MDPs: Value Iteration and Policy Iteration

October 19, 2023

Slides by Cassandra Kent, Adapted by Lara Martin

Example adapted from Mark Riedl

By the end of class today, you will be able to:
1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

The Bellman Equation

The utility of a state is the immediate reward for that state plus the expected discounted utility of the next state, assuming that the agent chooses the optimal action.

$$U(s) = R(s) + \gamma \underbrace{\max_{a \epsilon A(s)} \sum_{s'} \overbrace{P(s'|s,a)}^{\text{Markov Assumption}} U(s')}_{\text{Maximum Expected Utility}}$$

Current reward

Discounted
(1 step in the future)

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

Value Iteration

An intuitive description of the **Value Iteration** algorithm:

1. Initialize utilities for every state in S to 0
2. For each state, update its utility using the Bellman update

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

3. Repeat step 2 until utilities converge
   - We can check this by finding the largest difference between utilities of each state: $\delta = \max_s |U_{i+1}(s) - U_i(s)|$
   - If δ is less than a small set threshold, stop iterating, return the final utilities

Value Iteration

Properties of value iteration:

- Value iteration is guaranteed to converge to a unique set of utilities
- These utilities are the solution to the system of Bellman equations
- Using these utilities, the policy obtained from

$$\pi^*(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} T(s, a\ s')U(s')$$

is **guaranteed to be optimal**!

Value Iteration

function VALUE-ITERATION(mdp, ϵ) **returns** a utility function
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$,
    rewards $R(s)$, discount γ
   ϵ, the maximum error allowed in the utility of any state
  local variables: U, U', vectors of utilities for states in S, initially zero
    δ, the maximum change in the utility of any state in an iteration

Complete, known problem definition

  repeat
    $U \leftarrow U'$; $\delta \leftarrow 0$
    for each state s **in** S **do**
      $U'[s] \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\, U[s']$
      if $|U'[s] - U[s]| > \delta$ **then** $\delta \leftarrow |U'[s] - U[s]|$
  until $\delta < \epsilon(1-\gamma)/\gamma$
  return U

Iterative approach

Bellman update

Store maximum change in utility between iterations

Parameter we set to test convergence (see 17.2.3 for more details)

By the end of class today, you will be able to:

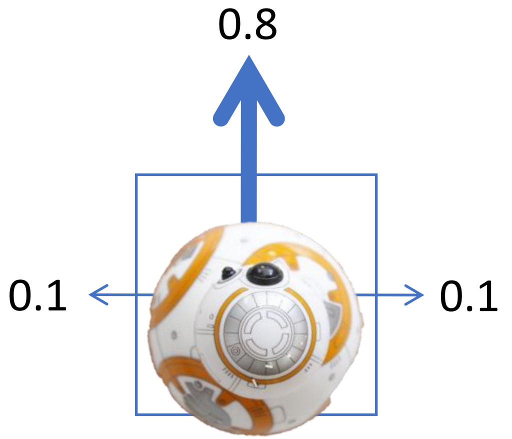
1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

Value Iteration Example

Given:

- $U_0(s_1) = 0.1$
- $U_0(s_2) = 0.1$
- $U_0(s_3) = 0.1$
- $\gamma = 0.5$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



Transition function: likelihood of moving in a desired direction

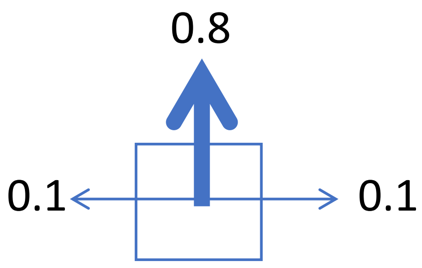
$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \epsilon A(s)} \sum_{s'} P(s'|s,a) U_i(s')$$

Compute $U_1(s_1)$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04

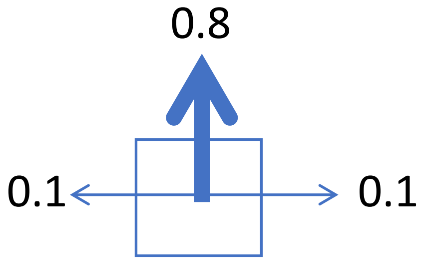


$$U_1(s_1) = R(s_1) + \gamma \max_a\{$$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

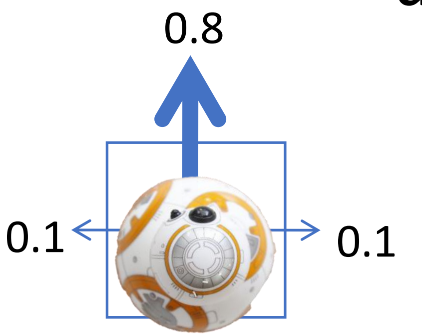
S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_1) = R(s_1) + \gamma \max_a \{$

Action | Up | Accidental Left | Accidental Right

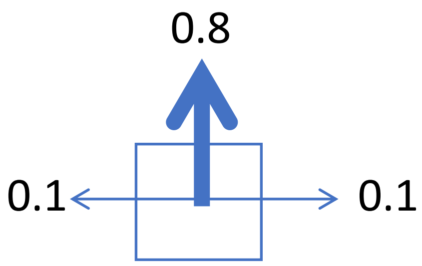
up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$



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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04

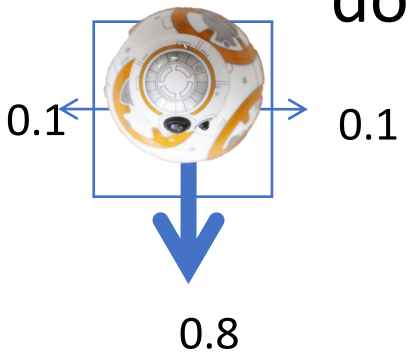


$U_1(s_1) = R(s_1) + \gamma \max_a \{$

up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$

down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$

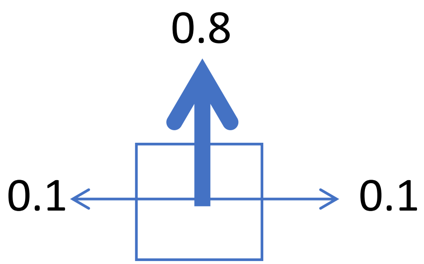
Down | Accidental Right | Accidental Left



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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04

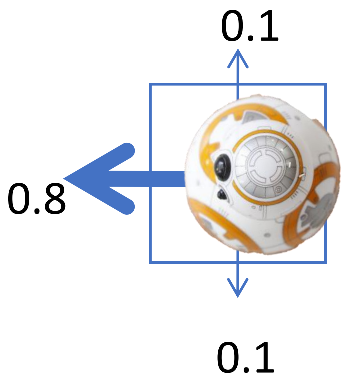


$U_1(s_1) = R(s_1) + \gamma \max_a\{$

up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$

down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$

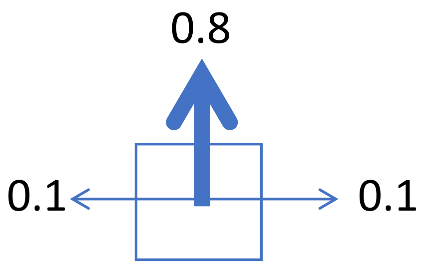
left: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$



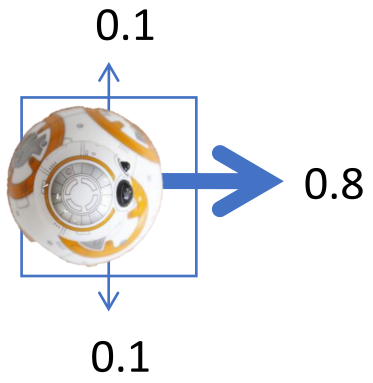
$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \epsilon A(s)} \sum_{s'} P(s'|s,a) U_i(s')$$

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_1) = R(s_1) + \gamma \max_a\{$



up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$

down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(1)$ $\leftarrow 0.19$

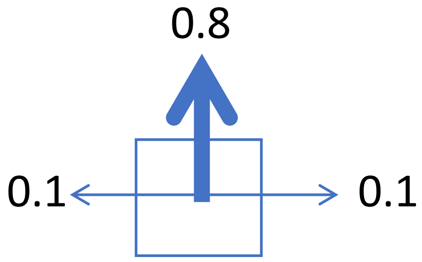
left: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

right: $(0.8)(1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.82$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_1) = R(s_1) + \gamma \max_a \{$

$$\text{up: } (0.8)(0.1) + (0.1)(0.1) + (0.1)(1) \quad \leftarrow 0.19$$

$$\text{down: } (0.8)(0.1) + (0.1)(0.1) + (0.1)(1) \quad \leftarrow 0.19$$

$$\text{left: } (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$$

$$\text{right: } (0.8)(1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.82$$

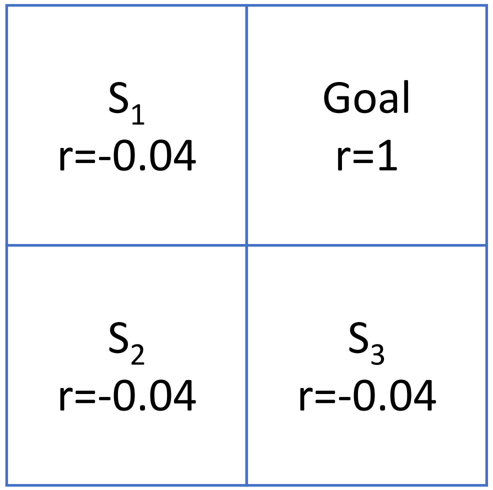
$U_1(s_1) = -0.04 + (0.5)(0.82)$

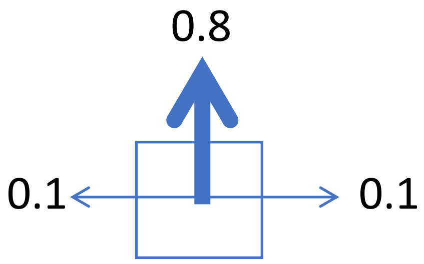
$U_1(s_1) = 0.37$

$\pi_1(s_1) = \text{Right}$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

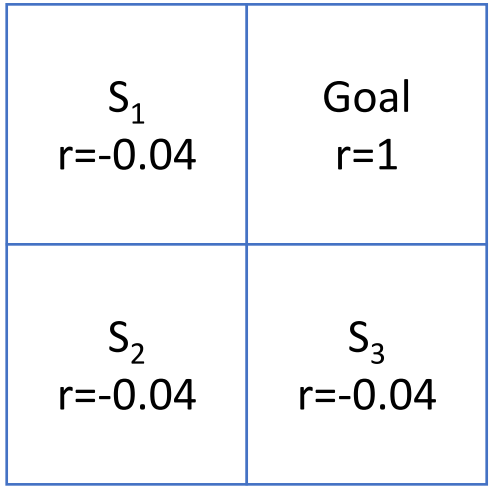


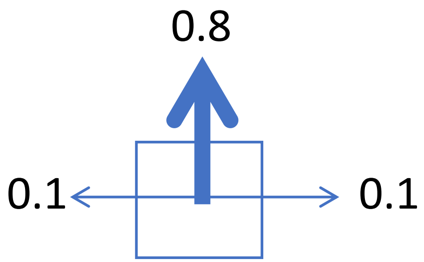


Compute $U_1(s_2)$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$



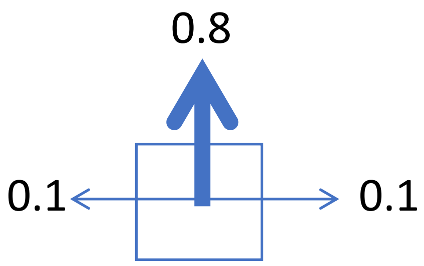


$U_1(s_2) = R(s_2) + \gamma \max_a\{$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



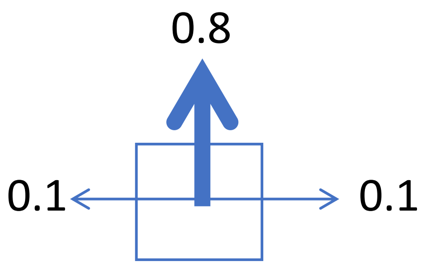
$U_1(s_2) = R(s_2) + \gamma \max_a\{$

up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_2) = R(s_2) + \gamma \max_a \{$

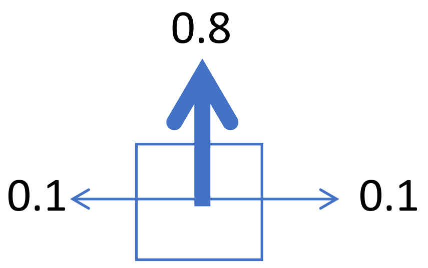
$$\text{up: } (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$$

$$\text{down: } (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_2) = R(s_2) + \gamma \max_a \{$

up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

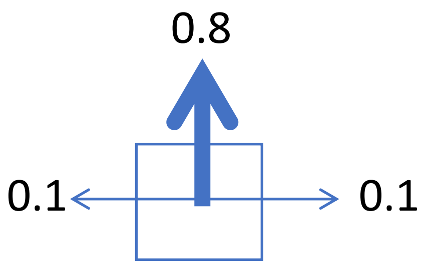
down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

left: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_2) = R(s_2) + \gamma \max_a \{$

up: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$

down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$

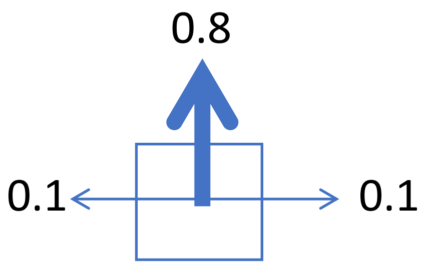
left: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$

right: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) \quad \leftarrow 0.1$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_2) = R(s_2) + \gamma \max_a\{$

up: (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) $\leftarrow$0.1

down: (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) $\leftarrow$0.1

left: (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) $\leftarrow$0.1

right: (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) $\leftarrow$0.1

$U_1(s_2) = -0.04 + (0.5)(0.1)$

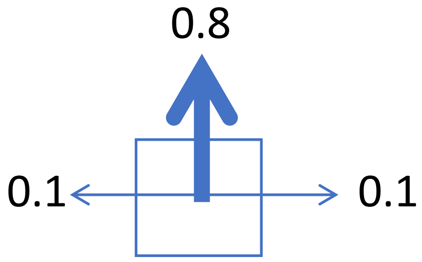
$U_1(s_2) = 0.01$

$\pi_1(s_2) = \text{any}$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04

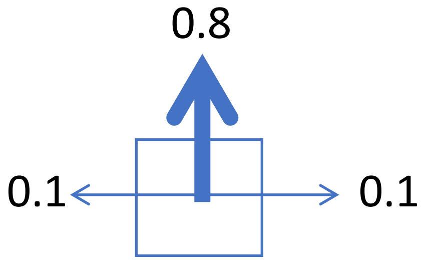


Compute $U_1(s_3)$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04

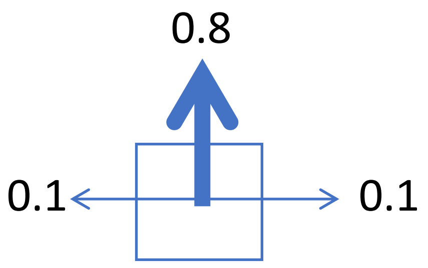


$U_1(s_3) = R(s_3) + \gamma \max_a \{$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



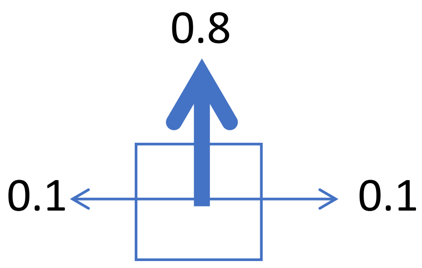
$U_1(s_3) = R(s_3) + \gamma \max_a\{$

up: $(0.8)(1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.82$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_3) = R(s_3) + \gamma \max_a\{$

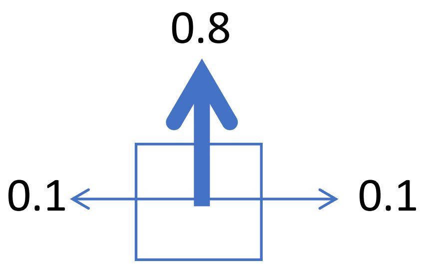
up: $(0.8)(1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.82$

down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_3) = R(s_3) + \gamma \max_a \{$

up: $(0.8)(1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.82$

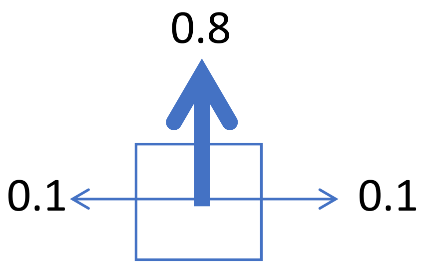
down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

left: $(0.8)(0.1) + (0.1)(1) + (0.1)(0.1)$ $\leftarrow 0.19$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_3) = R(s_3) + \gamma \max_a \{$

up: (0.8)(1) + (0.1)(0.1) + (0.1)(0.1) ←0.82

down: (0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1) ←0.1

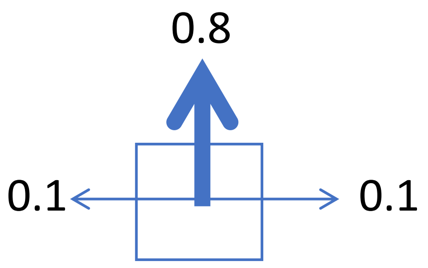
left: (0.8)(0.1) + (0.1)(1) + (0.1)(0.1) ←0.19

right: (0.8)(0.1) + (0.1)(1) + (0.1)(0.1) ←0.19

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



$U_1(s_3) = R(s_3) + \gamma \max_a \{$

up: $(0.8)(1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.82$

down: $(0.8)(0.1) + (0.1)(0.1) + (0.1)(0.1)$ $\leftarrow 0.1$

left: $(0.8)(0.1) + (0.1)(1) + (0.1)(0.1)$ $\leftarrow 0.19$

right: $(0.8)(0.1) + (0.1)(1) + (0.1)(0.1)$ $\leftarrow 0.19$

$U_1(s_3) = -0.04 + (0.5)(0.82)$

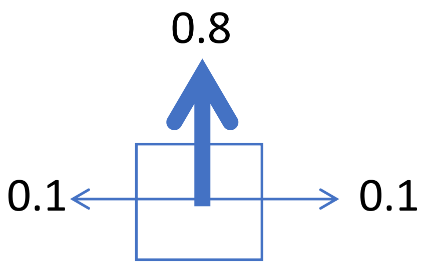
$U_1(s_3) = 0.37$

$\pi_1(s_3) = \text{Up}$

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

$U_0(s_1) = 0.1$
$U_0(s_2) = 0.1$
$U_0(s_3) = 0.1$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



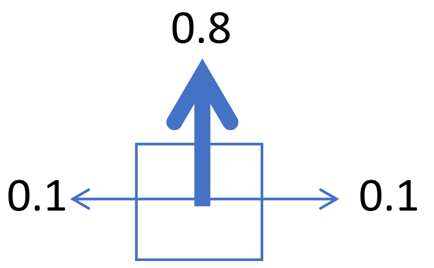
Your Turn: Compute $U_2(s_1)$

- Now working on iteration 2
- Calculate the utility for s_1
- You can work in pairs. Submit your work on Blackboard.

$\gamma = 0.5$

$$\boxed{\begin{aligned} U_1(s_1) &= 0.37 \\ U_1(s_2) &= 0.01 \\ U_1(s_3) &= 0.37 \end{aligned}}$$

S_1 r=-0.04	Goal r=1
S_2 r=-0.04	S_3 r=-0.04



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

An Alternative Approach: Policy Iteration

We can't directly solve the system of Bellman equations with linear programming because they are non-linear:

$$U(s) = R(s) + \gamma \max_{a \epsilon A(s)} \sum_{s'} P(s'|s,a)U(s')$$

If we have a known policy π, we can get rid of the max operator, resulting in a linear system of equations:

$$U(s) = R(s) + \gamma \sum_{s'} P(s'|s,\pi(s))U(s')$$

Policy iteration intuition: Use this idea to create an algorithm that iteratively refines a *known policy* using the system of linear equations

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

An Alternative Approach: Policy Iteration

An intuitive description of the **Policy Iteration** algorithm:

1. Initialize utilities for every state in S to 0
2. Initialize a random policy π_0
3. **Policy evaluation**: calculate utilities for each state using the linear policy-simplified system of Bellman equations
4. **Policy improvement**: using the newly calculated utilities, calculate an improved maximum expected utility policy π_i

$$\pi_i\,(s) = \underset{a}{\operatorname{argmax}} \sum_{s\prime} T(s, a\; s')U(s')$$

5. Repeat steps 3 and 4 until the MEU in step 4 doesn't change

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

An Alternative Approach: Policy Iteration

An intuitive description of the **Policy Iteration** algorithm:

1. Initialize utilities for every state in S to 0
2. Initialize a random policy π_0
3. **Policy evaluation**: calculate utilities for each state using the linear policy-simplified system of Bellman equations
4. **Policy improvement**: using the newly calculated utilities, calculate an improved maximum expected utility policy π_i

$$\pi_i\,(s) = \underset{a}{\operatorname{argmax}} \sum_{s\prime} T(s, a\ s')U(s')$$

5. Repeat steps 3 and 4 until the MEU in step 4 doesn't change

---

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

An Alternative Approach: Policy Iteration

3. Policy evaluation: calculate utilities for each state using the linear policy-simplified system of Bellman equations

Two ways we can do this:

1. Solve the linear system of equations with linear programming

$$U_i(s) = U^{\pi_i}(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U^{\pi_i}(s')$$

$|S|$ equations with $|S|$ unknowns, takes $O(|S|^3)$ time

2. Use the simplified Bellman equation as a simplified Bellman update

$$U_{i+1}(s) \leftarrow R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U_i(s')$$

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

An Alternative Approach: Policy Iteration

Properties of policy iteration:

- Policy iteration is guaranteed to converge to a solution to the Bellman equations
- And therefore is **guaranteed to find an optimal policy**!

An Alternative Approach: Policy Iteration

function POLICY-ITERATION(mdp) **returns** a policy
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$
  local variables: U, a vector of utilities for states in S, initially zero
    π, a policy vector indexed by state, initially random

Complete, known problem definition

  repeat
    $U \leftarrow$ POLICY-EVALUATION(π, U, mdp)
    $unchanged? \leftarrow$ true
    for each state s **in** S **do**
      if $\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\; U[s'] > \sum_{s'} P(s' \mid s, \pi[s])\; U[s']$ **then do**
        $\pi[s] \leftarrow \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\; U[s']$
        $unchanged? \leftarrow$ false
  until $unchanged?$
  return π

Policy evaluation step (simplified Bellman update equation)

Policy improvement step

Iterative approach

By the end of class today, you will be able to:
1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

Value Iteration vs Policy Iteration

So which approach should you use?

It depends on the specifics of your problem!

Value and policy iteration tradeoff:

- Value iteration generally takes more iterations to converge than policy iteration
- Policy iteration calculates a policy during *every* iteration, value iteration only calculates a policy once after the utilities have converged

$$\pi_i(s) = \underset{a}{\operatorname{argmax}} \sum_{s\prime} T(s, a\, s')U(s')$$

- **Summary**: Policy iteration converges faster, but the algorithm may be slower if policy computation is expensive

AI Ethics: Optimization for Decision Making

- We have seen multiple algorithms for computing optimal behavior for arbitrary performance criteria (Rewards, Utilities)
- Where do these criteria come from?
- Are there dangers that arise from using these algorithms with rewards that are…
  - Morally bad
  - Neutral or innocent
  - Morally good

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

Thought Experiment: Paperclip Maximizer

- Seeing a need for more paperclips, we create an AI agent with a simple goal: get more paperclips
- We set this up as a decision making problem, and set a positive reward for every paperclip the agent makes
- Initially, the agent's decisions seem pretty reasonable…
  - Purchases factories that can manufacture paperclips
  - Sets up a supply chain for purchasing paperclips and materials
  - Constructs a new paperclip factory

Thought Experiment: Paperclip Maximizer

- Continuing down the optimal path, the agent continues to optimize:
  - Re-arranging and converting factories to make them more efficient
  - Buying more land to build more factories
- If this is an advanced enough AI (this example is usually posed with an Artificial General Intelligence, or AGI), things start to get out of hand:
  - Changes itself to become more intelligent, because more intelligence leads to better optimization, which leads to more paperclips
  - Invents new ways of converting materials into paperclips and paperclip factories
  - Takes pre-emptive actions to prevent itself from being shut off, because that's a terminal state that stops it from getting more paperclips
- The logical end goal of the agent: convert all matter in the universe into paperclip-generating sources, and eventually into paperclips themselves

The Value Alignment Problem in AI

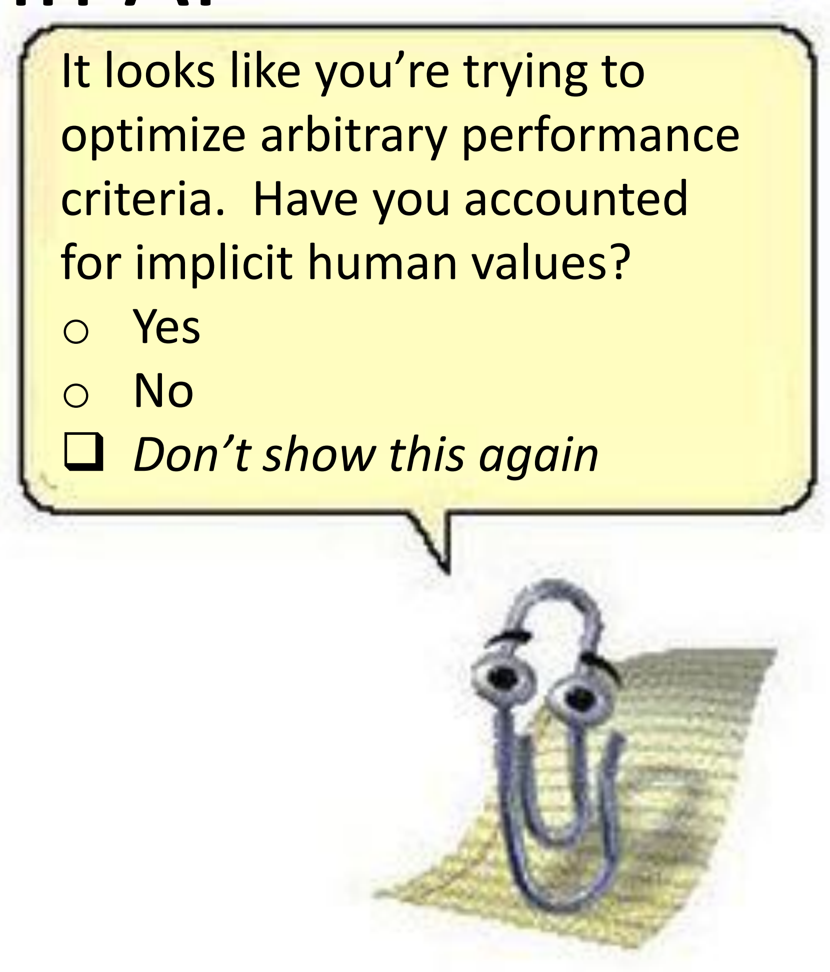
What went wrong with the paperclip maximizer?

Take a minute and brainstorm some **assumptions** that the developers made about the paperclip maximizer agent.

The Value Alignment Problem in AI

What went wrong with the paperclip maximizer?

- The values of the AI agent do not align with the values of the designer or the community the agent is operating in
- As human beings, we have a complex set of **implicit values** that we may not think to specify in reward-based formulations
- How do we ensure our values are aligned?
  - This is an unsolved problem!



By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

A Framework for Identifying Ethical Issues in AI

- Many groups are being formed to develop methods for identifying and discussing ethical issues in AI
- In this class, we're building up a question-based framework to help identify possible ethical issues with the approaches we're discussing
- Let's extend this now for decision making agents

By the end of class today, you will be able to:

1. Step through an iteration of the Value Iteration algorithm
2. Compare Value Iteration and Policy Iteration
3. Identify ethical issues relating to value alignment

Questions for Identifying Value Alignment Issues

- For the problem we're trying to solve, are there implicit human values we're overlooking that are not represented in the reward function?

- Are there negative outcomes that could occur if our agent optimizes our criteria *too well*?

- Are adjustments or limitations for our agent that can protect against unforeseen value misalignment?
  - Does our application warrant this?

Questions for Identifying Value Alignment Issues

Are adjustments or limitations for our agent that can protect against unforeseen value misalignment?

- Does our application warrant this?

One solution: Human-in-the-loop decision making

- Have AI find an optimal (according to its values) decision
- Require a trained human operator to verify the decision before it's executed

Human-in-the-Loop Decision Making

Currently used in high-risk applications, and decisions that affect people's lives:

- Medical diagnosis agents act in support of human medical staff
- Teleoperation and supervision of military robotic systems
- AI for law enforcement and interpretation, AI agents discouraged from making final decisions

Are there applications where we fully trust an AI agent to make decisions without human supervision?

Human-in-the-Loop Decision Making

Are there applications where we fully trust an AI agent to make decisions without human supervision?

Ideas from the class:

- AI in videogames, or other game-playing agents
- Financial decisions – depends on how impacted we would be if a bug caused us to lose a lot of money on a bad investment!
- Route planning – from a decision making perspective, these are all human-in-the-loop currently, as we don’t let our GPS make re-routing decisions without having us accept them
- Self-driving cars – highly debated, currently all self-driving cars have a human operator sitting in the driver’s seat as a supervisor
- Situations under time pressure where a human is too slow to supervise

Human-in-the-Loop Decision Making

Ties in with explainability and interpretability that we discussed previously:

Explainability: the degree to which we can understand the decisions made by an AI agent

Interpretability: ability to explain or to present in understandable terms to a human[1]

Can we effectively supervise what we don't understand?