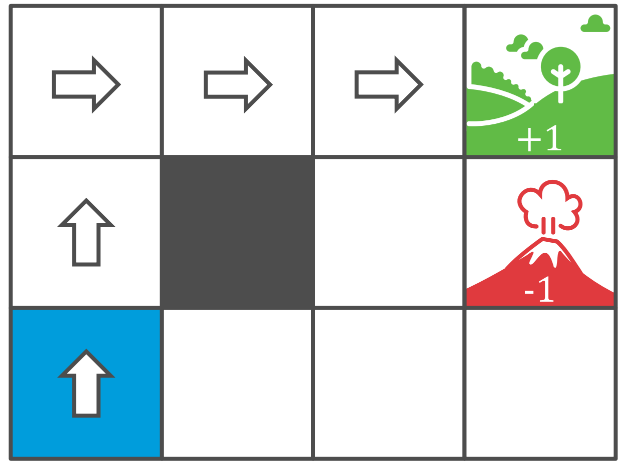
Fall 2022 | Lecture 15

Markov Decision Processes

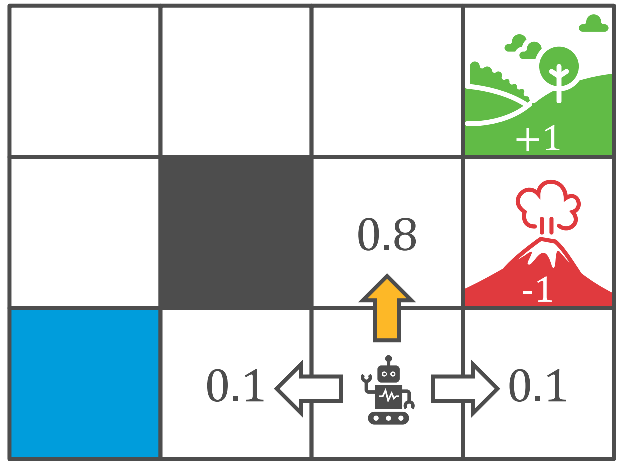
Ariel Procaccia | Harvard University

SEQUENTIAL DECISIONS: EXAMPLE



- We control a robot that can move in a grid world environment
- The robot gets a reward of +1 or −1 if it ends up in one of two special cells
- It’s clear what the solution is

SEQUENTIAL DECISIONS: EXAMPLE



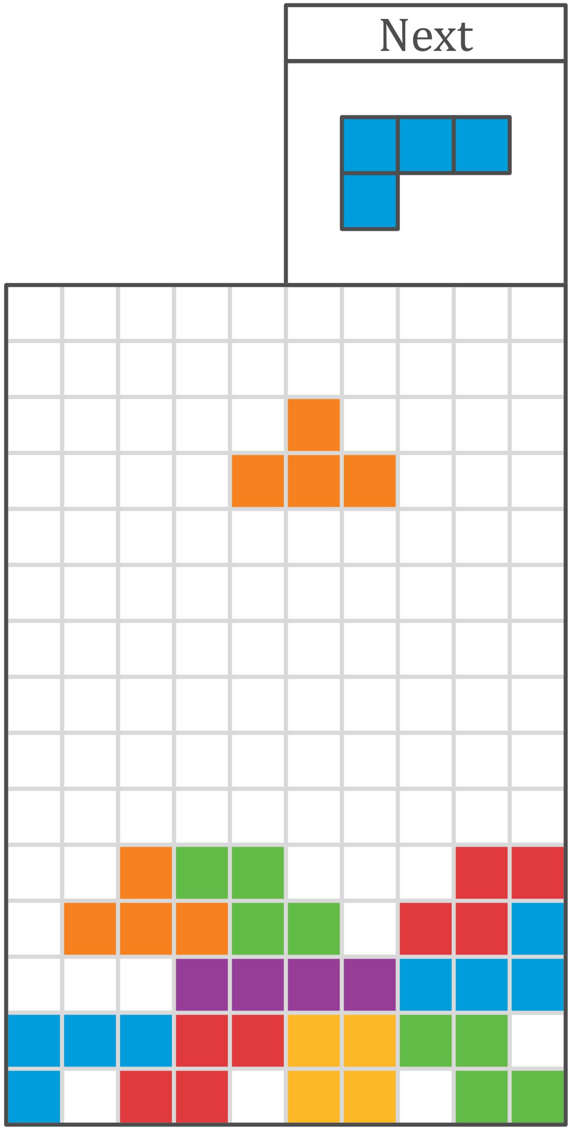
- To spice things up, assume that the robot only moves in the intended direction with probability 0.8 and with probability 0.2 moves at a right angle (collision leads to no movement)
- Reward of -0.04 in every cell except terminal ones

MARKOV DECISION PROCESSES

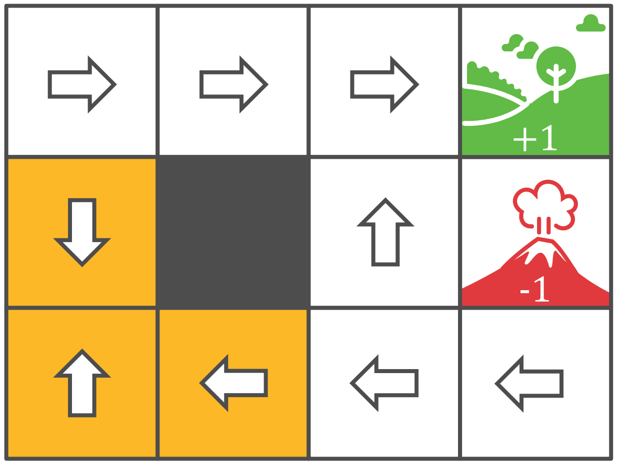
- A Markov Decision Process (MDP) consists of:
  - A set of states S with an initial state s_0
  - A set $A(s)$ of actions for each $s \in S$
  - A transition model $P(s' \mid s, a)$ that gives the probability of reaching s' if action a is taken in state s (transitions are Markovian)
  - A reward function $R(s)$ that specifies the reward of state s
- Assume for now that the agent's utility is the sum of rewards

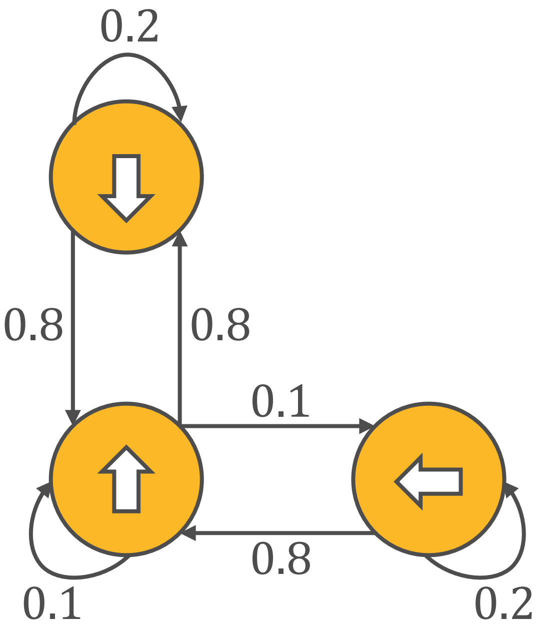
EXAMPLE: TETRIS

- Each state consists of the current piece, the next piece, and a bit matrix indicating which cells are filled
- Actions correspond to positions where the current piece can be placed, and only affect filled cells
- The only source of randomness in the transitions is the choice of the next piece



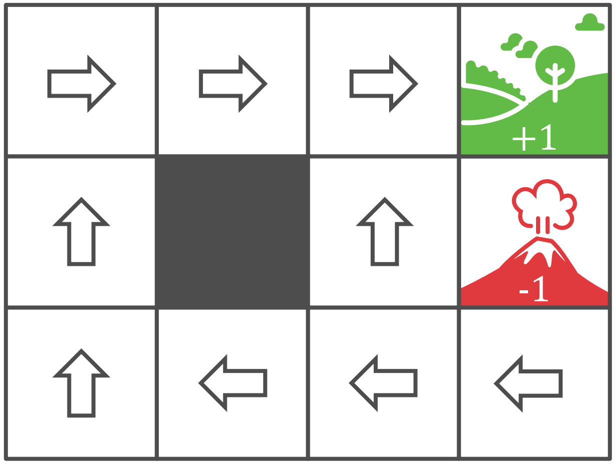
POLICIES





- A policy π specifies an action $\pi(s) \in A(s)$ for each $s \in S$
- With a fixed policy, an MDP induces a Markov chain

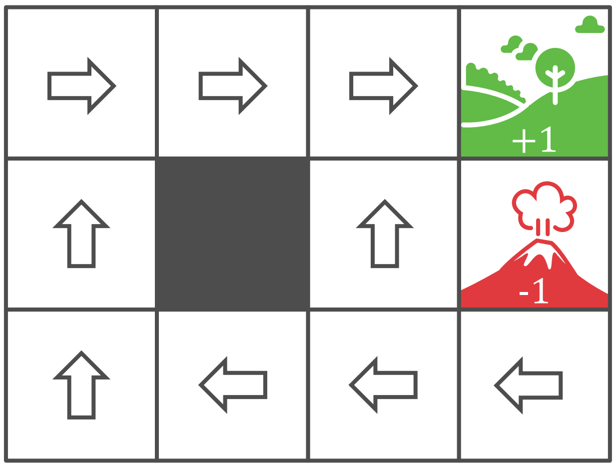
OPTIMAL POLICIES



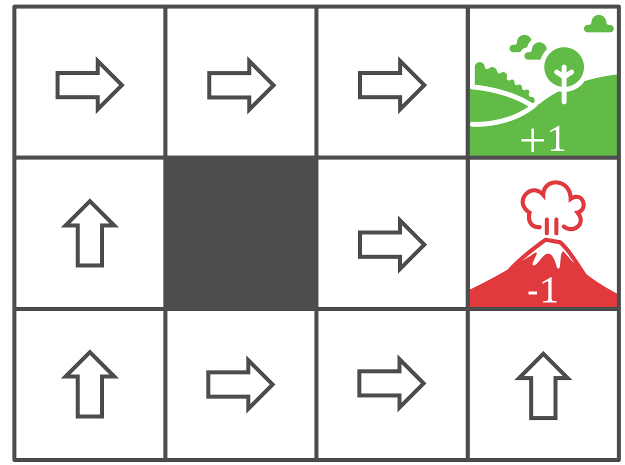
- We are interested in the expected utility yielded by a policy
- The optimal policy $\pi^\star$ maximizes expected utility

OPTIMAL POLICIES: EXAMPLE

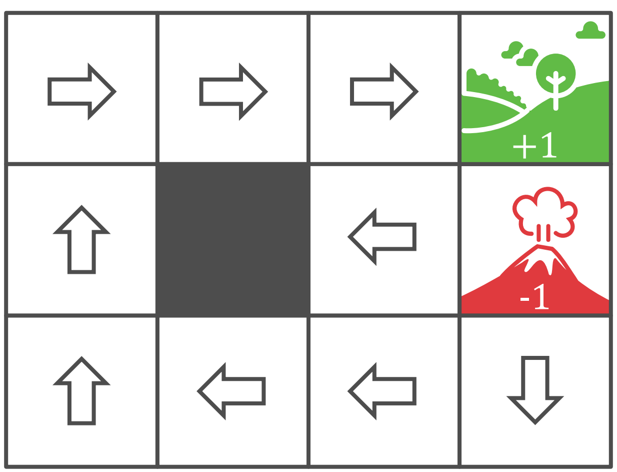
$R(s) = -0.04$



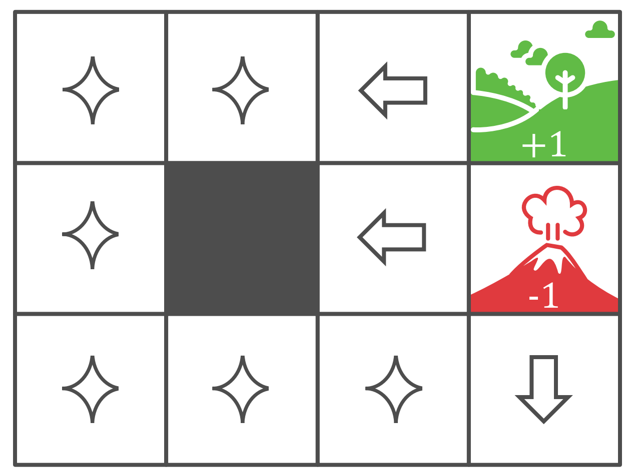
$R(s) = -2$



$R(s) = -0.01$



$R(s) > 0$

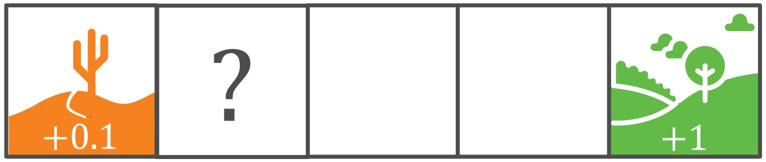


DISCOUNTED UTILITIES

- We assume that there's an infinite horizon
- We also assume that there's a discount factor $\gamma \in [0,1]$ such that the utility for a sequence of states $s_0, s_1, s_2, \ldots$ is $R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$
- Let $R(s) \leq r^\star$ for all $s \in S$
- If $\gamma < 1$ then the utility is bounded by

$$\sum_{t=0}^{\infty} \gamma^t r^\star = \frac{r^\star}{1-\gamma}$$

DISCOUNTED UTILITIES



- Poll 1: Assume $R(s) = 0$. What is the optimal policy at the question mark for $\gamma = 0.99$ and $\gamma = 0.01$?
  - Left and right
  - Right and left
  - Right and right
  - Left and left

BELLMAN EQUATIONS

- The utility of the optimal policy $U(s)$ at each state s is given by the Bellman equations:

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) \cdot U(s')$$

- Once we have the value of $U(s)$ for each $s \in S$ we can derive the optimal policy by taking

$$\pi^{\star}(s) \in \operatorname*{argmax}_{a \in A(s)} \sum_{s'} P(s' \mid s, a) \cdot U(s')$$

BELLMAN EQUATIONS: EXAMPLE

$\gamma = 1, R(s) = -0.04$

3	.812	.868	.918	 +1
2	.762		.660	 -1
1	.705	.655	.611	.388
	1	2	3	4

$$
\begin{aligned}
U(1,1) = -0.04 + \max\{&0.8U(1,2) + 0.1U(2,1) + 0.1U(1,1) && \Uparrow \\
&0.9U(1,1) + 0.1U(1,2) && \Leftarrow \\
&0.9U(1,1) + 0.1U(2,1) && \Downarrow \\
&0.8U(2,1) + 0.1U(1,2) + 0.1U(1,1)\} && \Rightarrow
\end{aligned}
$$

VALUE ITERATION

- Iteratively update utility estimates $U_i(s)$ via

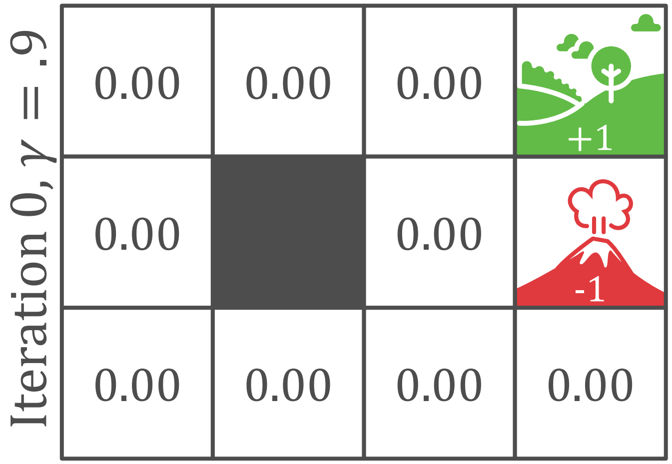
$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U_i(s')$$

- Stopping condition: for a given $\epsilon > 0$,

$$\max_{s \in S} |U_{i+1}(s) - U_i(s)| < \frac{\epsilon(1-\gamma)}{\gamma}$$

- Theorem: If $\gamma < 1$ then the algorithm terminates with utility estimates U_t such that for all $s \in S$, $|U(s) - U_t(s)| < \epsilon$

VALUE ITERATION: EXAMPLE



- Poll 2: Assume $R(s) = 0$. How many nonzero values are there (excluding the forest and the volcano) after one iteration of value iteration?
  - 0
  - 1
  - 2
  - 3

VALUE ITERATION: EXAMPLE

Iteration 0, $\gamma = .9$

0.00	0.00	0.00	+1
0.00		0.00	-1
0.00	0.00	0.00	0.00

Iteration 1, $\gamma = .9$

0.00	0.00	0.72	+1
0.00		0.00	-1
0.00	0.00	0.00	0.00

Iteration 2, $\gamma = .9$

0.00	0.52	0.78	+1
0.00		0.43	-1
0.00	0.00	0.00	0.00

Iteration 3, $\gamma = .9$

0.37	0.66	0.83	+1
0.00		0.51	-1
0.00	0.00	0.31	0.00

POLICY ITERATION

- Alternate between two steps, beginning at an initial policy π_0
- Step 1 (policy evaluation): given a policy π_i, calculate its utility U_i via

$$U_i(s) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi_i(s)) \cdot U_i(s')$$

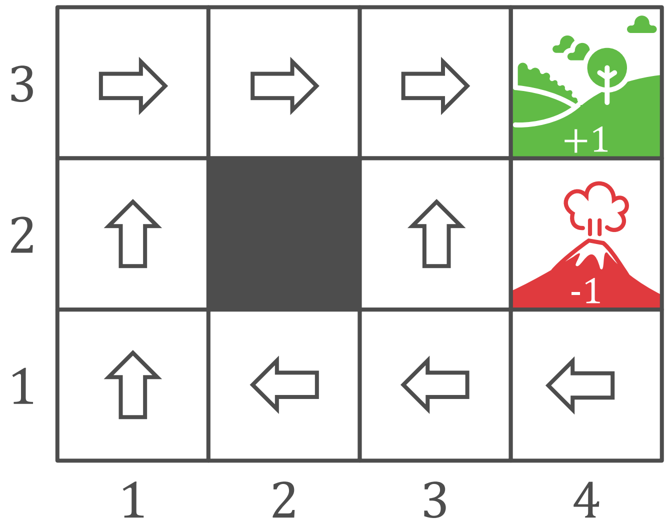
- Step 2 (policy improvement): calculate a new policy π_{i+1} based on U_i via

$$\pi_{i+1}(s) \in \underset{a \in A(s)}{\operatorname{argmax}} \sum_{s'} P(s' \mid s, a) \cdot U_i(s')$$

- Terminate on bounded change in utilities

POLICY EVALUATION: EXAMPLE

$$\gamma = 1, R(s) = -0.04$$



$$U_i(1,1) = -0.04 + 0.8U_i(1,2) + 0.1U_i(2,1) + 0.1U_i(1,1)$$

$$U_i(1,2) = -0.04 + 0.8U_i(1,3) + 0.2U_i(1,2)$$

$$U_i(1,3) = -0.04 + 0.8U_i(2,3) + 0.1U_i(1,2) + 0.1U_i(1,3)$$

LINEAR PROGRAMMING

The optimal utility function U can be also be computed by a linear program with variables $U(s)$ for all $s \in S$:

$$\min \sum_{s \in S} U(s)$$

$$\text{s.t} \quad \forall s \in S, a \in A(s), U(s) \geq R(s) + \gamma \sum_{s'} P(s' \mid s, a) \cdot U(s')$$