Markov Decision Process

- Components:
  - **States** s, beginning with initial state s_0
  - **Actions** a
    - Each state s has actions A(s) available from it
  - **Transition model** P(s' | s, a)
    - Markov assumption: the probability of going to s' from s depends only on s and a and not on any other past actions or states
  - **Reward function** R(s)
- **Policy** π(s): the action that an agent takes in any given state
  - The "solution" to an MDP

1

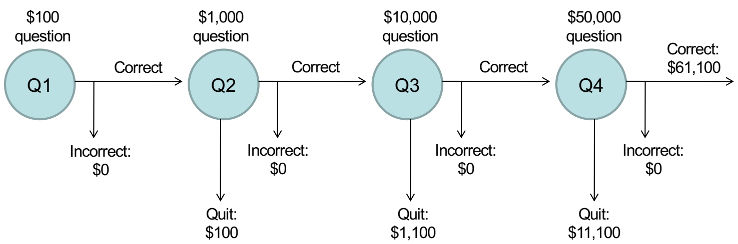
Overview

- First, we will look at how to "solve" MDPs, or find the optimal policy when the transition model and the reward function are known
- Next time, we will consider **reinforcement learning**, where we don't know the rules of the environment or the consequences of our actions

2

Example 1: Game show

- A series of questions with increasing level of difficulty and increasing payoff
- Decision: at each step, take your earnings and quit, or go for the next question
  - If you answer wrong, you lose everything

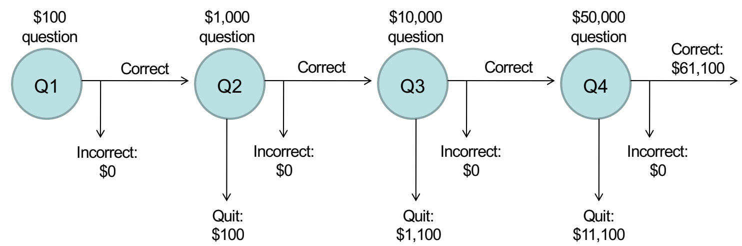


3

Example 1: Game show

- Consider $50,000 question
  - Probability of guessing correctly: 1/10
  - Quit or go for the question?
- What is the expected payoff for continuing?

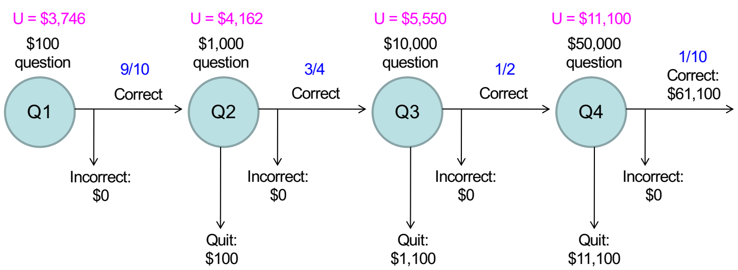
  0.1 * 61,100 + 0.9 * 0 = 6,110
- What is the optimal decision?



4

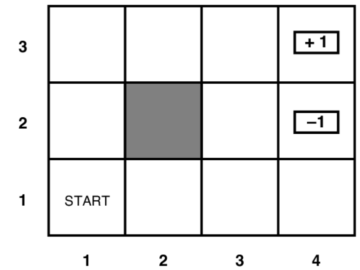
Example 1: Game show

- What should we do in Q3?
  - Payoff for quitting: $1,100
  - Payoff for continuing: 0.5 * $11,100 = $5,550
- What about Q2?
  - $100 for quitting vs. $4162 for continuing
- What about Q1?

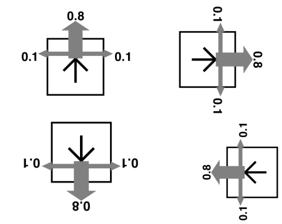


5

Example 2: Grid world



Transition model:



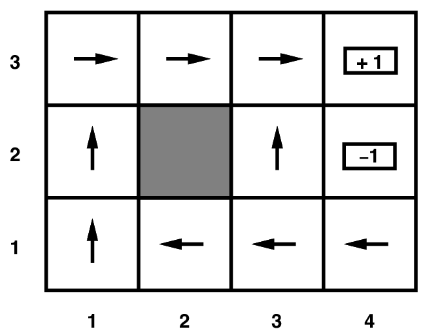
R(s) = -0.04 for every non-terminal state

Can the sequence [UP, Up, Right, Right, Right] take the agent in the terminal state?

Can the agent reach the goal in any other way?

6

Example 2: Grid world

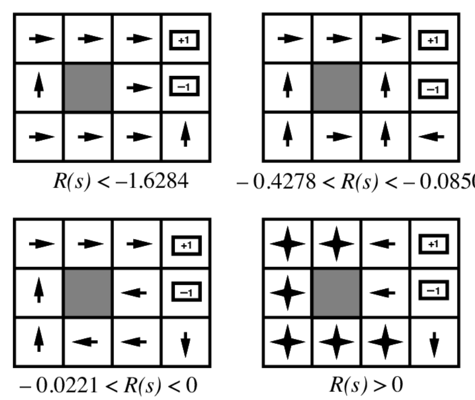


Optimal policy when R(s) = -0.04 for every non-terminal state

7

Example 2: Grid world

- Optimal policies for other values of R(s):



8

Markov Decision Process

- Components:
  - **States** s
  - **Actions** a
    - Each state s has actions A(s) available from it
  - **Transition model** P(s' | s, a)
    - Markov assumption: the probability of going to s' from s depends only on s and a, and not on any other past actions and states
  - **Reward function** R(s)
- The solution:
  - **Policy** π(s): mapping from states to actions

9

Partially observable Markov decision processes (POMDPs)

- Like MDPs, only state is not directly observable
  - States s
  - Actions a
  - Transition model P(s' | s, a)
  - Reward function R(s)
  - **Observation model P(e | s)**
- We will only deal with fully observable MDPs
  - Key question: given the definition of an MDP, how to compute the optimal policy?

10

Maximizing expected utility

- The optimal policy should maximize the *expected utility* over all possible state sequences produced by following that policy:

$$\sum_{\substack{\text{state sequences} \\ \text{starting from } s_0}} P(\text{sequence}) U(\text{sequence})$$

- How to define the utility of a state sequence?
  - Sum of rewards of individual states
  - Problem: infinite state sequences

11

Utilities of state sequences

- Normally, we would define the utility of a state sequence as the sum of the rewards of the individual states
- **Problem:** infinite state sequences
- **Solution:** *discount* the individual state rewards by a factor γ between 0 and 1:

$$U([s_0, s_1, s_2, \ldots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$$

  - Sooner rewards count more than later rewards
  - Makes sure the total utility stays bounded
  - Helps algorithms converge

12

Utilities of states

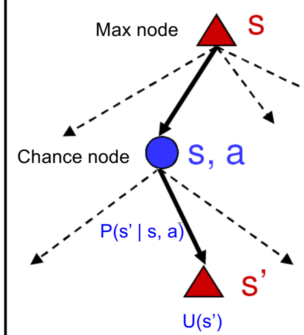
- Expected utility obtained by policy π starting in state s:

$$U^{\pi}(s) = \sum_{\substack{\text{state sequences} \\ \text{starting from } s}} P(\text{sequence})U(\text{sequence})$$

- The "true" utility of a state, denoted U(s), is the expected sum of discounted rewards if the agent executes an *optimal* policy starting in state s
- Reminiscent of minimax values of states…

13

Finding the utilities of states



- What is the expected utility of taking action **a** in state **s**?

$$\sum_{s'} P(s' \mid s, a)U(s')$$

- How do we choose the optimal action?

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

- What is the recursive expression for **U(s)** in terms of the utilities of its successor states?

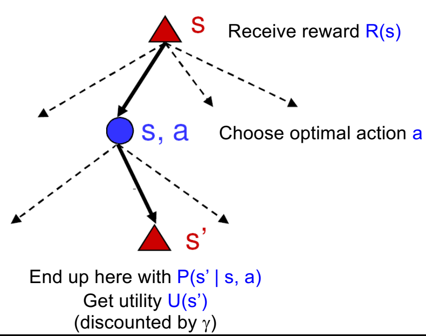
$$U(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)U(s')$$

14

The Bellman equation

- Recursive relationship between the utilities of successive states:

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



15

The Bellman equation

- Recursive relationship between the utilities of successive states:

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

- For N states, we get N equations in N unknowns
  - Solving them solves the MDP
  - We could try to solve them through expectimax search, but that would run into trouble with infinite sequences
  - Instead, we solve them algebraically
  - Two methods: **value iteration** and **policy iteration**

16

Method 1: Value iteration

- Start out with every U(s) = 0
- Iterate until convergence
  - During the ith iteration, update the utility of each state according to this rule:

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

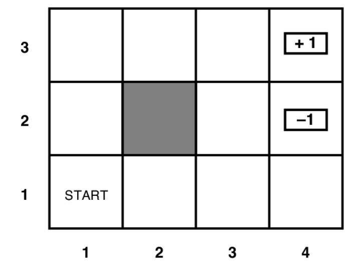
- In the limit of infinitely many iterations, guaranteed to find the correct utility values
  - In practice, don't need an infinite number of iterations…

17

Value iteration

- What effect does the update have?

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



18

Method 2: Policy iteration

- Start with some initial policy π_0 and alternate between the following steps:
  - **Policy evaluation:** calculate $U^{\pi_i}(s)$ for every state s
  - **Policy improvement:** calculate a new policy π_{i+1} based on the updated utilities

$$\pi^{i+1}(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a) U^{\pi_i}(s')$$

19

Policy evaluation

- Given a fixed policy π, calculate $U^{\pi}(s)$ for every state s
- The Bellman equation for the optimal policy:

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

  - How does it need to change if our policy is fixed?

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

  - Can solve a linear system to get all the utilities!
  - Alternatively, can apply the following update:

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

20