

L15 – Discrete partially observable domains

AIMA 4.4, 17.4; KAlg 20.1-3; 21.1-2, 22.4-5

What you should know

- How to plan in belief space with set-based uncertainty
- How to model planning problems with a POMDP
- The Belief Space MDP
- Expectimax for partially observable problems
- Most-likely state and most-likely observation deterministic approximations

Non-observability

What if the robot does not know what state it is in and can't gain any information? We'll call this UOND (un-observable non-deterministic) planning.

- Could be uncertainty about the initial state
- Could be additional non-determinism in transitions that increase uncertainty
- If no possibility for observations, then we can try to find a conformant plan, which is guaranteed to succeed for all possible initial states and non-deterministic outcomes
- Do this through state-space search where now our states are belief states, which are sets of original states.

Vacuum world doesn't (always) suck!

Example environment from AIMA 4.4:

- Robot in grid world has a vacuum cleaner
- Each cell in grid has dirt or not
- Robot can suck (with vacuum) or move left, right, up, or down
- Want the room to be completely clean

If completely observable and deterministic, then we can solve easily using min-cost-path search. But what if:

- The robot doesn't know what room it's in
- The robot doesn't know which rooms are dirty
- Its actions are deterministic

And it wants to clean the whole environment

Reducing UOND planning to belief-space search

Given a non-observable-path problem $(\mathcal{S}, \mathcal{A}, \tilde{T}, G, C, S_0)$ where \tilde{T} is non-deterministic as in FOND and $S_0 \subset \mathcal{S}$ is the set of possible starting states, we can generate a standard min-cost-path problem $(\mathcal{B}, \mathcal{A}', T', G', C, b_0)$ so that solution to the min-cost-path problem is a solution to the original non-observable problem:

- $\mathcal{B} = \text{powerset}(\mathcal{S})$ // Set of subsets of \mathcal{S}
- $\mathcal{A}' = \mathcal{A}$
- $b_0 = S_0$ // Single belief state is a set of env states
- $T'(b, a) = \bigcup_{s \in b} T(s, a)$
- $G' = \{b \mid b \subseteq G\} = \text{powerset}(G)$
- $C'(b, a, b')$ can really only depend on a

Important to note that even though transition on states \tilde{T} is non-deterministic, the belief-space transition function T' is deterministic.

Adding observations

If we can make some observations, then we get the POND (partially observable non-deterministic) planning setting.

- Finite observation set \mathcal{O}
- perception function $O : \mathcal{S} \rightarrow \mathcal{O}$

It tells us what we will see in each state. For example, a vacuum robot with a local dirt detector might have $\mathcal{O} = \{clean, dirty\}$ and perception function telling it about its current location.

Includes completely observable and completely unobservable cases.
How?

Use it to do a belief update:

- Given current belief (set of possible states) S and an actual observation o , what should we believe?
- Rule out states that could not have generated o :
- $UPDATE(b, o) = \{s \in b \mid O(s) = o\}$

Searching in PO environments

And/Or search in belief space! Have to do **and** branches on the possible observations. Given non-deterministic, partially observable problem $(\mathcal{S}, \mathcal{A}, \mathcal{O}, S_0, \tilde{T}, O, G, C)$ we can generate a non-deterministic observable problem (solvable by AO search):

- $\mathcal{B} = \text{powerset}(\mathcal{S})$ // Set of subsets of \mathcal{S}
- $\mathcal{A}' = \mathcal{A}$
- $b_0 = S_0$
- $\tilde{T}'(b, a) = \{\text{UPDATE}(\tilde{T}(b, a), o) \mid o \in \text{POSSIBLE-PERCEPTS}(T(b, a))\}$
- $G' = \{b \mid b \subseteq G\} = \text{powerset}(G)$
- $C(b, a, b')$ can really only depend on a

What observations could we possibly get in belief state b ?

$$\text{POSSIBLE-PERCEPTS}(b) = \{O(s) \mid s \in b\}$$

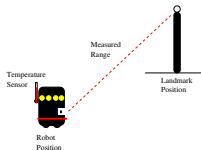
Agent for partially observable environment

- Initial belief $b = b_0$
- Make AO plan
- Take action a at root node
- Receive observation o from environment
- Update $b = \text{UPDATE}(\tilde{T}(b, a), o)$
- Follow o branch in plan to get next action
- ...

Idea for later: In receding-horizon control, you can get away with making an approximate plan, and then replanning after every belief update.

Estimation and Planning

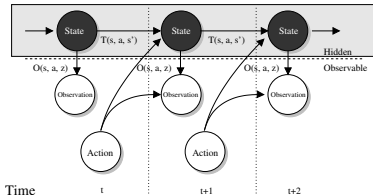
- Recall our motivating problem of a robot estimating its position.



- Things we know how to do:
 - Compute a distribution over the robot's position at time t , $P(x_t|z_t, a_t)$ based on a history of actions and observations, using message passing
 - Compute the robot's most likely trajectory from time 0 to t , $\hat{x}_{0:t} = \operatorname{argmax}_{x_{0:t}} P(x_{0:t}|z_{0:t}, a_{0:t})$ based on a history of actions and observations, using Viterbi / max-sum
- What if the estimate of the robot's position is really noisy, so that \hat{x}_t isn't particularly high probability?
- What if we had an action, such as "look at a map", that doesn't affect \hat{x}_t but has a lot of impact on $P(x_t|z_{0:t}, a_{0:t})$?

Sequential decision-making with state uncertainty

- We are uncertain about the initial state
- Our goal remains to choose actions that maximize $E_{s_0:T} [\sum_t \gamma^t r(s_t)]$
- Since we don't know the state, future states and observations are not independent of past observations.



- Choice of action seems to depend on whole history of actions and observations
- We have to reason about the future observation sequences
- When the true state of the world cannot be observed directly (or is "inaccessible"), the world is "Partially Observable."

Partially Observed MDPs

MDP with added observation process

- MDP has
 - a set of states \mathcal{S}
 - a set of actions \mathcal{A}
 - transition model $T(s, a, s') = P(S_{t+1} = s' \mid S_t = s, A_t = a)$
 - reward function $R : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$
 - discount factor γ (and possibly horizon T)
- POMDP adds
 - a set of observations \mathcal{O}
 - observation model
$$O(s', a, o) = P(O_{t+1} = o \mid A_t = a, S_{t+1} = s')$$
- optionally
 - an initial distribution over states, $b_0 = P(s_0)$

Belief states

The key to (somewhat) efficient action selection in POMDPs is the idea of a belief state.

- A belief state (or belief) b is a probability distribution over states
- Initial belief $b_0 = P(s_0)$ represents the agent's knowledge about the state when it starts.
- We define $b_t = P(s_t \mid a_0, o_0, \dots, a_{t-1}, o_{t-1})$
- Computing b_t is a filtering problem.
- **Theorem:** b_t separates future states and observations from past actions and observations.
- So, if you know b_t you don't need to remember (or consider during decision-making) previous actions and observations.
- A policy for a POMDP can be described as a mapping from belief states to actions.

Recall Bayes filter belief update

Given previous belief b , action a and observation o , what is the new belief $b' = \text{bf}(b, a, o)$?

$$\begin{aligned}\text{bf}(b, a, o)(s') &= P(S_{t+1} = s' \mid A_t = a, O_{t+1} = o, B_t = b) \\ &= \frac{1}{\eta} P(O_{t+1} = o \mid S_{t+1} = s', A_t = a, B_t = b) \cdot \\ &\quad P(S_{t+1} = s' \mid B_t = b, A_t = a) \\ &= \frac{1}{\eta} O(s', a, o) \sum_s P(S_{t+1} = s' \mid B_t = b, A_t = a, S_t = s) \cdot \\ &\quad P(S_t = s \mid B_t = b, A_t = a) \\ &= \frac{1}{\eta} O(s', a, o) \sum_s T(s, a, s') b(s)\end{aligned}$$

where $\eta = P(o \mid a, b) = \sum_{\tilde{s}} O(\tilde{s}, a, o) \sum_s T(s, a, \tilde{s}) b(s)$

The Belief MDP

The process characterizing the way the beliefs change over time in a POMDP, is actually an MDP in belief space! Whoa!

- $S' = \mathcal{B}$: In POMDP with n underlying world states, a belief $b \in \mathcal{B}$ is (p_1, \dots, p_n) with $p_i = P(s_i)$, $0 \leq p_i$ and $\sum_i p_i = 1$
- \mathcal{A}' : Same as the POMDP
- $R' : \mathcal{B} \times \mathcal{A} \rightarrow \mathbb{R}$ where

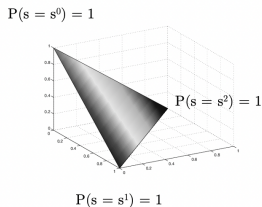
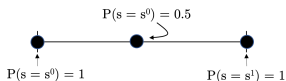
$$R'(b, a) = \sum_s R(s, a)b(s)$$

- $T' : \mathcal{B} \times \mathcal{A} \rightarrow \mathcal{B}$ where

$$T'(b, a, b') = \sum_{\{o | b' = bf(b, a, o)\}} P(o | b, a)$$

$$\text{where } P(o | b, a) = \sum_s P(o | s, a)b(s) = \sum_s b(s) \sum_{s'} O(s', a, o)T(s, a, s')$$

What exactly is Belief Space?



A simplex: the convex hull of the distributions where one of the states has probability one.

- For a two state MDP, this simplex is the line from 0 to 1.
 - At one end, the belief is that one of the states has probability 1.
 - At the other end, the belief is that the other state has probability 1.
 - All the points in between correspond to beliefs where the MDP might be in one of the two states, with different probabilities.
- For a three state MDP, the simplex is a 2D triangle embedded in 3D.
 - Each corner of the triangle corresponds to a belief where one of the states has probability 1.

Formulation: Tiger domain

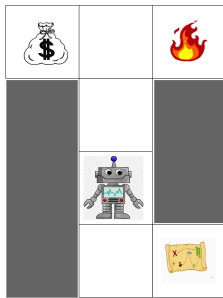
- Agent doesn't know which door the tiger lurks behind: prior is 50/50
- Agent can either listen to the tiger roar (-1), or open a door and escape (+10) or get eaten (-100).
- Listening is accurate with 85%.
- Tiger doesn't move until after we open a door, and escape or are eaten (world is stochastic).
 - What is the state space? What is $P(s_0)$?
 - Suppose we listen once, and hear the tiger roar behind the left door. What is $P(s_1)$? What action should we take next?



Problem forces the tradeoff between listening (gathering more information), or trusting the estimate

Formulation: Fortune or Ruin

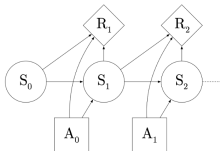
- World is deterministic.
 - Agent doesn't initially know which side of the T junction has +100 reward or -100 reward.
 - Map at the bottom of the L describes which side has which reward with probability 1
- What is the state space?
 - What is $P(s_0)$?
 - Suppose the robot moves down one space. What is $P(s_t)$? What action should the robot take next?



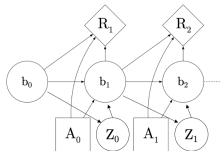
Problem forces the robot to plan to gather information.

Selecting actions in a belief-space MDP

Influence diagrams for an MDP and for a belief MDP:



Influence diagram for a Markov Decision Process (MDP)



Influence diagram for a belief state MDP

- The good news: can define and execute policies easily with respect to b_t
- The bad news: computing optimal policies exactly is very hard: will see that the worst-case computational cost is doubly-exponential in the horizon and $|O|$
- Two solution strategies:
 - On-line: expectimax, MCTS, approximations
 - Off-line: value-iteration, approximations

Online POMDP solution methods

Because the belief-space process is an MDP, we can use online MDP methods:

- Expectimax search:
 - Start search at current belief b_0
 - Branch on our choice of actions a
 - Branch on nature's choice of outcomes, which in a POMDP, is o
 - Result of doing a and observing o is $b' = \text{bf}(b, a, o)$ with probability $\sum_s b(s) \sum_{s'} O(s', a, o) T(s, a, s')$
- Sparse sampling (expectimax with samples over o)
- Monte-Carlo Tree Search
- Determinization and other approximations: we outline a few choices

Very simple (bad) approximation: Most likely state

First, solve the underlying MDP to get π^{MDP} . Then

$$\pi^{\text{POMDP}}(\mathbf{b}) = \pi^{\text{MDP}}(\underset{s}{\text{argmax}} \mathbf{b}(s))$$

where π^{POMDP} is the policy of the POMDP, and π^{MDP} is the policy of the corresponding MDP that assumes the state is fully observable.

What's wrong with this approach?

- Can't tell the difference between beliefs where the maximum-likelihood state is really certain (and just assuming it is the true state is reasonable) and where beliefs are really uncertain, and the true state could be anything.
- How would this approximation perform on Tiger?
- How would this approximation perform on Fortune-or-Ruin?
- **Key idea:** The uncertainty of the belief matters for the optimal policy. (We'll need a way to characterise the uncertainty.)
- Actually provably optimal for linear-quadratic-Gaussian systems.

Better approximation: QMDP

Assume the state will become fully observable on the next step. Solve the underlying MDP to get Q^{MDP} . Then

$$\pi(\mathbf{b}) = \operatorname{argmax}_{\mathbf{a}} \sum_s \mathbf{b}(s) Q^{\text{MDP}}(s, \mathbf{a})$$

where Q^{MDP} is the Q-function of the corresponding MDP that assumes the state is fully observable.

- Takes into uncertainty into account for one step, but only one step, by hedging
- If the belief has high uncertainty, but there is some action that is not too bad across all the possible states, then that action will be preferred to some other action that is high risk/reward.
- What's wrong with this approximation? Tiger? Fortune?
- Hedging against uncertainty for one step may not be enough
- **Key idea:** The optimal policy may need to perform information gathering that incurs cost in order to resolve state uncertainty and ensure the agent can get large rewards (e.g., get to the goal with confidence).

Sometimes better approximation: Most-likely Observation

This time, we will turn our problem into a deterministic reward-maximization search problem in belief space.

- $S' = \mathcal{B}$
- $s'_0 = b_0$
- $R'(b, a, b') = R(b, a, b')$
- $T'(b, a) = \text{bf}(b, a, o^*)$ where $o^* = \text{argmax}_o P(o | b, a)$

This works very nicely for a POMDP version of a SSP problem, where instead of having an R' , we can let the belief-space goal be something like believing with high probability that we are in a goal state.

$$G' = \left\{ b \mid \sum_{s \in G} b(s) > .9 \right\}$$

MLO: strengths and weaknesses

- Requires replanning if ML observation is not obtained (or if resulting b is substantially different from the expected one)
- How would it work on Tiger or Fortune-or-ruin?
- How would this approach do on a linear-Gaussian MDP with continuous states?
 - Surprising result: the mean of this belief state is locally controllable, and the variance is deterministic.
 - Can plan directly in the configuration space, predict what the full beliefs will be, and compute the belief state costs.
 - Leads to very efficient planning performance.