Tutorial 9: MDPs & POMDPs

Probabilistic Artificial Intelligence

Ilija Bogunovic

Friday 22$^{\text{nd}}$ November, 2019

ETH Zürich

Outline

Today:

- Value iteration (HW Problem 2,3)
- Convergence of Value iteration (HW Problem 1)
- Policy iteration (HW Problem 4)
- POMDPs

Markov Decision Processes

Environment in which all states are Markov.

Definition

A Markov Decision Process is specified by $\langle X, A, P, r, \gamma \rangle$:

- X is a finite set of states
- A is a finite set of actions
- $P(x'|x, a)$ is a transition probability
- $r(x, a)$ is a reward function
- γ is a discount factor $\gamma \in [0, 1]$

Today, we assume r and P are known.

Value function

Planning problem: Discover deterministic policy $\pi : X \to A$

Value of a policy: $V^\pi(x) = \mathbb{E}[\sum_{t=0}^\infty \gamma^t r(X_t, \pi(X_t))|X_0 = x]$

Recursive relation: $V^\pi(x) = r(x, \pi(x)) + \gamma \sum_{x' \in X} P(x'|x, \pi(x)) V^\pi(x')$

Bellman equation: For the optimal policy π^* it holds that

$$V^*(x) = \max_{a \in A}[r(x, a) + \gamma \sum_{x' \in X} P(x'|x, a) V^*(x')]$$

Value and Policy Iteration

HW Problem 2,3: Value iteration

Algorithm:

- Initialize: $V_0(x)$ for every $x \in X$, accuracy ϵ
- For $t = 1, 2, \ldots$
    - For each $x \in X$ set:

      $$V_t(x) = \max_{a \in A} \left[r(x,a) + \gamma \sum_{x' \in X} P(x'|x,a) V_{t-1}(x') \right]$$

    - Break in the case $\|V_t - V_{t-1}\|_\infty \leq \epsilon$
- Return V_t

HW Problem 2,3: Value iteration

Algorithm:

- Initialize: $V_0(x)$ for every $x \in X$, accuracy ϵ
- For $t = 1, 2, \ldots$
    - For each $x \in X$ set:

      $$V_t(x) = \max_{a \in A} \left[r(x,a) + \gamma \sum_{x' \in X} P(x'|x,a) V_{t-1}(x') \right]$$

    - Break in the case $\|V_t - V_{t-1}\|_\infty \leq \epsilon$
- Return V_t

Main idea: Solve Bellman equation by using dynamic programming
Greedy policy: For every $x \in X$, use the obtained V_t to set $\pi(x)$ to

$$\arg\max_{a \in A} \left[r(x,a) + \gamma \sum_{x' \in X} P(x'|x,a) V_t(x') \right]$$

Recall: Policy is optimal iff it is greedy w.r.t. its induced value function

HW Problem 1: Convergence of Value iteration

- Bellman update operator $\mathcal{B} : \mathbb{R}^n \to \mathbb{R}^n$, and $n = |X|$,

$$\mathcal{B}(V(x)) = \max_{a \in A} [r(x, a) + \gamma \sum_{x' \in X} P(x'|x, a)V(x')]$$

- Question: Show that the Bellman operator is a contraction, i.e., for any V, V' holds that:

$$\|\mathcal{B}V - \mathcal{B}V'\|_\infty \leq \gamma \|V - V'\|_\infty$$

HW Problem 1: Convergence of Value iteration

- Bellman update operator $\mathcal{B} : \mathbb{R}^n \to \mathbb{R}^n$, and $n = |X|$,

$$\mathcal{B}(V(x)) = \max_{a \in A} [r(x,a) + \gamma \sum_{x' \in X} P(x'|x,a)V(x')]$$

- Question: Show that the Bellman operator is a contraction, i.e., for any V, V' holds that:

$$\|\mathcal{B}V - \mathcal{B}V'\|_\infty \le \gamma \|V - V'\|_\infty$$

- Two important properties of contraction: 1) has at most one fixed point, 2) repeated application reaches the fixed point in the limit.
- Implication: Exponential convergence for $\gamma < 1$
  - We have $\|\mathcal{B}V_t - V^*\| \le \gamma \|V_t - V^*\|$, where we used $\mathcal{B}V^* = V^*$
  - Max. initial error: $\|V_0 - V^*\| \le \frac{2R_{\max}}{(1-\gamma)}$ (for all x, a: $|r(x,a)| \le R_{\max}$)
  - Run N iterations to reach error ϵ: $\gamma^N \frac{2R_{\max}}{(1-\gamma)} \le \epsilon$
  - Solve for N to obtain $N = \lceil (\log \frac{1}{\gamma})^{-1} \log \frac{2R_{\max}}{\epsilon(1-\gamma)} \rceil$

HW Problem 4: Policy iteration

Algorithm:

- Initialize: policy π'
- Repeat
  - $\pi \leftarrow \pi'$
  - Policy evaluation: Compute values by solving the linear system

    $$V^{\pi}(x) = r(x, \pi(x)) + \gamma \sum_{x' \in X} P(x'|x, \pi(x)) V^{\pi}(x')$$

  - Policy improvement: For every state $x \in X$

    $$\pi(x) \leftarrow \arg\max_{a \in A} \left[r(x, a) + \gamma \sum_{x' \in X} P(x'|x, a) V^{\pi}(x') \right]$$

- until $\pi = \pi'$
- Return π

HW Problem 4: Policy iteration

Algorithm:

- Initialize: policy π'
- Repeat
  - $\pi \leftarrow \pi'$
  - Policy evaluation: Compute values by solving the linear system

$$V^{\pi}(x) = r(x, \pi(x)) + \gamma \sum_{x' \in X} P(x'|x, \pi(x)) V^{\pi}(x')$$

  - Policy improvement: For every state $x \in X$

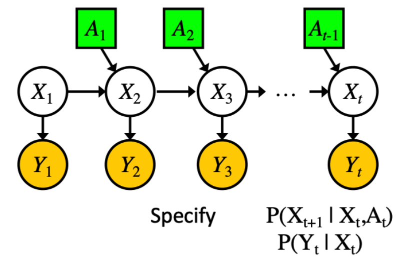
$$\pi(x) \leftarrow \arg\max_{a \in A} \left[r(x, a) + \gamma \sum_{x' \in X} P(x'|x, a) V^{\pi}(x') \right]$$

- until $\pi = \pi'$
- Return π

Main idea: Alternate between policy evaluation and improvement steps
Policy iteration: Finds optimal policy in polynomial number of iterations

POMDPs

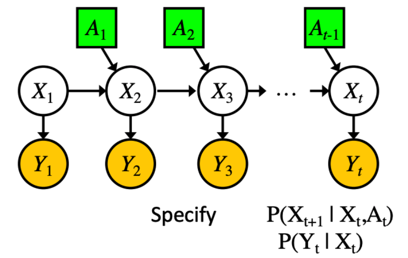
POMDPs



Specify $P(X_{t+1} \mid X_t, A_t)$
$P(Y_t \mid X_t)$

- POMDP contains same elements as MDP + sensor model
- Sensor (observation) model $P(y|x)$
- **Belief state:** distribution over states, $b(x)$ is assigned to being in x
- Agent keeps track of the belief state instead of observations
- Policies in POMDPs map from belief states to actions

Example

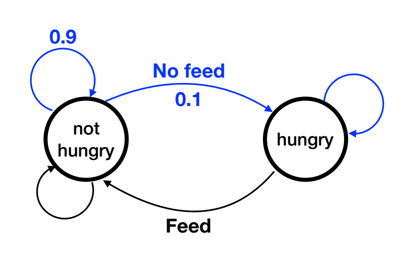


Example POMDP:

- Decide whether to feed baby given cry / no cry
- $A = \{\text{feed}, \text{no feed}\}$
- $X = \{\text{not hungry}, \text{hungry}\}$
- $Y = \{\text{cry}, \text{no cry}\}$
- Reward -10 for hungry, and -5 for feed; infinite horizon with γ

Example

Transition model $P(x'|x, a)$:



Sensor model $P(y|x)$:

- $P(y_1|x_2) = 0.8$, cry when hungry
- $P(y_1|x_1) = 0.1$, cry when not hungry

POMDP decision cycle

Protocol:

- Given the current belief state b, agent executes $a = \pi(b)$
- Agent receives observation y
- Agent updates its belief state based on previous b, a, y
- Repeat, i.e., go to the first step

POMDP decision cycle

Protocol:

- Given the current belief state b, agent executes $a = \pi(b)$
- Agent receives observation y
- Agent updates its belief state based on previous b, a, y
- Repeat, i.e., go to the first step

Example:

- Initial belief $b = [0.5, 0.5] \rightarrow$ no feed, cry $\rightarrow b = [0.0928, 0.9072]$
- $b = [0.0928, 0.9072] \rightarrow$ feed, no cry $\rightarrow b = [1, 0]$
- $b = [1, 0] \rightarrow$ no feed, no cry $\rightarrow b = [0.9759, 0.0241]$

How does the agent update its belief state b?

Updating of beliefs

- Agent does action a in belief state $b(x)$ and observes y, then new belief $b'(x')$ can be calculated using Bayes' rule:

$$b'(x') = \alpha P(y|x') \sum_{x} P(x'|x, a) b(x),$$

that is, $b' = \textsc{UpdateBelief}(b, a, y)$.

Updating of beliefs

- Agent does action a in belief state $b(x)$ and observes y, then new belief $b'(x')$ can be calculated using Bayes' rule:

  $$b'(x') = \alpha P(y|x') \sum_x P(x'|x, a) b(x),$$

  that is, $b' = \textsc{UpdateBelief}(b, a, y)$.

- The probability of observation can be computed by summing over all possible x'

  $$\begin{aligned} P(y|a, b) &= \sum_{x'} P(y|a, x', b) P(x'|a, b) && (1) \\ &= \sum_{x'} P(y|x') P(x'|a, b) && (2) \\ &= \sum_{x'} P(y|x') \sum_x P(x'|x, a) b(x) && (3) \end{aligned}$$

Example

- Initial $b = [0.5, 0.5]$
- Agent performs "no feed" and observes "cry"
- Belief state update:

Example

- Initial $b = [0.5, 0.5]$
- Agent performs ”no feed” and observes ”cry”
- Belief state update:

$$b(\text{hungry}) \propto P(\text{cry}|\text{hungry}) \sum_{x} 0.5 * P(\text{hungry}|x, \text{no feed}) \tag{4}$$

$$\propto 0.8 * (0.5 * 0.1 + 0.5 * 1) = 0.44 \tag{5}$$

$$b(\text{not hungry}) \propto P(\text{cry}|\text{not hungry}) \sum_{x} 0.5 * P(\text{not hungry}|x, \text{no feed}) \tag{6}$$

$$\propto 0.1 * (0.5 * 0 + 0.5 * 0.9) = 0.045 \tag{7}$$

- Renormalize to obtain the next state belief $b = [0.0928, 0.9072]$

POMDP as "belief-state" MDP

We can define a new "belief-state" MDP with:

- States: Beliefs over states, $B = \{b : b \in [0,1]^n, \sum_{x \in X} b(x) = 1\}$
- Actions: Set of actions remains the same
- Transition model:

$$P(b'|b,a) = \sum_y P(b'|y,a,b)P(y|a,b) \tag{8}$$

$$= \sum_y P(b'|y,a,b) \sum_{x'} P(y|x') \sum_x P(x'|x,a)b(x) \tag{9}$$

POMDP as ”belief-state” MDP

We can define a new ”belief-state” MDP with:

- States: Beliefs over states, $B = \{b : b \in [0,1]^n, \sum_{x \in X} b(x) = 1\}$
- Actions: Set of actions remains the same
- Transition model:

$$P(b'|b,a) = \sum_y P(b'|y,a,b)P(y|a,b) \tag{8}$$

$$= \sum_y P(b'|y,a,b) \sum_{x'} P(y|x') \sum_x P(x'|x,a)b(x) \tag{9}$$

- Reward function:

$$r(b,a) = \sum_x b(x)r(x,a)$$

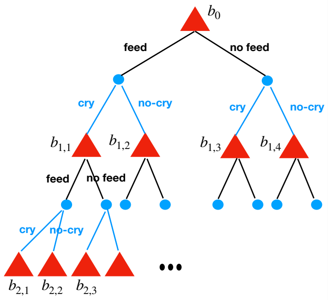
Optimal policy for this MDP is also optimal for the original POMDP.
Standard MDP planning methods cannot be used directly. Why?

Finite time horizon planning

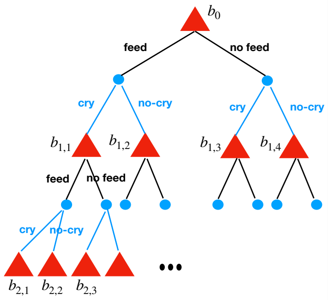
For **finite horizon** T, we can determine the optimal action(s) by planning from the current belief state.

- The number of belief states reachable (from the current belief state) is small compared to the full belief space.
- Still, the set of reachable belief states is typically exponential in T.
- Can work for small T, $|Y|$ and $|A|$.
- Idea: Tree-based search up to horizon T.

Example


b_0
feed
no feed
cry
no-cry
cry
no-cry
$b_{1,1}$
$b_{1,2}$
$b_{1,3}$
$b_{1,4}$
feed
no feed
cry
no-cry
•••
$b_{2,1}$
$b_{2,2}$
$b_{2,3}$

Example



The 1-step lookahead strategy:

- For initial belief state b and every action a compute

$$Q(b,a) = r(b,a) + \sum_y P(y|b,a) r(b_{a,y})$$

  where $b_{a,y} = \textsc{UpdateBelief}(b,a,y)$.
- Find optimal action: $\arg\max_{a \in A} Q(b,a)$

Exercise

Find 1-step optimal lookahead strategy in our example for initial belief state $b = [0.5, 0.5]$ by computing $Q(a, b)$ for every a.

Exercise

Find 1-step optimal lookahead strategy in our example for initial belief state $b = [0.5, 0.5]$ by computing $Q(a, b)$ for every a.

Some hints:

- Rewards: $r(\text{hungry}, \text{feed}) = -15$, $r(\text{hungry}, \text{no feed}) = -10$, $r(\text{not hungry}, \text{feed}) = -5$, $r(\text{not hungry}, \text{no feed}) = 0$, $r(\text{hungry}) = -10$, $r(\text{not hungry}) = 0$
- Example: $r(b, \text{feed}) = -10$
- Example: When $a = \text{no feed}$, $y = \text{cry}$, it follows $b_{a,y} = [0.0928, 0.9072]$, $r(b_{a,y}) = 0.9072 * (-10) + 0 * 0.0928 = -9.072$.

Forward search

In general, for finite horizons T, we can search for optimal action:

```
function ActionSearch(b,T)
    if T=0 then
        return [None, $r(b)$]
    $[a^*, v^*] \leftarrow [None, -\infty]$
    for $a \in A$ do
        $v \leftarrow r(b, a)$
        for $y \in Y$ do
            $b' \leftarrow$ UpdateBelief$(b, a, y)$
            $[a', v'] \leftarrow$ ActionSearch$(b', T-1)$
            $v \leftarrow v + P(y|b, a)v'$
        if $v > v^*$ then
            $[a^*, v^*] \leftarrow [a, v]$
    return $[a^*, v^*]$
```

Outline

Today:

- Value iteration (HW Problem 2,3)
- Convergence of Value iteration (HW Problem 1)
- Policy iteration (HW Problem 4)
- POMDPs

Ack.

- Slides based on material accompanying the textbook “AI: A Modern Approach” (3rd edition) by S. Russell and P. Norvig, as well as material by A. Krause and M. Kochenderfer.