CS294-40 Learning for Robotics and Control Lecture 10 - 9/30/2008

Partially Observable Systems

Lecturer: Pieter Abbeel *Scribe: David Nachum*

Lecture outline

- POMDP formalism
- Point-based value iteration
- "Global" methods: polytree, enumeration, filtering, witness

1 Partially Observable Markov Decision Process (POMDP) Formalism

A Partially Observable Markov Decision Process (POMDP) is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma, , \Omega)$.
where

1. $\mathcal{S}$ is the set of possible states for the system
2. $\mathcal{A}$ is the set of possible actions
3. $\mathcal{T}$ represents the system dynamics
4. $R : \mathcal{S} \mapsto \Re$ is the reward function
5. $\mathcal{O}$ is the set of possible observations we can make
6. Ω is the probability distribution P(| $\mathcal{S}$)

We can convert a POMDP into a belief state MDP. A belief state space B is a $|\mathcal{S}| - 1$ dimensional simplex whose elements are probability distributions over states:

$$b \in \mathcal{B} : b(s) = Prob(s|\text{all information available at current time})$$

The transition model now describes transitions between beliefs, rather than transitions between states. We define b_a^o to be the belief state reached when starting in belief state b, taking action a, and observing o. Hence,

$$b_a^o(s') = \frac{P(o_{t+1} = o \mid s_{t+1} = s') \sum_s P(s_{t+1} = s' \mid s_t = s, a_t = a) b(s)}{\sum_{s''} P(o_{t+1} = o \mid s_{t+1} = s'') \sum_s P(s_{t+1} = s'' \mid s_t = s, a_t = a) b(s)} = \frac{P(o|s') \sum_s P(s'|s, a) b(s)}{P(o|b, a)}$$

As we iterate through time, the belief states are updated as follows:

$$P(b_a^o \mid b, a) = \sum_{s'} P(o_{t+1} = o \mid s_{t+1} = s') \sum_s P(s' \mid s, a) b(s)$$

Our policies now map beliefs to actions:

$$\pi : \mathcal{B} \to \mathcal{A}$$

and we compute the value of a policy as follows:

$$V^{\pi}(b) = E[\sum_{t=0}^{\infty} \gamma^t R(b_t, a_t) | b_0 = b; \pi]$$

Bellman back-ups for POMDP:

$$V(b) \leftarrow \max_{a \in \mathcal{A}} [\sum_{s \in \mathcal{S}} R(s,a) b(s) + \gamma \sum_{o} P(b_a^o | b, a) V(b_a^o)]$$

2 Point-based value iteration

A practical issue that arises is that, even when our state space is discrete and finite, the belief state space (a $|\mathcal{S}| - 1$ dimensional simplex) is continuous and hence has infinitely many states.

One solution: use function approximation, e.g., grid out the belief state space and use nearest neighbor as function approximation.

Here we study another solution: it turns out that for any finite horizon, the value function of a belief state MDP can be representated by the max over a set of linear functions. Concretely, in the first iteration of our value iteration we have:

$$V_1(b) = \max_{a \in \mathcal{A}} \sum_{s \in \mathcal{S}} R(s,a) b(s) = \max_{\{\alpha_i^{(0)}\}_i} {\alpha_i^{(0)}}^T b$$

This remains true for arbitrary n:

$$V_n(b) = \max_{\{\alpha_i^{(n)}\}_i} {\alpha_i^{(n)}}^T b$$

We define $\mathrm{r}_a = \begin{pmatrix} \mathrm{R(1,a)} \\ \mathrm{R(2,a)} \\ \vdots \\ \mathrm{R(s,a)} \end{pmatrix}$

We iteratively update the value of the belief state:

$$\begin{aligned}
V_{n+1}(b) &= \max_a [r_a^T b + \gamma \sum_o P(o|b,a) V_n(b_a^o)] \\
&= \max_a [r_a^T b + \gamma \sum_o P(o|b,a) \max_{\{\alpha_i^{(n)}\}_i} {\alpha_i^{(n)}}^T b_a^o] \\
&= \max_a [r_a^T b + \gamma \sum_o P(o|b,a) \max_{\{\alpha_i^{(n)}\}_i} \sum_{s'} {\alpha_i^{(n)}}^T (s') b_a^o(s')] \\
&= \max_a [r_a^T b + \gamma \sum_o P(o|b,a) \max_{\{\alpha_i^{(n)}\}_i} \sum_{s'} {\alpha_i^{(n)}}^T (s') \frac{P(o|s') \sum_s P(s'|s,a) b(s)}{P(o|b,a)}] \\
&= \max_a [r_a^T b + \gamma \sum_o \max_{\{g_{i,a,o}^{(n)}\}_i} {g_{i,a,o}^{(n)}}^T b]
\end{aligned}$$

where

$$g_{i,a,o}^{(n)}(s) = \sum_{s'} P(o|s') P(s'|s,a) \alpha_i^{(n)}(s')$$

To do the backup for b:

$$\forall i, a, o, \text{ compute } g_{i,a,o}^{(n)}(s) \text{ according to the equation above}$$

$$\text{let } g_a^{(n)} = r_a + \gamma \sum_o \arg\max_{g_{i,a,o}^{(n)}} {g_{i,a,o}^{(n)}}^T b$$

This yields

$$V_{n+1}(b) = {\alpha_b^{(n+1)}}^T b$$

for

$$\alpha_b^{(n+1)} \in arg\max_{g_a^{(n)}} {g_a^{(n)}}^T b$$

Point-based value iteration is efficient, but inexact due to the discrete choice of belief states. However, there are clever ways to pick the points:

1. Pineau, Gordon, Thrun: Point-Based Value Iteration—Begin at some initial belief. Then pick belief points by forward simulation and prune by distance.
2. Vlassis, Spaan: Perseus—In every iteration, only do the Bellman back-up for a point if the back-ups of other belief states has not yet increased that points value function. [Assumes you initialize the value function with a lower-bound.] This ensures the value function increases for every belief point in every iteration. It works very well in practice.

3 "Global" methods

Consider a policy with horizon H in a POMDP. We can represent this as a policy tree (assuming the policy is deterministic). We can compute the value of being at state s and following policy tree p:

$$V_p^H = R(s, p(s)) + \gamma \sum_{s' \in \mathcal{S}} P(s'|s, p(s)) \sum_o P(o|s') V_{\hat{p}}^{H-1}(s)$$

where $\hat{p}$ is a subtree of p.
If we do not know what state we are in, but do know what belief state we are in, we compute:

$$V_p^H(b) = \sum_s b(s) V_p^H(s)$$

For the optimal (within horizon H) tree, we define

$$V^H(b) = \max_p V_p^H(b)$$

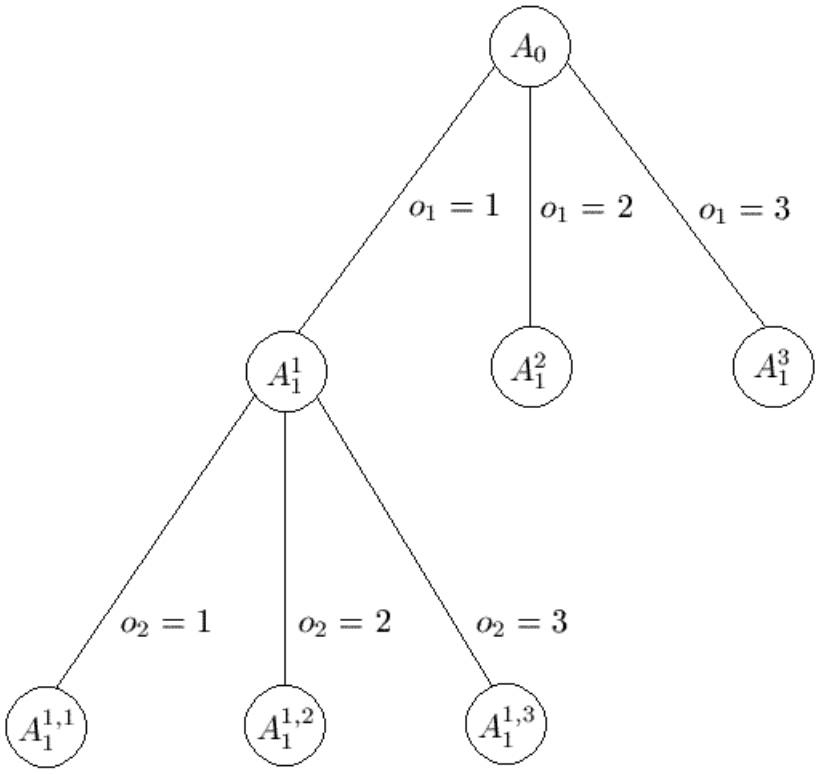

Figure 1: Policy tree illustration.

3.1 "Global" value iteration for belief state MDP (Algorithm)

t = 1
V_1 = set of 1-step policy trees
Loop:

1. t++
2. Compute V_t^+, the set of possibly useful t-step policy trees from V_{t-1}
3. Prune/Filter V_t^+ to get V_t, the set of useful t-step policy trees

Until $\sup_b |V_t(b) - V_{t-1}(b)| < \epsilon$

3.2 Basic filtering:

$\forall j$, solve

$$\begin{aligned} \max_b \; & t \\ & \forall i \neq j,\ \alpha_j^T b \geq \alpha_i^T b + t \\ & b \geq 0 \\ & 1^T b = 1 \end{aligned}$$

If for j we have that the solution $t > 0$, then the policy tree j is useful. In that, there is a belief point b for which policy tree j is the optimal policy. Otherwise, we prune out policy tree j.

3.3 Lark's filtering:

Incrementally build up the set of useful α_i. This way the size of the LP scales with V_t rather than V_t^+ For $j = 1, 2, ..., |V_t^+|$:

$$\begin{aligned}
&\max_b t \\
&\quad \forall k \in V_t, t \leq \alpha_j^T b - \alpha_k^T b \\
&\quad b \geq 0 \\
&\quad 1^T b = 1 \\
&\text{if } t > 0, \text{ find } \max_{j \in 1,2,...,|V_t^+|} b^T \alpha_j \\
&\text{add } argmax_{j \in 1,2,...,|V_t^+|} b^T \alpha_j \text{ to } V_t
\end{aligned}$$

3.4 Witness algorithm

In the witness algorithm, we try to avoid constructing V_t^+. We do this by only adding a tree if it is optimal for some belief b. It is is sufficient to check this per subtree: Is there a belief state b we can reach after action a and observation o such that the policy tree $p \in V_{t-1}$ would be optimal? This problem can be solved by linear programming.