Introduction to Artificial Intelligence
Probabilistic Planning with Markov Decision Processes

Janyl Jumadinova
November 4, 2016
Status of Classical Planning

- Classical planning works!
  - Large problems solved very fast (non-optimally)
Status of Classical Planning

- **Classical planning works!**
  - Large problems solved very fast (non-optimally)

- **limitations**
  - Does not model Uncertainty (no probabilities)
  - Does not deal with Incomplete Information (no sensing)
  - Deals with very Simple Cost Structure (no state dependent costs)
Static vs. Dynamic

Environment

Percepts

Actions

What action next?

Fully vs. Partially Observable

Deterministic vs. Stochastic

Perfect vs. Noisy

Sequential vs. Concurrent

Instantaneous vs. Durative
Beyond Classical Planning

- Develop solver for more general models; e.g., MDPs and POMDPs
  +: generality
  -: complexity
Markov Decision Processes (MDPs)

A fundamental framework for probabilistic planning

MDPs are fully observable, probabilistic state models:

- a state space $S$
- a set $G \subseteq S$ of goal states
- actions $A(s) \subseteq A$ applicable in each state $s \in S$
- transition probabilities $P_a(s_0|s)$ for $s \in S$ and $a \in A(s)$
- action costs $c(a, s) > 0$
Markov Decision Processes (MDPs)

A fundamental framework for probabilistic planning

MDPs are fully observable, probabilistic state models:

- a state space $S$
- a set $G \subseteq S$ of goal states
- actions $A(s) \subseteq A$ applicable in each state $s \in S$
- transition probabilities $P_a(s_0|s)$ for $s \in S$ and $a \in A(s)$
- action costs $c(a, s) > 0$

Solutions are functions (policies) mapping states into actions

Optimal solutions have minimum expected costs
\[ T(s_3, a_1, s_2, 1) = 1.0 \quad T(s_3, a_2, s_1, 2) = 1.0 \]
\[ R(s_3, a_1, s_2, 1) = -7.2 \quad R(s_3, a_1, s_3, 2) = 3.5 \]
MDP Solution

- Want a way to choose an action in a state, i.e., a policy $\pi$
- What does a policy look like?
  - Can pick action based on states visited + actions used so far, i.e., execution history $h = s(1)a(1)s(2)a(2)\ldots$
  - Can pick actions randomly
Want a way to choose an action in a state, i.e., a policy $\pi$

What does a policy look like?
- Can pick action based on states visited $+$ actions used so far, i.e., execution history $h = s(1)a(1)s(2)a(2)\ldots$
- Can pick actions randomly

Thus, in general an MDP solution is a probabilistic history-dependent $\pi : H \times A \rightarrow [0, 1]$
Evaluating MDP Solutions

- Executing a policy yields a sequence of rewards
Evaluating MDP Solutions

- Define utility function $u(R_1, R_2, \ldots)$ to be some “quality measure” of a reward sequence
- Define value function as $V : H \rightarrow [-\infty, \infty]$
- Define value function of a policy after history $h$ to be some utility function of subsequent rewards: $V^{\pi}(h) = u(R_1, R_2, \ldots)$
Optimal MDP Solution

Want a behavior that is "best" in every situation

$\pi^* \text{ is an optimal policy if } V^*(h) \geq V_\pi(h) \text{ for all } \pi, \text{ for all } h$
Optimal MDP Solution

- **Want**: a behavior that is “best” in every situation
- $\pi^*$ is an optimal policy if $V^*(h) \geq V^\pi(h)$ for all $\pi$, for all $h$
Partially Observable MDPs (POMDPs)

POMDPs are **partially observable, probabilistic** state models:

- states $s \in S$
- actions $A(s) \in A$
- costs $c(a, s) > 0$
- transition probabilities $P_a(s_0|s)$ for $s \in S$ and $a \in A(s)$
- initial belief state $b_0$
- final belief states $b_F$
- sensor model given by probabilities $P_a(o|s)$, $o \in O_{bs}$
Partially Observable MDPs (POMDPs)

POMDPs are partially observable, probabilistic state models:

- states \( s \in S \)
- actions \( A(s) \in A \)
- costs \( c(a, s) > 0 \)
- transition probabilities \( P_a(s_0|s) \) for \( s \in S \) and \( a \in A(s) \)
- initial belief state \( b_0 \)
- final belief states \( b_F \)
- sensor model given by probabilities \( P_a(o|s) \), \( o \in O_{bs} \)

**Belief** states are probability distributions over \( S \)
Solutions are **policies** that map belief states into actions
**Optimal** policies minimize expected cost to go from \( b_0 \) to \( b_F \)
Navigation Problems

Consider robot that has to reach target $G$ when

1. initial state is known and actions are deterministic

How do these problems map into the models considered?
Consider robot that has to reach target $G$ when

2. initial state is unknown and actions are deterministic
Consider robot that has to reach target $G$ when

3. states are fully observable and actions are stochastic
Navigation Problems

Consider robot that has to reach target $G$ when

4. states are partially observable and actions are stochastic