Optimal policy

- Reward structure for our example
- This is the policy that we obtain by applying Value Iteration to our example

\[ R(s) = \begin{cases} 
-0.04 & \text{(small penalty) for nonterminal states} \\
\pm1 & \text{for terminal states} 
\end{cases} \]
Rewards and Optimal Policy

Optimal Policy when reward in non-terminal states is $-0.04$

Is it possible that the optimal policy changes if the reward in the non-terminal states changes?

A. Yes

B. No
Rewards and Optimal Policy

If $r = -2$, what would be a reasonable policy?

A. 

B. 

Optimal Policy

$r = -0.4$
Rewards and Optimal Policy

Optimal Policy when $r < -1.6284$

Why is the agent heading straight into $(2,4)$ from its surrounding states?
Rewards and Optimal Policy

Optimal Policy when \(-0.427 < r < -0.085\)

The cost of taking a step is high enough to make the agent take the shortcut to \((3,4)\) from \((1,3)\)
Rewards and Optimal Policy

Optimal Policy when \(-0.0218 < r < 0\)

Why is the agent heading straight into the obstacle from (2,3)? And into the wall in (1,4)?

*see next slide* ....
Rewards and Optimal Policy

Optimal Policy when \(-0.0218 < r < 0\)

Stay longer in the grid is not penalized as much as before. The agent is willing to take longer routes to avoid (2,4)

- This is true even when it means banging against the obstacle a few times when moving from (2,3)
Rewards and Optimal Policy

Optimal Policy when \( r > 0 \)?

Which means the agent is rewarded for every step it takes
Rewards and Optimal Policy

Optimal Policy when \( r > 0 \)

Which means the agent is rewarded for every step it takes.

It avoids terminal states completely.

State where every action belongs to an optimal policy.

\( r > 0 \)
MDPs scalability (not required)

- Modern optimal algorithms draw from a vast repertoire of techniques, like graph algorithms, heuristic search, compact value function representations, and simulation-based approaches. E.g.,
  - Only compute $V$ for states “reachable” from $S_0$
  - Do not compute $V$ for really bad states (based on heuristics)

- An enormous number of approximation algorithms have been suggested that exploit several intuitions, such as inadmissible heuristics, interleaving planning and execution, special processing for dead-end states, domain determinization ideas, hybridizing multiple algorithms, and hierarchical problem decompositions.
Markov Models

- Markov Chains
- Hidden Markov Model
- Partially Observable Markov Decision Processes (POMDPs)
- Markov Decision Processes (MDPs)
Lecture Overview

Filtering for HMM (more when we will do temporal models)

Partially Observable Markov Decision Processes

• Formal Specification and example
  • Belief State
  • Belief State Update
Hidden Markov Model

- A Hidden Markov Model (HMM) starts with a Markov chain, and adds a noisy observation/evidence about the state at each time step:

- $P(X_0)$ specifies initial conditions

- $P(X_{t+1}|X_t)$ specifies the dynamics

- $P(E_t|S_t)$ specifies the sensor model

- $|\text{domain}(X)| = k$

- $|\text{domain}(E)| = h$
Hidden Markov Model (our example with no actions)

- \( E = \# \text{ of walls} \ \{1w, 2w\} \)

- \( P(X_0) \) specifies initial conditions

- \( P(X_{t+1} | X_t) \) specifies the dynamics
  \[ 11 \times 11 \]

- \( P(E_t | S_t) \) specifies the sensor model

\[ |\text{domain}(X)| = 11 \]

\[ |\text{domain}(E)| = 2 \]
Useful inference in HMMs

• In general (Filtering): compute the posterior distribution over the current state given all evidence to date

\[ P(X_t | e_{0:t}) \]
Intuitive Explanation for filtering recursive formula

$P(X_t | e_{0:t}) =$

sequence of evidences $e_0 : e_t$
Intuitive Explanation for filtering recursive formula

\[ P(X_t | e_{0:t}) = \alpha \cdot P(e_t | X_t) \cdot \sum_{X_{t-1}} P(X_t | X_{t-1}) \cdot P(X_{t-1} | e_0 : e_{t-1}) \]

- \( X_t \) generated evidence \( e_t \)
- whatever \( X_{t-1} \) was, \( X_t \) was reached from there
- and evidence \( e_0 : e_{t-1} \) must have been generated before getting to \( X_{t-1} \)
Lecture Overview

Filtering for HMM (more when we will do temporal models)

Partially Observable MDPs

- Formal Specification and example
  - Belief State
  - Belief State Update
POMDP: Intro

The MDPs we looked at so far were \textit{fully observable}

- The agent \textit{always knows which state it is in}
- The uncertainty is in \ldots\ldots\ldots?

- Policy only depends on\ldots\ldots\ldots?
Belief States

In POMDPs, the agent cannot tell for sure where it is in the space state, all it can have are *beliefs* on that

- *probability distribution over states*
- This is usually called *belief state* \( b \)
- \( b(s) \) is the probability assigned by \( b \) to the agent being in state \( s \)

**Example**: Suppose we are in our usual grid world, but

- the agent has no information at all about its position in non-terminal states
- It knows only when it is in a terminal state (because the game ends)

What is the initial belief state, if the agent knows that it is not in a terminal state?
Belief States

- Initial belief state:
  - $<1/9,1/9, 1/9,1/9,1/9,1/9, 1/9,1/9,1/9,0,0>$
Observation Model

- As in HMM, the agent can learn something about its actual state by *sensing* the environment:
  - **Sensor Model** $P(e|s)$: probability of observing the evidence $e$ in state $s$

- A POMDP is fully specified by
  - Reward function: $R(s)$ (we’ll forget about $a$ and $s'$ for simplicity)
  - Transition Model: $P(s'|a,s)$
  - Observation model: $P(e|s)$

- Agent’s belief state is updated by computing the conditional probability distribution over all the states given the sequence of observations and actions so far
State Belief Update

We just saw filtering for HMM?

- Compute conditional probability distribution over states at time t given all observations so far

\[ P(X_t | e_{0:t}) = \alpha P(e_t | X_t) \sum_{x_{t-1}} P(X_t | x_{t-1}) P(x_{t-1} | e_{0:t-1}) \]

Filtering at time \( t-1 \)

Inclusion of new evidence (sensor model)

Propagation to time \( t \)

State belief update is similar but includes actions

- If the agent has current belief state \( b(s) \), performs action \( a \) and then perceives evidence \( e \), the new belief state \( b'(s') \) is

\[ b'(s') = \alpha P(e | s') \sum_s P(s' | a, s) b(s) \]

Inclusion of new evidence: Probability of perceiving \( e \) in \( s' \)

Sum over all the states that can take to \( s' \) after performing \( a \)

Filtering at time \( t-1 \): State belief based on all observations and actions up to \( t-1 \)

Propagation at time \( t \): Probability of transition to \( s' \) given \( s \) and \( a \)
Grid World Actions Reminder

Agent moves in the above grid via actions *Up, Down, Left, Right*

Each action has:

- 0.8 probability to reach its intended effect
- 0.1 probability to move at right angles of the intended direction
- If the agent bumps into a wall, it says there
Example (no observation)

- Back to the grid world, what is the belief state after agent performs action \textit{left} in the initial situation?
- The agent has no information about its position
  - Only one fictitious observation: \textit{no observation}
  - \( P(\text{no observation} \mid s) = 1 \) for every \( s \)
- Let’s instantiate \( b'(s') = \alpha P(e \mid s') \sum_s P(s' \mid a, s)b(s) \)
- For state \((1,1)\) (action \(a = \text{left}\))

\[
b'(1,1) = \alpha \left[ P((1,1) \mid (1,1), \text{left})b(1,1) + P((1,1) \mid (2,1), \text{left})b(2,1) + \ldots \right]
\]

What is missing to get the correct answer?

A. \( P((1,1) \mid (1,2), \text{down})b(1,2) \)
B. \( P((1,1) \mid (1,3), \text{left})b(1,3) \)
C. \( P((1,1) \mid (1,2), \text{left})b(1,2) \)
Example

- Back to the grid world, what is the belief state after agent performs action left in the initial situation?

- The agent has no information about its position
  - Only one fictitious observation: no observation
  - \( P(\text{no observation} \mid s) = 1 \) for every \( s \)

- Let’s instantiate \( b'(s') = \alpha P(e \mid s') \sum_s P(s' \mid a, s)b(s) \)

\[
b'(1,1) = \alpha \left[ P((1,1) \mid (1,1), \text{left})b(1,1) + P((1,1) \mid (1,2), \text{left})b(1,2) + P((1,1) \mid (2,1), \text{left})b(2,1) \right]
\]

\[
b'(1,2) = \alpha \left[ P((1,2) \mid (1,1), \text{left})b(1,1) + P((1,2) \mid (1,2), \text{left})b(1,2) + P((1,2) \mid (1,3), \text{left})b(1,3) \right]
\]

- Do the above for every state to get the new belief state
After five *Left* actions
Example

Let’s introduce a sensor that perceives the number of adjacent walls in a location with a 0.1 probability of error

- \( P(2w|s) = 0.9 \); \( P(1w|s) = 0.1 \) if \( s \) is non-terminal and not in third column
- \( P(1w|s) = 0.9 \); \( P(2w|s) = 0.1 \) if \( s \) is non-terminal and in third column

Try to compute the new belief state if agent moves left and then perceives 1 adjacent wall

\[
b'(s') = \alpha P(e | s') \sum_{s} P(s'|a,s)b(s)
\]

\[
b'(1,1) = \alpha X \left[ P((1,1)| (1,1), left)b(1,1) + P((1,1)| (1,2), left)b(1,2) + P((1,1)| (2,1), left)b(2,1) \right]
\]

\( X \) should be equal to?

A. 0.1  B. 0.2  C. 0.9
Learning Goals for today’s class

You can:

• Define and compute **filtering** on an HMM
• Define a **POMDP**
• Define and compute a **state belief update** for a **POMDP**
• Define a **Policy** for a POMDP
TODO for Wed

Read Textbook 9.5.6 Partially Observable MDPs

Check what to do with readings (details on course webpage)
- Carefully read the paper before class
- Send by email
  - (at least 3) questions on the assigned paper
  - a brief summary of the paper (no more than half a page)
  - First Wed 28

Assignment 1 will be out on Wed
Partially Observable Markov Decision Process (POMDP): As the name suggests, POMDPs model scenarios where the agent cannot observe the world state fully [123]. A POMDP agent needs to execute actions for two reasons: for changing the world state (as in an MDP) and for obtaining additional information about the current world state. As Section 7.1.1 explains, a POMDP is a large Continuous MDP, in which a state-variable is the world state, and its value denotes the agent’s belief (probability) that it is in that state. Straightforward implementations of MDP algorithms do not scale up to POMDPs and, over the years, a large number of specialized POMDP techniques have been developed, with successes in scaling the algorithms to millions of states [214]. POMDPs have also seen several applications, e.g., dialog management [241], intelligent control of workflows [65], intelligent tutoring [200], and several robotic planning applications [233].