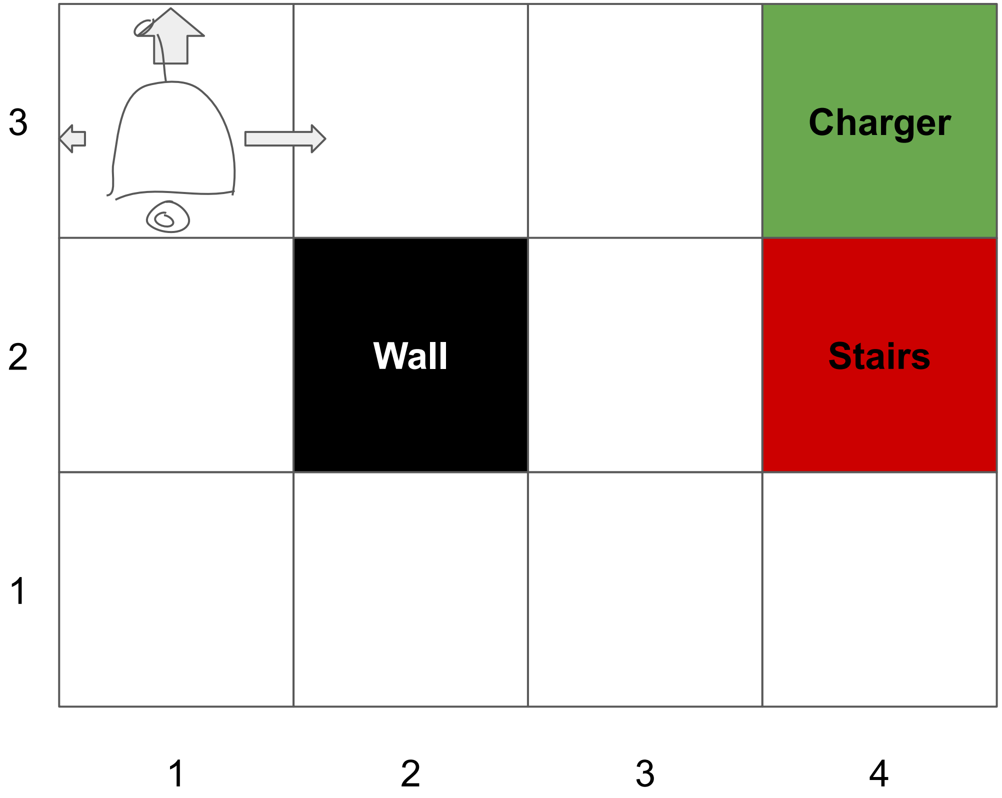
$$T((2,1),\mathrm{Up},(2,1)) = 0.8$$
$$T((2,1),\mathrm{Up},(1,1)) = 0.1$$
$$T((2,1),\mathrm{Up},(3,1)) = 0.1$$



Transition Model - Example (3)

“Up” in (1,3)→
(1,3) w/ p=0.9
(2,3) w/ p=0.1

$$T((1,3),\mathrm{Up},(1,3)) = 0.9$$
$$T((1,3),\mathrm{Up},(2,3)) = 0.1$$



Planning solution

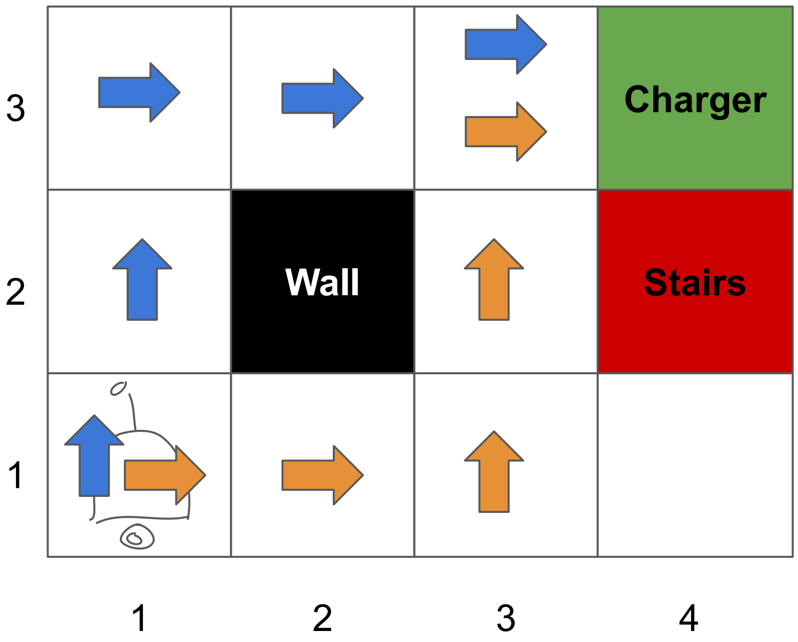
What's the planning solution for getting to the goal?

Deterministic version:
[U,U,R,R,R] or **[R,R,U,U,R]**

Probability of success?
(0.8)*(0.8)*(0.8)*(0.8)*(0.8) = 32%

Probability of success (by accident)?
(0.1)*(0.1)*(0.1)*(0.1)*(0.8) = 0.008%

Size of the **contingency plan**?
5 Layers, **with cycles for almost every action!**



Policy

What's an effective **policy** for getting to the charger?

A policy is a **mapping** from **states** to **actions**

$$\pi : S \to A$$

$$\text{Example:}$$

$$\pi((1,1)) = \text{“Up”}$$

The “best” policy is going to depend on our **performance measure** which we can encode as a **reward function**

3	π=Right	π=Right	π=Right	**Charger**
2	π=Up	**Wall**	π=Up	**Stairs**
1	π=Up	π=Right	π=Up	π=Left
	1	2	3	4

Reward

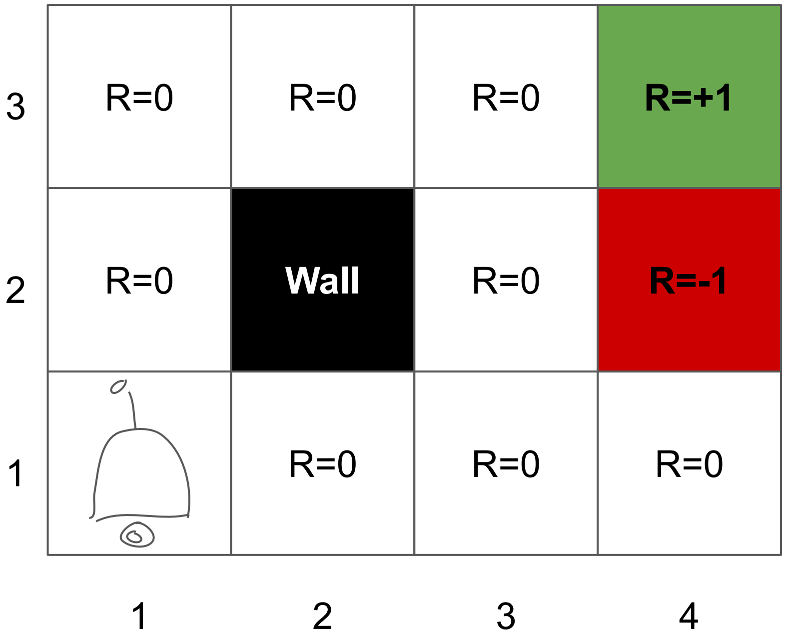
The **reward function** is a mapping from **states** to **real numbers** that gives a “score” for being in that state

$$R : S \to \mathbb{R}$$

$$\text{Example:}$$

$$R((4,3)) = +1$$

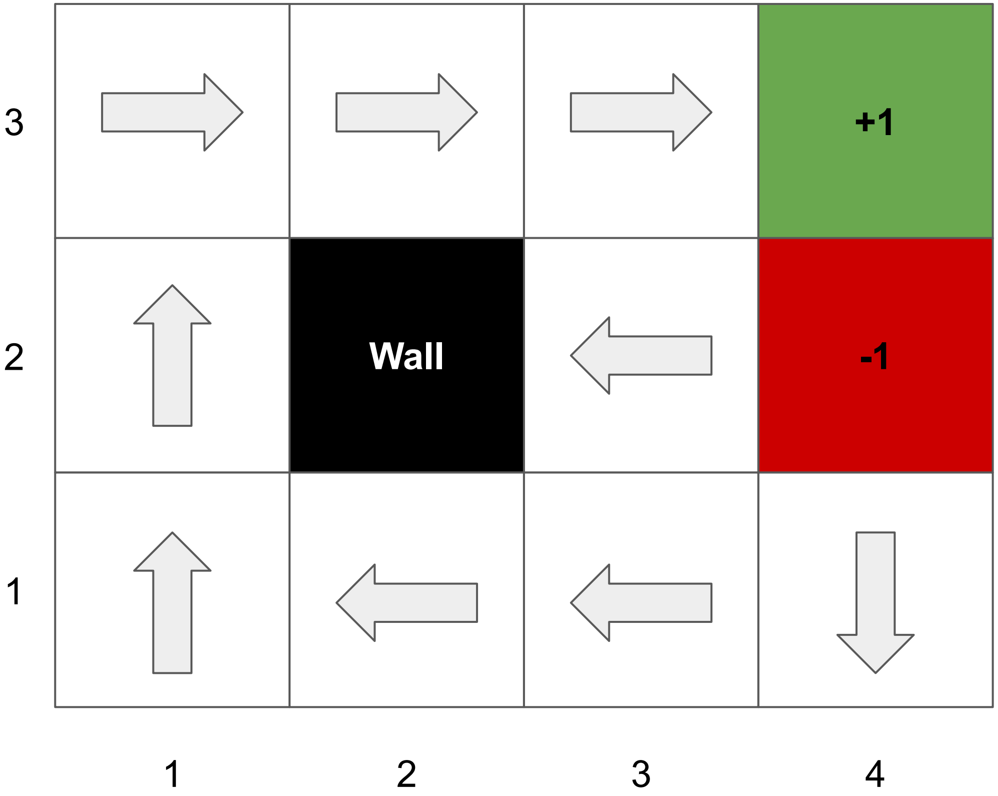
(Notation aside: sometimes reward is given as a function of taking a specific *action* in a state. These are mathematically equivalent)



Best policy, conservative version

Keep R(s) of red and green fixed

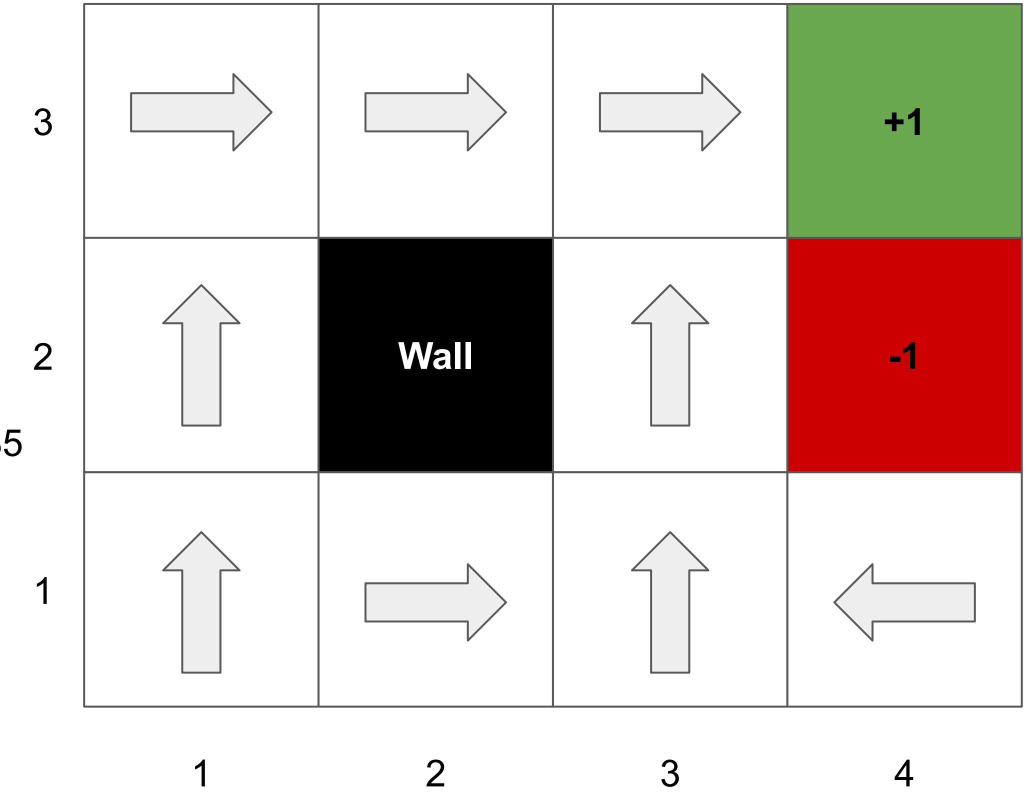
For every other state
$-0.0221 < R(s) < 0$



Best policy, speedy version

Keep R(s) of red and green fixed

For every other state
$-0.4278 < R(s) < -0.085$



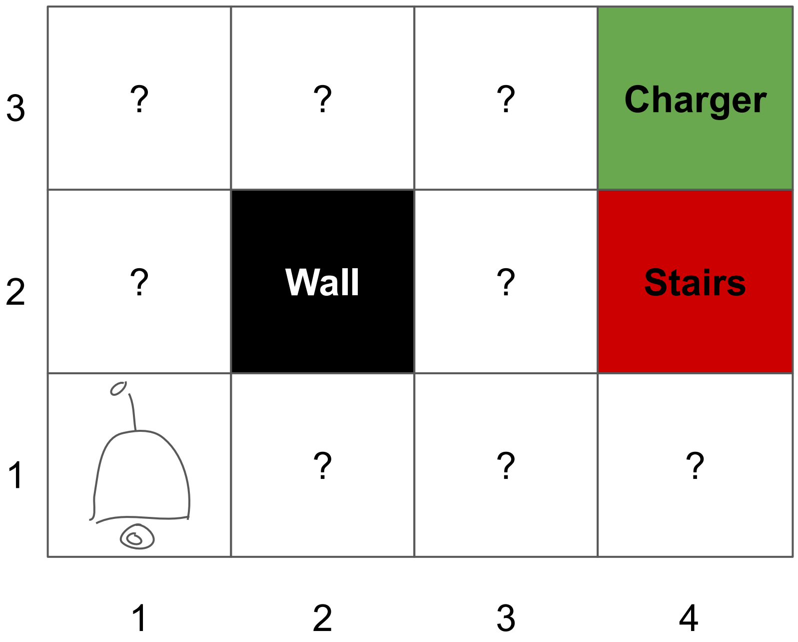
Finding the best policy

How can we **compute** the best policy given **R** and **T**?

High level
Use the **probabilities** in the transition model and the **values** of the reward to figure out the **utility** of each state. The optimal policy just greedily moves to the state with highest utility!

What is utility?

Expected long-term discounted reward



Defining Utility, attempt 1

How should we define utility? Let's try a few different approaches

Additive reward finite horizon

Sequence of visited states
$S_0, S_1, \ldots$

Initial state
$S_0 = s$

$$U(s) = \sum_{t=0}^{T} R(S_t)$$

Problem

How do we pick T?

Defining Utility, attempt 2

Additive reward infinite horizon

$$U(s) = \sum_{t=0}^{\infty} R(S_t)$$

Problem

Unbounded sum!

Defining Utility, attempt 3

Additive reward infinite horizon, discount factor

$$U(s) = \sum_{t=0}^{\infty} \gamma^t R(S_t),\ 0 < \gamma < 1$$

Problem

s_t are random variables!

Defining Utility, attempt 4

Expected discounted long-term reward

$$U^{\pi}(s) = \mathbb{E}_{S_t \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

(Note, equation 17.2 in the text)

What's so Markov about Markov Decision Processes

In order to decompose the utility in a useful way, we need to assert that our state space has the **Markov** property:

$$p(s_{t+1} \mid a, s_t, s_{t-1}, \ldots) = p(s_{t+1} \mid a, s_t)$$

This says is that the **sequence** of states that brought the agent to s_t doesn't matter for determining what the next state s_{t+1} will be. All that matters is the **immediate previous state** s_t. This in hand, we can write the optimal policy as

$$\pi^*(s) = \arg\max_a \sum_{s'} p(s' \mid s, a) U^{\pi^*}(s')$$

The Bellman equation

How do we compute $U^{\pi^*}(s)$ in the first place? There's an equation!

$$U^{\pi^*}(s) = R(s) + \gamma \max_a \sum_{s'} p(s' \mid s, a) U^{\pi^*}(s')$$

Notice that although this is looks like a circular definition, it's actually just recursive. We can solve this with a kind of dynamic programming, or maybe even with linear algebra.

Where did this equation come from? **Next time**

Summary and preview

Wrapping up

- MDPs are a framework for thinking about making decisions when actions have uncertain outcomes
- A **policy** is a mapping from any state to the “best” action for that state
- **Utility** is the **long-term expected discounted reward** of being in a particular state

Next time

- Bellman equation proof, Value Iteration, Policy Iteration