Paper Review

Title: ACTING OPTIMALLY IN PARTIAL OBSERVABLE STOCHASTIC DOMAINS

Artificial Intelligence – 2

Indian Institute of Technology, Jodhpur

Author:
Shristy Gupta (M20CS015)

Date: March 30, 2021

Abstract

In partially observable stochastic environments, there are many ways to find a near optimal control strategy and one such strategy is partially observable Markov decision process (POMDP).
POMDB is efficient enough to solve real world sequential decision problems. This process is chosen keeping in mind that the agent cannot know the underlining state.
This decision framework provides optimal action over the state for all beliefs. Hence for agent to interact with the environment, it needs to take set of optimal actions aka optimal policy. This is also necessary as the optimal action eventually the expected reward over infinite horizon.

Introduction

Agent has to make a decision without knowing the true state of the environment. Although this problem has been previously addressed by the AI community some of the examples are formalisms of epistemic logic by knowing the preconditions of the environment and the effects on their planners. The issue is that in these processes the environment is assumed to be deterministic.
POMDB addresses a probabilistic environment state and the optimal path is finally chosen based on actions as they enhance the agent's estimate.

Discussion

POMDB handles two kinds of uncertainty, first uncertainty of world state due to partial information and second uncertainty about outcome of an action.
From the below diagram,

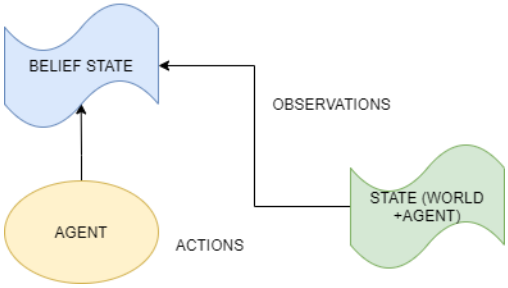

Figure 1 POMDB flow chart

Goal becomes selections of appropriate actions
It has 7 tuples:
A: Set of actions
S: set of States
T: Set of conditional transition probabilities between the state
R: Set if reward function also defined as S X A → R
Ω: set of observations
Ó: set of conditional observation policies
ɣ: Discount factor whose range varies from 0 to 1

Every time following process happens:

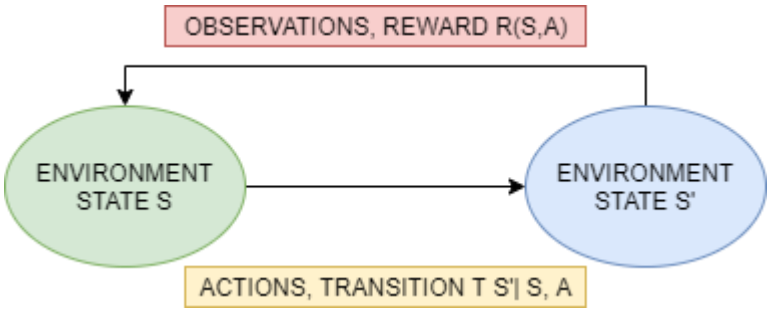

Figure 2 Rewards and observation in POMDB state space

Based on the reward the agent will take the action whose cumulative reward is highest. Also, this cumulative reward is dependent on Discount factor. The discount factor tells how much current reward of state is valued over distant rewards.
There are several concepts discussed in the paper:

Belief state:

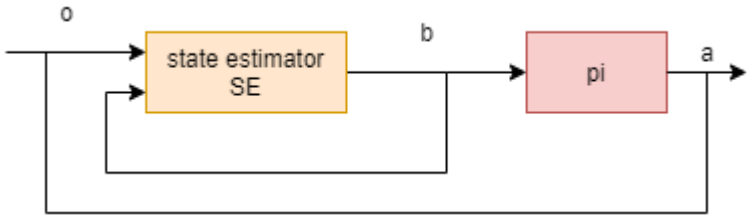

Figure 3 State space diagram

The above figure describes that the state estimator SE takes the input from last belief state b, most recent action a and most recent observation o, it then outputs the updated **Belief state.** Pi is policy that is function of belief state and it outputs actions a.

Optimal Policy:

Finding optimal policy in a non-deterministic environment is difficult. Therefore, the optimal policy is found out is to design it with assuming "completely observable world". The MDP values will be (S,A,T,R)
The equation for transition to a new state becomes as follows:

$$\tau(b, a, b') = \sum_{o \in O|SE(b,a,o)=b'} \Pr(o|a, b)$$

Where,
b: current belief states
a: current actions
O: possible successor of belief state b'
τ : Transition function
Optimal policy for finite-horizon is PSPACE-complete

Value Iteration:

Value iteration is necessary to find the optimal policy in a partially observable problem.
There are two segments, calculating the V-value and Q-value over a t-horizon space.
The V-value or V* is the optimal value of those actions that gives maximum possible expected reward + discounted long-term reward in the next state.

Therefore, the equation becomes as follows:

$$V^*(b) = max_{a \in A} [\rho(b,a) + \gamma(b,a,b')V*(b')]$$

Again, symbols are as follows:
b: current belief state
a: current actions
b': future belief state
γ: Discount factor {0,1}
$\rho(b,a)$: Maximum possible expected immediate reward

From the above equation it is evident that after number of steps the V- value converges. The converges value is the optimal value at optimal infinite horizon. This property makes the value iteration algorithm

The pseudocode for the same is as follows:

$V_0(b) = 0$ for $b \epsilon B$
Repeat below steps till V (s) converges
    For all the actions b
        For all the state s
            Q (s, a) = $\sum_s [\rho(b,a) + \gamma(b,a,b')V*(b')]$
        V(s) = max_a Q (s, a)
This algorithm will always converge in finite amount of steps/iteration. Therefore, the policy is within $2\,\gamma\varepsilon/(1-\gamma)$ of the optimal policy.

The witness Algorithm:
This is based on cheng's linear support algorithm
In whiteness algorithm a linear program returns a single point. That single point is called whiteness. The linear program is as follows:

$$V = V_t^*(b) = max_{a \in A}[\rho(b,a) + \gamma \sum_{b' \epsilon B} \tau(b,a,b')\, max\alpha \,.\, b']$$

Where, $Vt(b) = max(\alpha, b)$

The main advantage is that the running time is not exponential.

Representing policies:
Representing policies give birth to what we know as policy graph. This is done by partition of state vectors v. For an optimal action and observation, all the belief state of a partition functions to belief state of same partition but in next step.

Using Policy Graph:
The steps are as follows:

- The agent decides the action only based on start state
- The arcs of the graph are observations that provides new information to the agent, this also helps the agent to make decision
- There is no use of state estimation in the policy graph as the graph itself is optimal

One of the examples of policy graph is as follows:

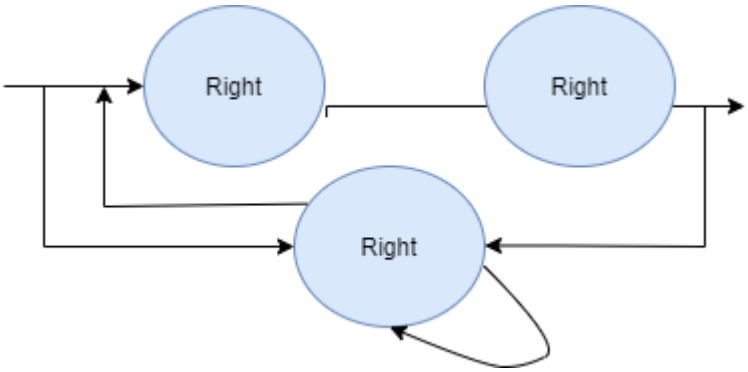

Figure 4 Execute the loop right, right and left to reach the goal/final state

Conclusions

The discussion ends with famous Tiger problem

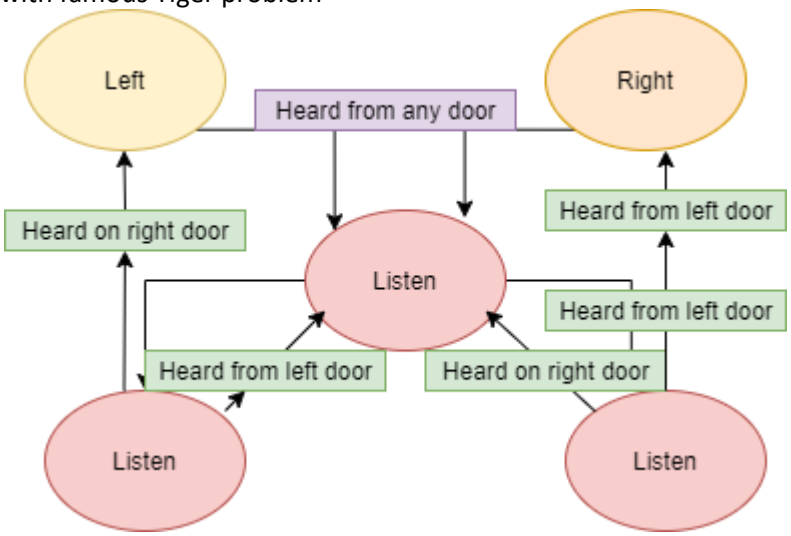

Figure 5 Policy graph of tiger problem

The problem states that the tiger is put into one of the two doors with equal probability and the treasure is put behind the other.
Therefore, the probability that the tiger is behind left or right door is ½
If the agent opens the door with tiger a negative reward is awarded and if the agent opens the door with treasure a positive reward is awarded.
Here one flexibility is provided that the agent can also listen the doors before opening. However, the catch is that the voice of tiger can resemble from any door.
The above graph is the optimal policy of the tiger problem. In the beginning the agent has no understanding of the environment therefore chooses to remain in "Listen" state. If the agent hears a roar from right door, it again goes to listen state but waits to hear the roar again. Once the agent hears the roar again in the right, he is certain and opens the left door for treasure. However, if he hears the roar in the left door, he again goes to the listen state in order to hear consecutive roars from either door. Similar steps are for door in the right.

References

Spaan, Matthijs TJ, Tiago S. Veiga, and Pedro U. Lima. "Decision-theoretic planning under uncertainty with information rewards for active cooperative perception." *Autonomous Agents and Multi-Agent Systems* 29, no. 6 (2015): 1157-1185.

Seuken, Sven, and Shlomo Zilberstein. "Formal models and algorithms for decentralized decision making under uncertainty." Autonomous Agents and Multi-Agent Systems 17.2 (2008): 190-250.

Cassandra, Anthony R., Leslie Pack Kaelbling, and James A. Kurien. "Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation." Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96. Vol. 2. IEEE, 1996.