

Planning, Execution & Learning: Planning with POMDPs

Reid Simmons
Manuela Veloso

POMDP Models

- What is a **POMDP**?
 - Basically an **MDP**, except that state is not known with certainty
 - Represent **beliefs** about the state of the world
 - Handles hidden (unobservable or partially observable) state
- POMDP Model
 - Finite set of states: $s_1, \dots, s_n \in S$
 - Finite set of actions: $a_1, \dots, a_m \in A$
 - Probabilistic state, action transitions: $p(s_j | a, s_i)$
 - Reward (cost) for each state: $r(s, a)$
 - **Conditional observation probabilities**: $p(o | s)$ (more generally, $p(o | s, a, s')$)

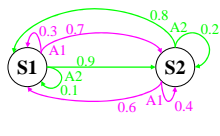
Belief State

- Belief State
 - Probability distribution (belief state) over world states
 - $b(s) = p(s)$
 - Continuous
 - Action update rule:
 - $b'(s) = \sum_{s' \in S} p(s' | a, s) \cdot b(s')$
 - Observation update rule:
 - $b'(s) = p(o | s) \cdot b(s) / k$

Converting POMDP to Belief-State MDP

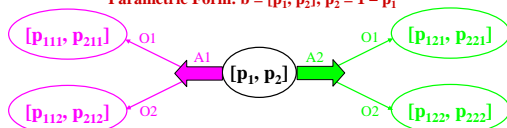
- Equivalent MDP Model
 - Each MDP state is a **probability distribution** (continuous belief state b) over the states of the original POMDP
 - State transitions are product of actions and observations
 - $b'(s') = p(s' | a, o, b) = p(s' | a, b) \cdot p(o | s', a, b) / p(o | a, b)$
 - $p(s' | a, b) = \sum_{s \in S} p(s' | a, s) \cdot b(s)$
 - $p(o | s', a, b) = p(o | s')$
 - $p(o | a, b) = \sum_{s \in S} p(o | s') \cdot p(s' | a, b)$
 - **Note**: Actions are **deterministic** in belief space
 - MDP rewards are expected rewards of original POMDP
 - $R(a, b) = \sum_{s \in S} r(a, s) \cdot b(s)$

POMDP Example (I)

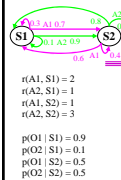


$r(A1, S1) = 2$
 $r(A2, S1) = 1$
 $r(A1, S2) = 1$
 $r(A2, S2) = 3$
 $p(O1 | S1) = 0.9$
 $p(O2 | S1) = 0.1$
 $p(O1 | S2) = 0.5$
 $p(O2 | S2) = 0.5$

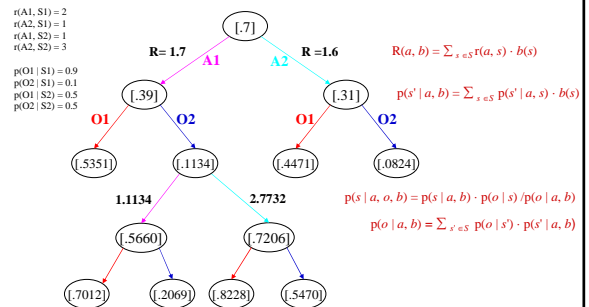
Parametric Form: $b = [p_1, p_2]; p_2 = 1 - p_1$



POMDP Example (II)



$r(A1, S1) = 2$
 $r(A2, S1) = 1$
 $r(A1, S2) = 1$
 $r(A2, S2) = 3$
 $p(O1 | S1) = 0.9$
 $p(O2 | S1) = 0.1$
 $p(O1 | S2) = 0.5$
 $p(O2 | S2) = 0.5$



POMDP Conversion (I)

- Transformed Rewards**

$$R(A1, b) = 2p_1 + p_2 = 2p_1 + (1 - p_1) = (p_1 + 1)$$

$$R(A2, b) = p_1 + 3p_2 = p_1 + 3(1 - p_1) = (3 - 2p_1)$$

- Transformed Action Transition Probabilities**

$$\begin{aligned} p(S1 | A1, b) &= p(S1 | A1, S1) \cdot p(S1) + p(S1 | A1, S2) \cdot p(S2) \\ &= 0.3p_1 + 0.6p_2 = 0.3p_1 + 0.6(1 - p_1) \\ &= 0.6 - 0.3p_1 \end{aligned}$$

$$p(S2 | A1, b) = 0.7p_1 + 0.4p_2 = 0.4 + 0.3p_1$$

$$p(S1 | A2, b) = 0.1p_1 + 0.8p_2 = 0.8 - 0.7p_1$$

$$p(S2 | A2, b) = 0.9p_1 + 0.2p_2 = 0.2 + 0.7p_1$$

POMDP Conversion (II)

- Transformed Observation Probabilities**

$$\begin{aligned} p(O1 | A1, b) &= p(O1 | S1) \cdot p(S1 | A1, b) + p(O1 | S2) \cdot p(S2 | A1, b) \\ &= 0.9(0.6 - 0.3p_1) + 0.5(0.4 + 0.3p_1) \\ &= 0.74 - 0.12p_1 \end{aligned}$$

$$\begin{aligned} p(O2 | A1, b) &= p(O2 | S1) \cdot p(S1 | A1, b) + p(O2 | S2) \cdot p(S2 | A1, b) \\ &= 0.1(0.6 - 0.3p_1) + 0.5(0.4 + 0.3p_1) \\ &= 0.26 + 0.12p_1 \end{aligned}$$

$$\begin{aligned} p(O1 | A2, b) &= p(O1 | S1) \cdot p(S1 | A2, b) + p(O1 | S2) \cdot p(S2 | A2, b) \\ &= 0.9(0.8 - 0.7p_1) + 0.5(0.2 + 0.7p_1) \\ &= 0.82 - 0.28p_1 \end{aligned}$$

$$\begin{aligned} p(O2 | A2, b) &= p(O2 | S1) \cdot p(S1 | A2, b) + p(O2 | S2) \cdot p(S2 | A2, b) \\ &= 0.1(0.8 - 0.7p_1) + 0.5(0.2 + 0.7p_1) \\ &= 0.18 + 0.28p_1 \end{aligned}$$

POMDP Conversion (III)

- State Transition Probabilities (Actions and Observations)**

$$p(s | a, o, b) = p(o | s) \cdot p(s | a, b) / p(o | a, b)$$

$$\begin{aligned} p(S1 | A1, O1, b) &= p(S1 | A1, b) \cdot p(O1 | S1) / p(O1 | A1, b) \\ &= 0.9(0.6 - 0.3p_1) / (0.74 - 0.12p_1) \\ &= (0.54 - 0.27p_1) / (0.74 - 0.12p_1) \end{aligned}$$

$$p(S2 | A1, O1, b) = (0.20 + 0.15p_1) / (0.74 - 0.12p_1)$$

$$p(S1 | A2, O1, b) = (0.72 - 0.63p_1) / (0.82 - 0.28p_1)$$

$$p(S2 | A2, O1, b) = (0.10 + 0.35p_1) / (0.82 - 0.28p_1)$$

$$p(S1 | A1, O2, b) = (0.06 - 0.03p_1) / (0.26 + 0.12p_1)$$

$$p(S2 | A1, O2, b) = (0.20 + 0.15p_1) / (0.26 + 0.12p_1)$$

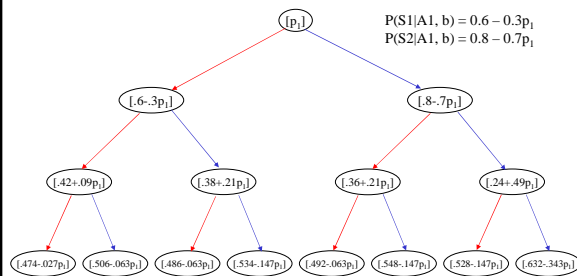
$$p(S1 | A2, O2, b) = (0.08 - 0.07p_1) / (0.18 + 0.28p_1)$$

$$p(S2 | A2, O2, b) = (0.10 + 0.35p_1) / (0.18 + 0.28p_1)$$

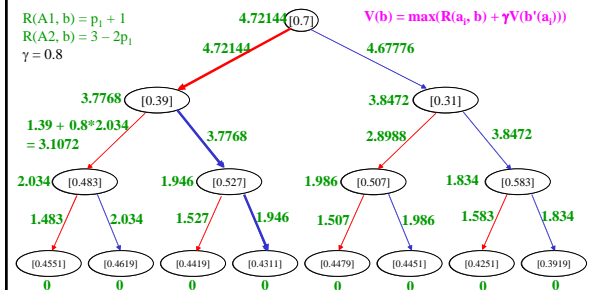
Solving POMDPs

- Maximize Expected Reward Over Belief Space: $V(b)$
 - Optimal policy may trade off reward-producing actions for actions that reduce uncertainty
- Representational Choices
 - Exact V , exact b
 - Optimal solutions, but intractable
 - Approximate V , exact b
 - Differentiable function approximators (higher-order polynomials, neural nets, ...)
 - Exact V , Approximate b
 - Dynamic Bayes Net, Particle Filters
 - Approximate V , Approximate b
 - Combos of above
- Greedy Approaches Based on Solving Underlying MDP

Solving POMDPs Without Observations I



Solving POMDPs Without Observations II



Solving POMDPs Without Observations III

$R(A1, b) = p_1 + 1$
 $R(A2, b) = 3 - 2p_1$
 $\gamma = 0.8$

$V(b) = \max_a \{R(a, b) + \gamma V(b'(a))\}$

Planning, Execution & Learning: POMDP 13 Simmons & Veloso: Fall 2008

Solving POMDPs With Observations

$V(b) = \max_a \{R(a, b) + \gamma \sum_{b'} p(b' | a, b) V(b')\}$

Planning, Execution & Learning: POMDP 14 Simmons & Veloso: Fall 2008

Exact Solution to POMDP (I)

- Convert to MDP, and use Value Iteration
 - $V(b) = \max_a \{R(a, b) + \gamma \sum_{b'} p(b' | a, b) V(b')\}$
- Use Fact that Value Function is *Piece-Wise Linear Convex*
 - $V(b) = \max_{v \in \Psi} (v \bullet b)$
- Policy Composed of *Alpha Vectors*
 - Value of taking an action, then following a *fixed* policy

Planning, Execution & Learning: POMDP 15 Simmons & Veloso: Fall 2008

Exact Solution to POMDP (II)

$V(b) = \max_a \{R(a, b) + \gamma \sum_{b'} p(b' | a, b) V(b')\}$

$p(b' | a, b) = p(o | a, b)$

$b' = [p(S1 | a, o, b), p(S2 | a, o, b), \dots]$

$b' = [p(o | S1)p(S1 | a, b)/p(o | a, b), p(o | S2)p(S2 | a, b)/p(o | a, b), \dots]$

$\underline{b}' = [p(o | S1)p(S1 | a, b), p(o | S2)p(S2 | a, b), \dots]$

$V(b) = \max_a \{R(a, b) + \gamma \sum_{\underline{b}'} p(o | a, b) \max_{v \in \Psi} (v \bullet \underline{b}')/p(o | a, b)\}$

$V(b) = \max_a \{R(a, b) + \gamma \sum_{\underline{b}'} \max_{v \in \Psi} (v \bullet \underline{b}')\}$

Planning, Execution & Learning: POMDP 16 Simmons & Veloso: Fall 2008

Exact Solution to POMDP (III)

Horizon-Zero Solution: $V_0(b) = 0$

Horizon-One Solution:

$V_1^{A1}(b) = R(A1, b) + \gamma \cdot 0 = (p_1 + 1)$
 $V_1^{A2}(b) = R(A2, b) + \gamma \cdot 0 = (3 - 2p_1)$
 $\Psi_1 = \{[2, 1], [1, 3]\}$
 $V_1([0.5, 0.5]) = \max([2, 1] \bullet [0.5, 0.5], [1, 3] \bullet [0.5, 0.5])$
 $= \max(1.5, 2) = 2$

Crossover Point:

$[2, 1] \bullet [p_1, p_2] = [1, 3] \bullet [p_1, p_2]$
 $2p_1 + (1 - p_1) = p_1 + 3(1 - p_1)$
 $3p_1 = 2$

Planning, Execution & Learning: POMDP 17 Simmons & Veloso: Fall 2008

Exact Solution to POMDP (IV)

Horizon-Two Solution:

Value Function for Action A1:

$V_2^{A1}(b) = R(A1, b) + \gamma \cdot \sum_{\underline{b}'} \max_{v \in \Psi} (v \bullet \underline{b}')$
 $= (p_1 + 1) + \gamma \cdot \{ \max_{v \in \Psi} (v \bullet \underline{b}'_1) + \max_{v \in \Psi} (v \bullet \underline{b}'_2) \}$
 $\underline{b}'_1 = [p(O1 | S1)p(S1 | A1, b), p(O1 | S2)p(S2 | A1, b)]$
 $= [(0.54 - 0.27p_1), (0.20 + 0.15p_1)]$
 $\underline{b}'_2 = [(0.06 - 0.03p_1), (0.20 + 0.15p_1)]$
 $V_2^{A1}(b)_a = (p_1 + 1) + \gamma \cdot ([2, 1] \bullet \underline{b}'_1 + [2, 1] \bullet \underline{b}'_2)$
 $= (p_1 + 1) + \gamma \cdot \{2(0.54 - 0.27p_1) + (0.20 + 0.15p_1) + 2(0.06 - 0.03p_1) + (0.20 + 0.15p_1)\}$
 $= (p_1 + 1) + \gamma \cdot (1.6 - 0.3p_1)$
 $V_2^{A1}(b)_b = (p_1 + 1) + \gamma \cdot ([2, 1] \bullet \underline{b}'_1 + [1, 3] \bullet \underline{b}'_2) = (p_1 + 1) + \gamma \cdot (1.94 + 0.03p_1)$
 $V_2^{A1}(b)_c = (p_1 + 1) + \gamma \cdot ([1, 3] \bullet \underline{b}'_1 + [2, 1] \bullet \underline{b}'_2) = (p_1 + 1) + \gamma \cdot (1.46 + 0.27p_1)$
 $V_2^{A1}(b)_d = (p_1 + 1) + \gamma \cdot ([1, 3] \bullet \underline{b}'_1 + [1, 3] \bullet \underline{b}'_2) = (p_1 + 1) + \gamma \cdot (1.86 + 0.6p_1)$

Planning, Execution & Learning: POMDP 18 Simmons & Veloso: Fall 2008

Exact Solution to POMDP (V)

Find Value Vectors (Assume $\gamma = 0.9$):

$$V_2^{A1}(b)_a = (p_1 + 1) + \gamma \cdot (1.6 - 0.3p_1)$$

$$= 2.44 + 0.73p_1$$

$$= [3.17, 2.44]$$

$$V_2^{A1}(b)_b = 2.746 + 1.027p_1 = [3.773, 2.746]$$

$$V_2^{A1}(b)_c = 2.314 + 1.243p_1 = [3.557, 2.314]$$

$$V_2^{A1}(b)_d = 2.62 + 1.54p_1 = [4.16, 2.62]$$

Simmons & Veloso: Fall 2008

Exact Solution to POMDP (VI)

Value Function for Action A2

$$V_2^{A2}(b)_a = (3 - 2p_1) + \gamma \cdot (1.8 - 0.7p_1)$$

$$= [1.99, 4.62]$$

$$V_2^{A2}(b)_b = 2.746 + 1.027p_1 = [2.791, 4.728]$$

$$V_2^{A2}(b)_c = 2.314 + 1.243p_1 = [2.719, 4.152]$$

$$V_2^{A2}(b)_d = 2.62 + 1.54p_1 = [3.52, 4.26]$$

Combining Value Function for A1 and A2

Horizon-Two Value Function:

$$\Psi_2 = \{[2.791, 4.728], [3.52, 4.26], [4.16, 2.62]\}$$

Simmons & Veloso: Fall 2008

Exact Solution to POMDP (VII)

Converges with:

- 5 alpha vectors
- 21.07 expected value

$r(A1, S1) = 2$
 $r(A2, S1) = 1$
 $r(A1, S2) = 1$
 $r(A2, S2) = 3$

$p(O1 | S1) = 0.9$
 $p(O2 | S1) = 0.1$
 $p(O1 | S2) = 0.5$
 $p(O2 | S2) = 0.5$

Simmons & Veloso: Fall 2008

Witness Algorithm (Littman, 1994)

- A **Witness** is a Counter-Example
 - Idea: Find places where the value function is suboptimal
 - Operates action-by-action and observation-by-observation to build up value (alpha) vectors
- Algorithm**
 - Start with value vectors for known ("corner") states
 - Define a linear program (based on Bellman's equation) that finds a point in the belief space where the value of the function is incorrect
 - Add a new vector (a linear combination of the old value function)
 - Iterate

Simmons & Veloso: Fall 2008

Witness Algorithm: Example

- Choose some belief that has the wrong value (by solving system of linear equations)
- Choose: $b_1 = [0.5, 0.5]$
- $V(b_1) = \max([2, 1] \cdot [0.5, 0.5], [1, 3] \cdot [0.5, 0.5]) = 2$
- $V_2^{A1}(b) = (p_1 + 1) + \gamma \cdot \{ \max_{v \in \Psi} (v \cdot \underline{b}'_1) + \max_{v \in \Psi} (v \cdot \underline{b}'_2) \}$
- $= (p_1 + 1) + \gamma \cdot \{ \max([2, 1] \cdot [(0.54 - 0.27p_1), (0.20 + 0.15p_1)], [1, 3] \cdot [(0.54 - 0.27p_1), (0.20 + 0.15p_1)]) + \max([2, 1] \cdot [(0.06 - 0.03p_1), (0.20 + 0.15p_1)], [1, 3] \cdot [(0.06 - 0.03p_1), (0.20 + 0.15p_1)]) \}$
- $V_2^{A1}(b) = 1.5 + \gamma \cdot \{ \max([2, 1] \cdot [0.405, 0.275], [1, 3] \cdot [0.405, 0.275]) + \max([2, 1] \cdot [0.045, 0.275], [1, 3] \cdot [0.045, 0.275]) \}$
- $= 1.5 + \gamma \cdot \{ \max(1.085, \underline{1.23}) + \max(0.365, \underline{0.87}) \}$
- Create new value vector using support vectors: $[1, 3], [1, 3]$

Simmons & Veloso: Fall 2008

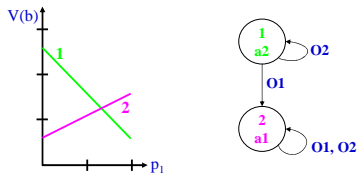
Policy Iteration for POMDPs

- Policy Iteration**
 - Choose a policy
 - Determine the value function, based on the current policy
 - Update the value function, based on Bellman's equation
 - Update the policy and iterate (if needed)
- Policy Iteration for POMDPs**
 - Original algorithm (Sondik) very inefficient and complex
 - Mainly due to evaluation of value function from policy!
 - Represent policy using finite-state controller (Hansen 1997):
 - Easy to evaluate
 - Easy to update

Simmons & Veloso: Fall 2008

POMDP Policy Iteration (Hansen 1997)

- Key Idea: Represent Policy as Finite-State Controller (**Policy Graph**)
 - Explicitly represents: “do action then continue with given policy”
 - Nodes correspond to vectors in value function
 - Edges correspond to transitions based on observations



Planning, Execution & Learning: POMDP

25

Simmons & Veloso: Fall 2008

POMDP Policy Iteration

$$V(b) = \max_a \{R(a, b) + \gamma \sum_{b'} p(b' | a, b) V(b')\}$$

- Associate actions with initial vectors: $\pi(v^i) = a$
- Determine the value function, based on the current policy
 - Solve system of linear equations
$$v^i(s) = r(s, \pi(v^i)) + \gamma \sum_{s', o} p(o | \pi(v^i), s') v^{(o, i)}(s')$$
- Update the value function, based on Bellman's equation
 - Can use any standard dynamic-programming method
- Update the policy ...

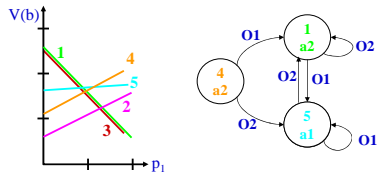
Planning, Execution & Learning: POMDP

26

Simmons & Veloso: Fall 2008

POMDP Policy Iteration

- Update the Policy
 - Ignore new vectors that are point-wise dominated by other vectors
 - New vectors that duplicate current vectors (same actions and observation links; point-wise equal)
 - Replace current vectors that are dominated by new vectors
 - Add new controller state otherwise



Planning, Execution & Learning: POMDP

27

Simmons & Veloso: Fall 2008