University of Colorado **Boulder**

Department of Computer Science

CSCI 5622: Machine Learning

Chris Ketelsen

Lecture 20:
Reinforcement Learning Part 1

Reinforcement Learning - Motivation

Reinforcement Learning addresses the question of how an autonomous agent that senses and acts in its environment can learn to choose optimal actions to achieve its goals.

Tom Mitchell, *Machine Learning*

Reinforcement Learning - Motivation

Reinforcement Learning addresses the question of how an autonomous agent that senses and acts in its environment can learn to choose optimal actions to achieve its goals.

Tom Mitchell, *Machine Learning*

Examples:

- Train a roomba to find its docking station
- Train a computer to play Super Mario
- Train a computer to play backgammon

Reinforcement Learning - Motivation

This is a new type of learning environment we haven't seen before

$$\text{Supervised Learning:} \qquad y = f(x)$$

Reinforcement Learning - Motivation

This is a new type of learning environment we haven't seen before

$$\text{Supervised Learning:} \qquad y = f(x)$$

$$\text{Unupervised Learning:} \qquad f(x)$$

Reinforcement Learning - Motivation

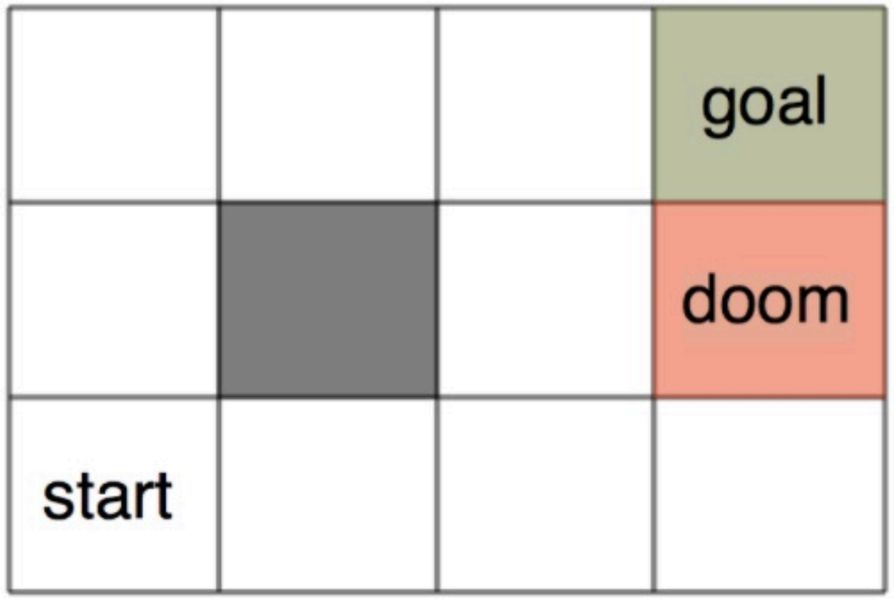
This is a new type of learning environment we haven't seen before

Supervised Learning: $y = f(x)$

Unupervised Learning: $f(x)$

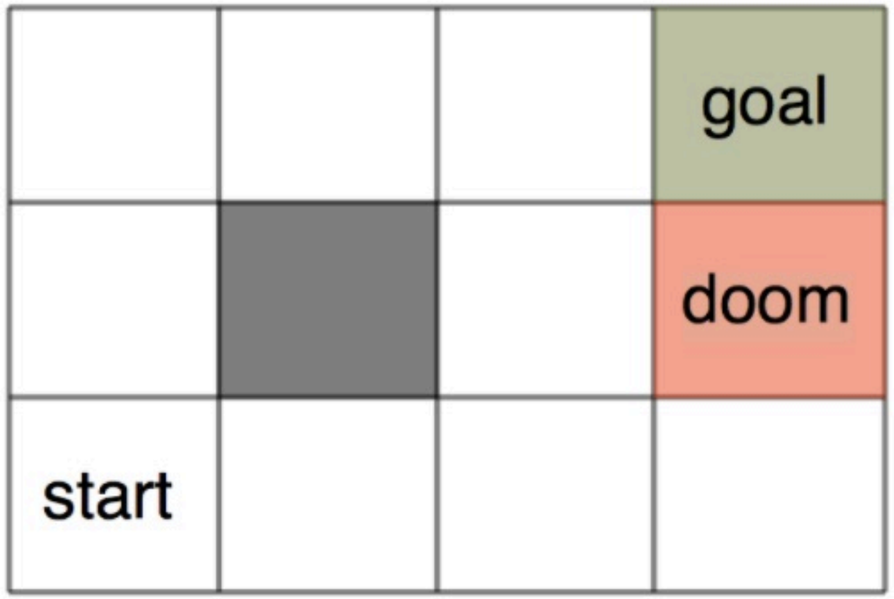
Reinforcement Learning: $y = f(x), \quad z$

The Grid World



Available Actions: Up, Down, Left, Right

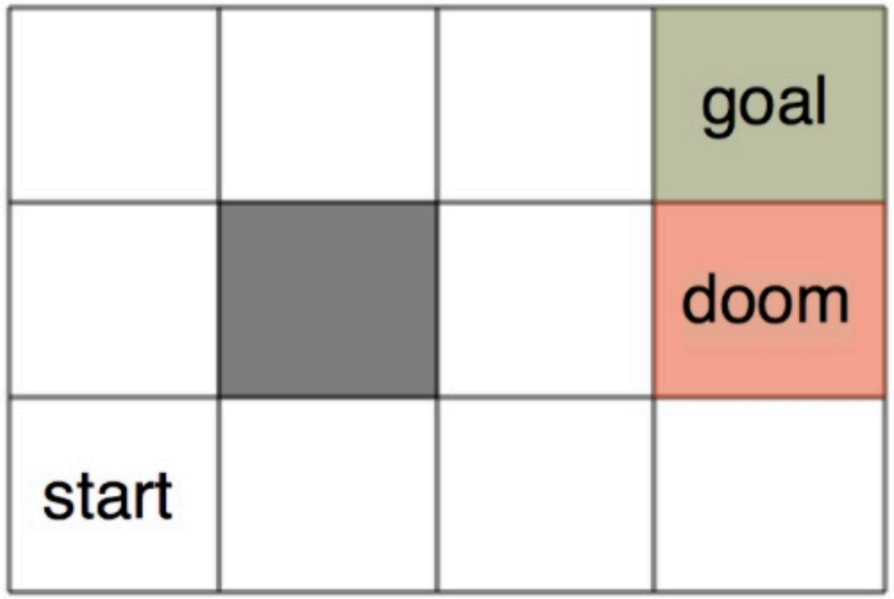
The Grid World



Available Actions: Up, Down, Left, Right

Question: What is shortest sequence from the **start** to the **goal**?

The Grid World

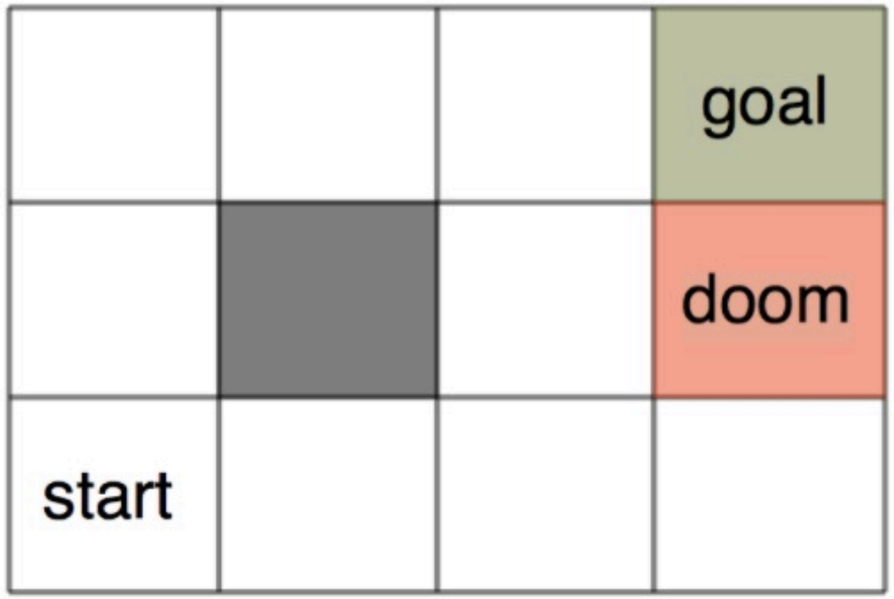


Available Actions: Up, Down, Left, Right

Question: What is shortest sequence from the **start** to the **goal**?

Answer: Up, Up, Right, Right, Right

The Grid World



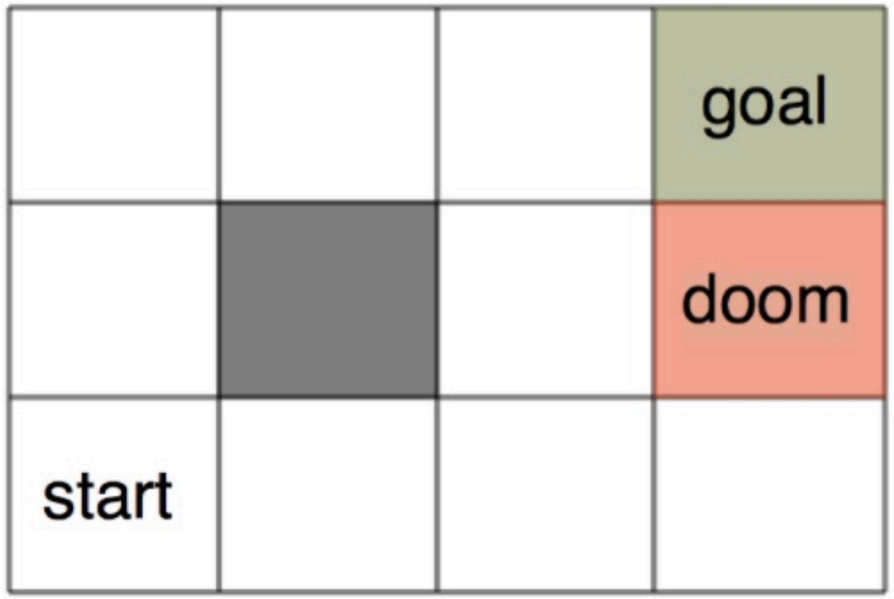
Available Actions: Up, Down, Left, Right

Question: What is shortest sequence from the **start** to the **goal**?

Answer: Up, Up, Right, Right, Right

OK... Right, Right, Up, Up, Right would work too

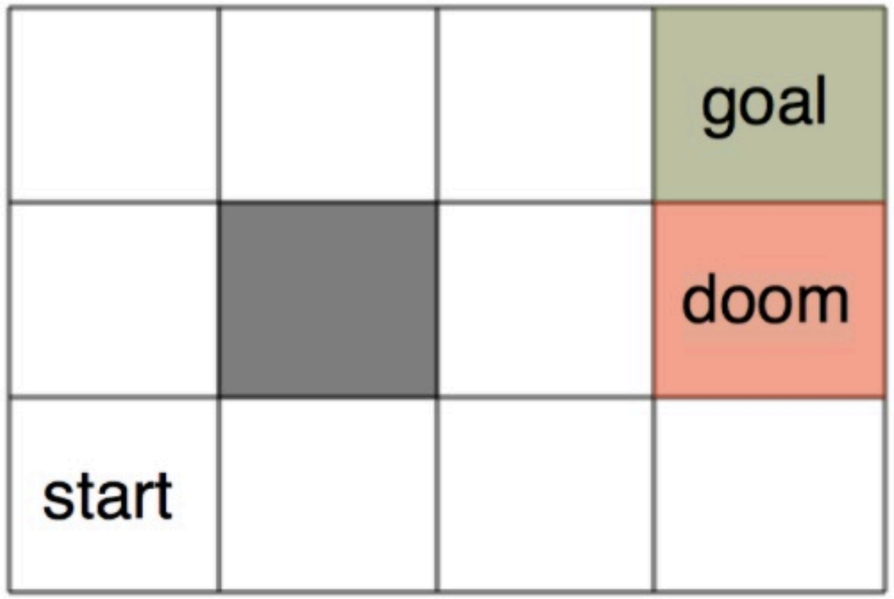
The Grid World with Randomness



Available Actions: Up, Down, Left, Right

Random Rules: $P(\text{action}) = 0.8$ and $P(\text{right angle}) = 0.1$

The Grid World with Randomness

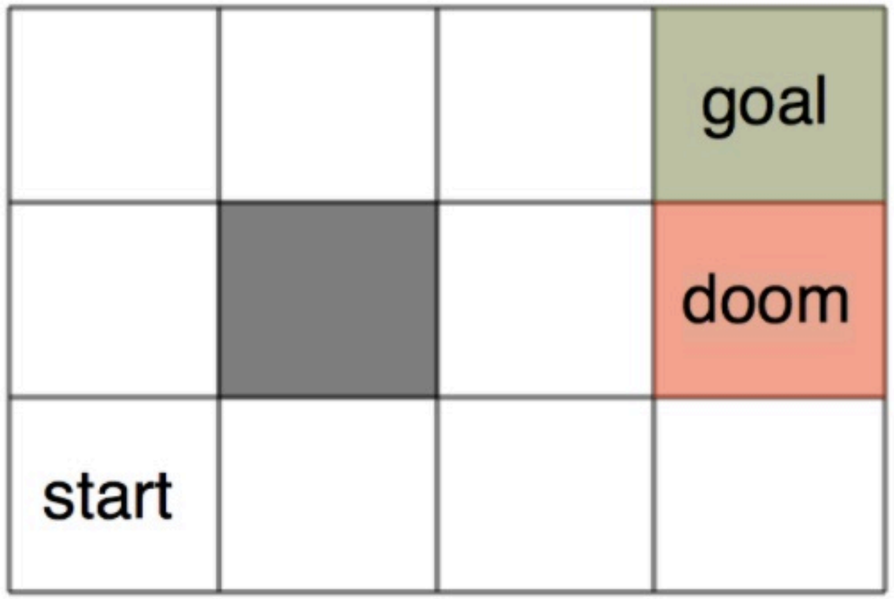


Available Actions: Up, Down, Left, Right

Random Rules: $P(\text{action}) = 0.8$ and $P(\text{right angle}) = 0.1$

Question: What is probability U, U, R, R, R gets to goal?

The Grid World with Randomness

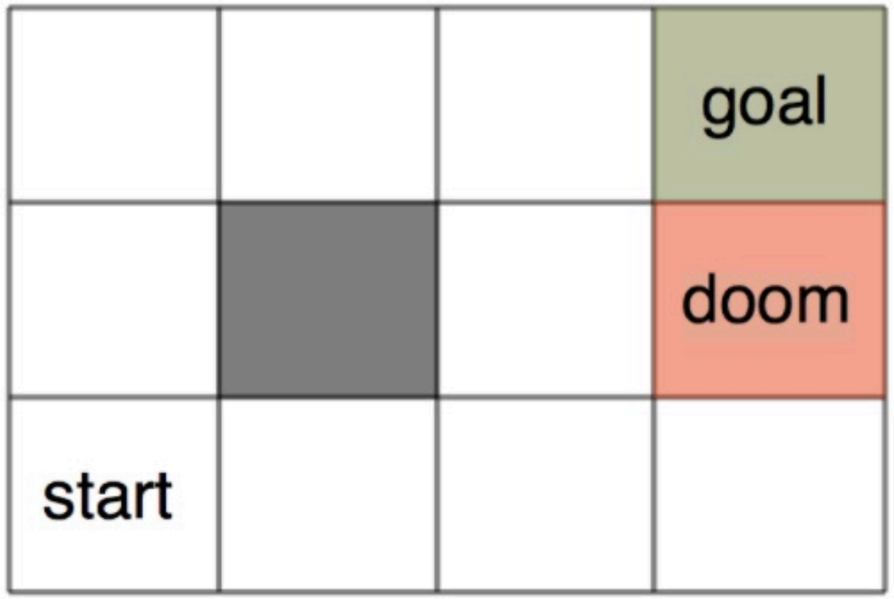


Random Rules: $P(\text{action}) = 0.8$ and $P(\text{right angle}) = 0.1$

Question: What is probability U, U, R, R, R gets to goal?

Answer: $(0.8)^5 = 0.32768$ (not quite right...)

The Grid World with Randomness



Random Rules: $P(\text{action}) = 0.8$ and $P(\text{right angle}) = 0.1$

Question: What is probability U, U, R, R, R gets to goal?

Answer: $(0.8)^5 + (0.1)^4 \times 0.8 = 0.32768 + 0.00008 = 0.32776$

Markov Decision Processes

States: S

Actions: A, or $A(s)$

Transition Model: $T(s, a, s') = P(s' \mid s, a)$

Markov Decision Processes

States: S

Actions: A, or $A(s)$

Transition Model: $T(s, a, s') = P(s' \mid s, a)$

Rewards: $R(s)$, $R(s, a)$, $R(s, a, s')$

Markov Decision Processes

States: S

Actions: A, or $A(s)$

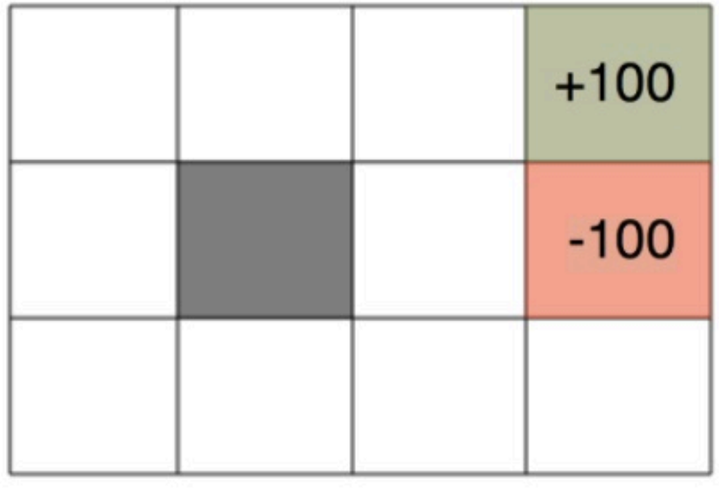
Transition Model: $T(s, a, s') = P(s' \mid s, a)$

Rewards: $R(s)$, $R(s, a)$, $R(s, a, s')$

Policy: $\pi(s) \rightarrow a$

Goal: Find **optimal** policy $\pi^\star$ that maximizes reward

More About Rewards

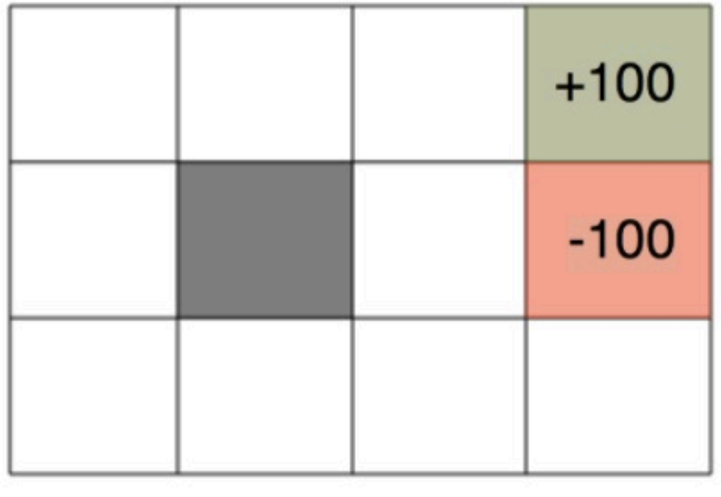


Rewards can be positive or negative

Delayed Reward: Might not get reward until you reach goal

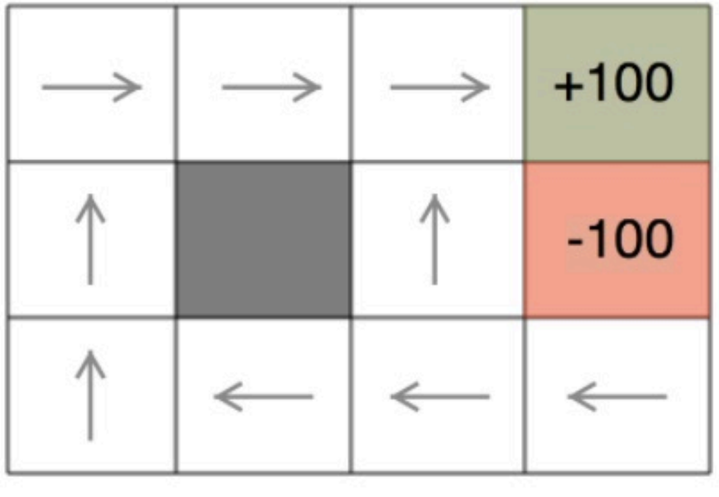
Might have **negative** reward until you reach goal

More About Rewards



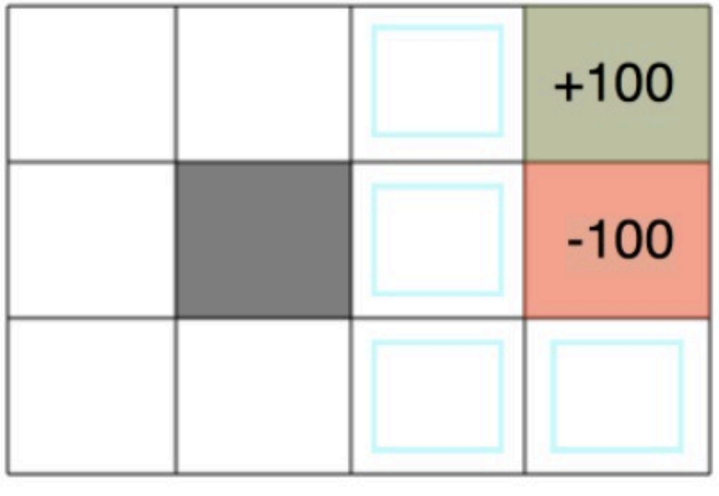
Example: What is the optimal policy if $R(s) = -5$?

More About Rewards



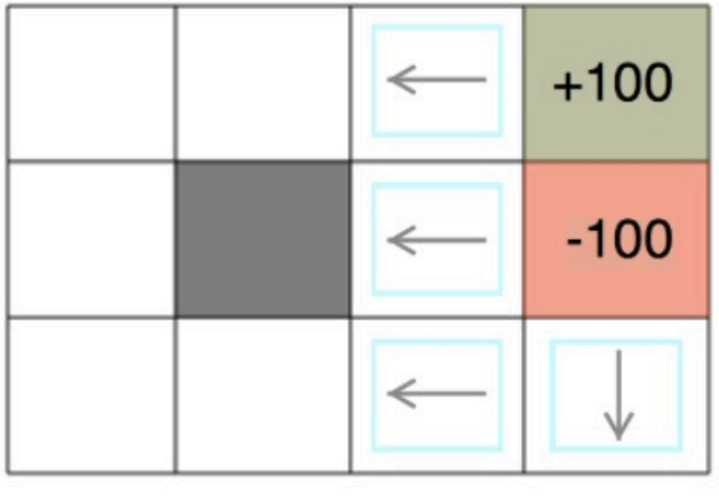
Example: What is the optimal policy if $R(s) = -5$?

More About Rewards



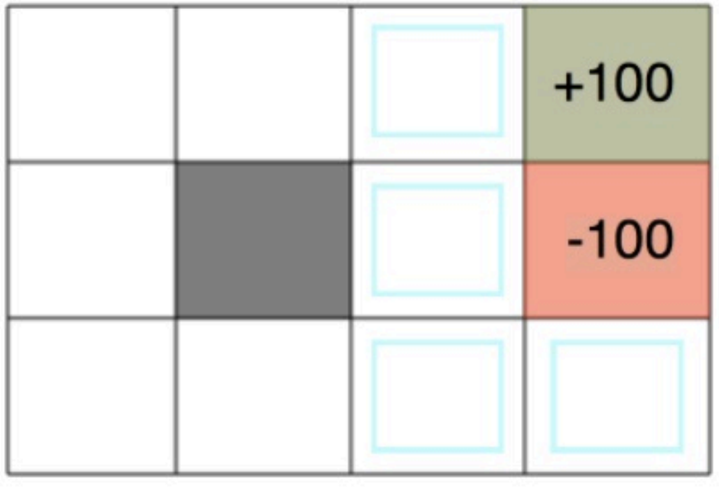
Example: What is the optimal policy if $R(s) = +200$?

More About Rewards



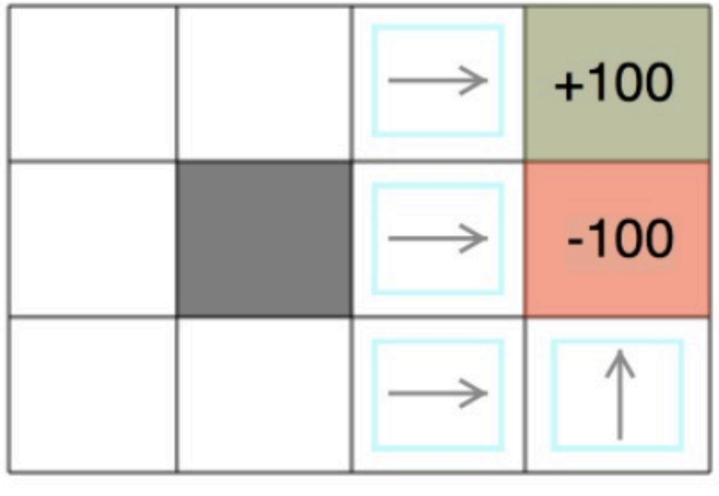
Example: What is the optimal policy if $R(s) = +200$?

More About Rewards



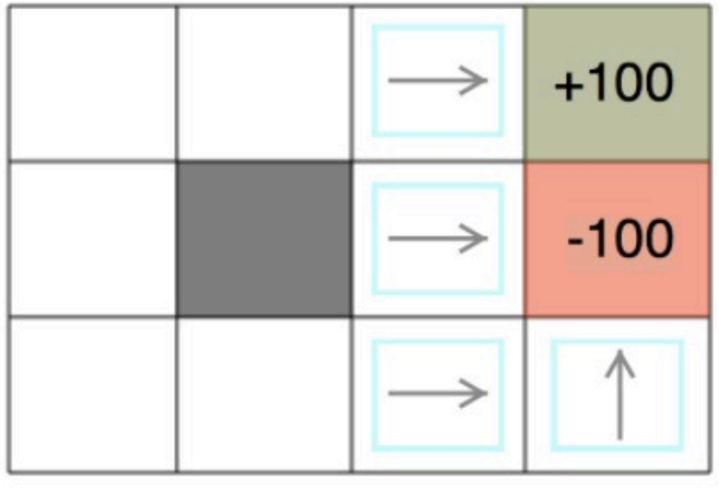
Example: What is the optimal policy if $R(s) = -200$?

More About Rewards



Example: What is the optimal policy if $R(s) = -200$?

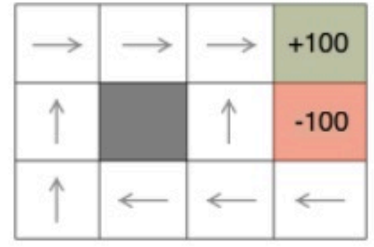
More About Rewards



Example: What is the optimal policy if $R(s) = -200$?

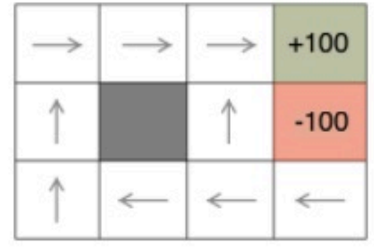
Take-Away: Optimal policy highly dependent on details of reward

Sequences of Rewards



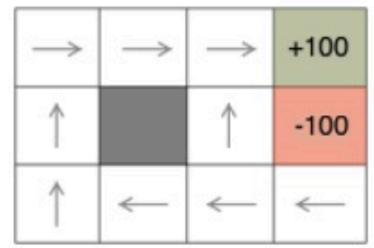
Assume infinite horizon

Sequences of Rewards



Assume infinite horizon ⇒ stationarity

Sequences of Rewards



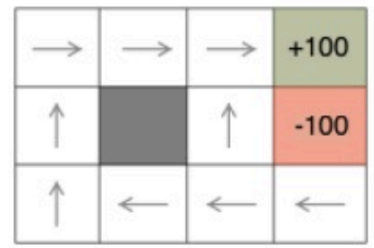
Assume infinite horizon ⇒ stationarity

Value or Utility of a Sequence

If $V(s_0, s_1, s_2, \ldots) > V(s_0, s'_1, s'_2, \ldots)$

Then $V(s_1, s_2, \ldots) > V(s'_1, s'_2, \ldots)$

Sequences of Rewards



Assume infinite horizon ⇒ stationarity

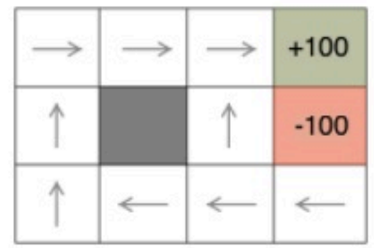
Value or Utility of a Sequence

If $V(s_0, s_1, s_2, \ldots) > V(s_0, s'_1, s'_2, \ldots)$

Then $V(s_1, s_2, \ldots) > V(s'_1, s'_2, \ldots)$

⇒ Stationarity of Preference

Sequences of Rewards



Assume infinite horizon $\Rightarrow$ stationarity

Value or Utility of a Sequence

If $V(s_0, s_1, s_2, \ldots) > V(s_0, s'_1, s'_2, \ldots)$

Then $V(s_1, s_2, \ldots) > V(s'_1, s'_2, \ldots)$

$\Rightarrow$ Stationarity of Preference

Question: What does this imply about form of V?

Sequences of Rewards

Assume for a second

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} R(s_t)$$

Sequences of Rewards

Assume for a second

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} R(s_t)$$

Which sequence of states/rewards is better?

$+1 \quad +1 \quad +1 \quad +1 \quad +1 \quad +1$

$+1 \quad +1 \quad +2 \quad +1 \quad +1 \quad +2$

Sequences of Rewards

Assume for a second

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} R(s_t)$$

Which sequence of states/rewards is better?

$+1 \quad +1 \quad +1 \quad +1 \quad +1 \quad +1$

$+1 \quad +1 \quad +2 \quad +1 \quad +1 \quad +2$

Under this definition of value it doesn't actually matter

Sequences of Rewards

Better model for value:

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \quad \text{for some } 0 \leq \gamma < 1$$

This is called **cumulative value** based on **discounted** reward

The parameter γ is the called the **discount factor**

The discount factor controls how long we're willing to wait for delayed rewards

Sequences of Rewards

Better model for value:

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \quad \text{for some } 0 \leq \gamma < 1$$

Notice

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq \sum_{t=0}^{\infty} \gamma^t R_{max} = \frac{R_{max}}{1-\gamma}$$

Sequences of Rewards

In case you've forgotten... this is just a **geometric series**

$$
\begin{aligned}
x &= \gamma^0 + \gamma^1 + \gamma^2 + \cdots \\
&= \gamma^0 + \gamma\left(\gamma^0 + \gamma^1 + \cdots\right) \\
&= 1 + \gamma x
\end{aligned}
$$

$$
x - \gamma x = 1 \qquad \Rightarrow \qquad x = \frac{1}{1-\gamma}
$$

Sequences of Rewards

Better model for value:

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \quad \text{for some } 0 \leq \gamma < 1$$

Notice

$$V(s_0, s_1, s_2, \ldots) = \sum_{t=0}^{\infty} \gamma^t R(s_t) \leq \sum_{t=0}^{\infty} \gamma^t R_{max} = \frac{R_{max}}{1-\gamma}$$

Punchline: Discounted reward renders an infinite horizon value function **finite**. Great b/c we can actually compare value of different sequences

Optimal Policies

The optimal policy $\pi^\star$ is the policy that maximizes the expected long-term discounted reward

$$\pi^\star = \text{argmax}_\pi \, \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \,\middle|\, \pi \right]$$

Optimal Policies

The optimal policy $\pi^\star$ is the policy that maximizes the expected long-term discounted reward

$$\pi^\star = \text{argmax}_\pi \, \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \,\middle|\, \pi \right]$$

Define the **cumulative value** of being in state s and following policy π in terms of the expected reward from starting in that state

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) \,\middle|\, \pi, s_0 = s \right]$$

Note that $V^\pi(s)$ is **not** $R(s)$

Optimal Policies

Can define *implicit* definition of $\pi^\star$ in terms of cumulative value

$$\pi^\star(s) = \text{argmax}_a \sum_{s'} P(s' \mid s, a)\, V^{\pi^\star}(s')$$

Unpack: The optimal policy at a state s is the action that maximizes the expected cumulative value

Simplify Notation: $V(s) \equiv V^{\pi^\star}(s)$

Another expression for cumulative value

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V(s')$$

Optimal Policies

This is called **Bellman's Equation**:

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V(s')$$

This allows us to solve the entire MDP

If we can find the optimal policy that maximizes the cumulative value at all states, then we're done

Actually Finding Optimal Policies

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V(s')$$

Suppose there are N total states in the system

Bellman's equation is really N equations in N unknowns

Should be easy!

Actually Finding Optimal Policies

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V(s')$$

Suppose there are N total states in the system

Bellman's equation is really N equations in N unknowns

Should be easy!

Except the $\max$ makes the system nonlinear

Nonlinear + Recursive definition ...

Need an **iterative** solution

Actually Finding Optimal Policies

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V(s')$$

Iterative Idea:

1. Start with arbitrary values for $V(s)$
2. Update V's based on neighbors
3. Repeat until values don't change anymore

Define the *approximation* of the value at iteration k as $\hat{V}_k(s)$

Update Rule: $\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$

Actually Finding Optimal Policies

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V(s')$$

Value Iteration:

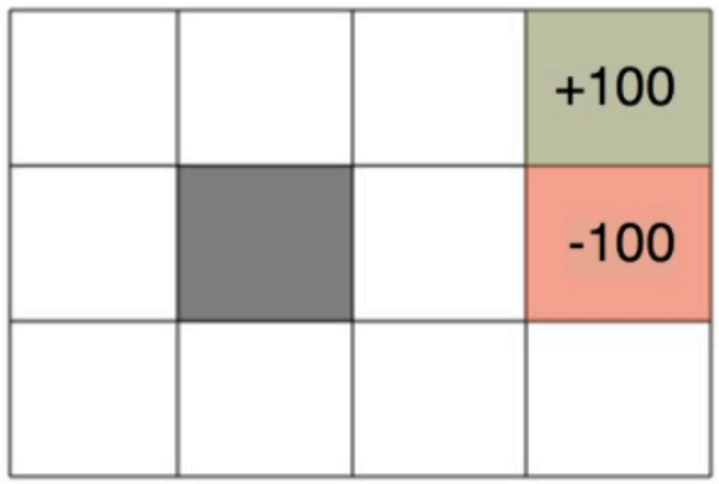
1. Start with arbitrary values for $V(s)$
2. Update V's based on neighbors
3. Repeat until values don't change anymore

Define the *approximation* of the value at iteration k as $\hat{V}_k(s)$

Update Rule: $\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

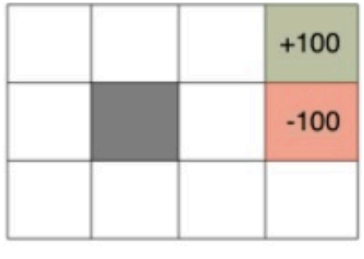


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

Question: What is $V_1(3, 3)$?

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

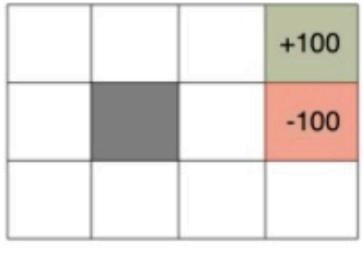


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

$$V_1(3,3) = -5 + \frac{1}{2} \max \left\{ \begin{array}{l} U: \\ D: \\ L: \\ R: \end{array} \right\}$$

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

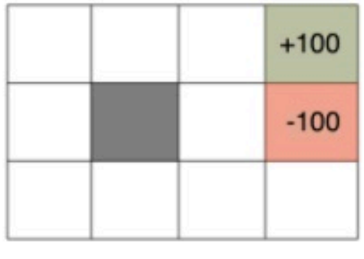


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

$$V_1(3,3) = -5 + \frac{1}{2} \max \left\{ \begin{array}{l} U : .8V_0(3,3) + .1V_0(4,3) + .1V_0(2,3) \\ D : .8V_0(3,2) + .1V_0(2,3) + .1V_0(4,3) \\ R : .8V_0(4,3) + .1V_0(3,2) + .1V_0(3,3) \\ L : .8V_0(2,3) + .1V_0(3,3) + .1V_0(3,2) \end{array} \right\}$$

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

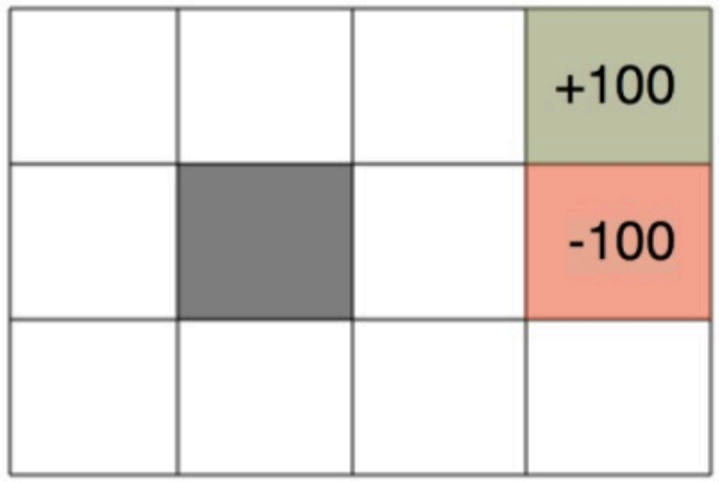


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

$$V_1(3,3) = -5 + \frac{1}{2}\max\left\{\begin{array}{l} U: .8\times 0 + .1\times 100 + .1\times 0 \\ D: .8\times 0 + .1\times 0 + .1\times 100 \\ R: .8\times 100 + .1\times 0 + .1\times 0 \\ L: .8\times 0 + .1\times 0 + .1\times 0 \end{array}\right\} = -5 + \frac{1}{2}80 = 35$$

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

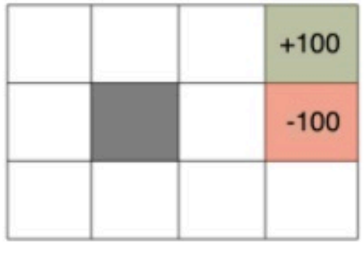


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

Question: What is $V_2(2,3)$?

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_{a} \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

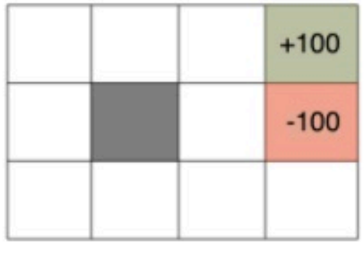


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

$$V_2(2,3) = -5 + \frac{1}{2} \max \left\{ \begin{array}{l} U: \\ D: \\ L: \\ R: \end{array} \right\}$$

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$

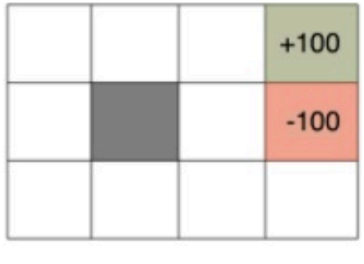


Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

$$V_2(2,3) = -5 + \frac{1}{2} \max \left\{ \begin{array}{l} U : .8V_1(2,3) + .1V_1(3,3) + .1V_1(1,3) \\ D : .8V_1(2,3) + .1V_1(1,3) + .1V_1(3,3) \\ R : .8V_1(3,3) + .1V_1(2,3) + .1V_1(2,3) \\ L : .8V_1(1,3) + .1V_1(2,3) + .1V_1(2,3) \end{array} \right\}$$

Value Iteration Example

$$\hat{V}_{k+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, \hat{V}_k(s')$$



Assume $\gamma = \frac{1}{2}$, $V_0(s) = 0$, $R(s) = -5$ (except absorbing states)

$$V_2(2,3) = -5 + \frac{1}{2} \max \left\{ \begin{array}{l} U : .8 \times -5 + .1 \times 35 + .1 \times -5 \\ D : .8 \times -5 + .1 \times -5 + .1 \times 35 \\ R : .8 \times 35 + .1 \times -5 + .1 \times -5 \\ L : .8 \times -5 + .1 \times -5 + .1 \times -5 \end{array} \right\} = -5 + \frac{1}{2} 27 = 8.5$$

Value Iteration

Once we've converged to $V^{\pi^\star}$, get optimal policy via

$$\pi^\star(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) V^{\pi^\star}(s')$$

- Guaranteed to converge to the optimal $V^{\pi^\star}$
- which gives us the optimal $\pi^\star$
- $\hat{V}_k$ might converge very slowly to $V^{\pi^\star}$
- but might converge quickly to $\pi^\star$

So far we've assumed complete knowledge of the MDP

Haven't really had to *learn* anything yet ...

Value Iteration

Once we've converged to $V^{\pi^\star}$, get optimal policy via

$$\pi^\star(s) = \underset{a \in A(s)}{\operatorname{argmax}} \sum_{s'} P(s' \mid s, a) V^{\pi^\star}(s')$$

- Guaranteed to converge to the optimal $V^{\pi^\star}$
- which gives us the optimal $\pi^\star$
- $\hat{V}_k$ might converge very slowly to $V^{\pi^\star}$
- but might converge quickly to $\pi^\star$

Next Time: What happens if we don't know model parameters

Next Time: Tweak Bellman Equations to get to Q-Learning

Acknowledgments

Many of these slides were adopted from the course *Reinforcement Learning* by Charles Isbell (Georgia Tech) and Michael Littman (Brown).

Many of these slides were adopted from Michael S. Lewicki (Carnegie Mellon)

The Grid World examples were adopted from *Artificial Intelligence: A Modern Approach* by Stuart Russell and Peter Norvig.

Some terminology/notation were adopted from the book *Machine Learning* by Tom Mitchell.

In Class

In Class

In Class

In Class

In Class

In Class