Overview

- POMDPs and their solutions
- Fixing memory with controllers
- Previous approaches
- Representing the optimal controller
- Some experimental results

POMDPs

- Partially observable Markov decision process (POMDP)
- Agent interacts with the environment
  - Sequential decision making under uncertainty
  - At each stage receives:
    - an observation rather than the actual state
    - Receives an immediate reward

POMDP definition

- A POMDP can be defined with the following tuple: $M = (S, A, P, R, \Omega, O)$
  - $S$, a finite set of states with designated initial state distribution $b_0$
  - $A$, a finite set of actions
  - $P$, the state transition model: $P(s'|s,a)$
  - $R$, the reward model: $R(s,a)$
  - $\Omega$, a finite set of observations
  - $O$, the observation model: $O(o|s',a)$
POMDP solutions

- A policy is a mapping $\delta : \Omega^* \rightarrow A$
- Goal is to maximize expected discounted reward over an infinite horizon
- Use a discount factor, $\gamma$, to calculate this

Example POMDP: Hallway

Minimize number of steps to the starred square for a given start state distribution

- States: grid cells with orientation
- Actions: turn $\leftarrow, \rightarrow$, move forward, stay
- Transitions: noisy
- Observations: red lines
- Goal: starred square

Previous work

- Optimal algorithms
  - Large space requirement
  - Can only solve small problems
- Approximation algorithms
  - provide weak optimality guarantees, if any

Policies as controllers

- Fixed memory
- Randomness used to offset memory limitations
- Action selection, $\psi : Q \rightarrow \Delta A$
- Transitions, $\eta : Q \times A \times O \rightarrow \Delta Q$
- Value given by Bellman equation:

$$V(q,s) = \sum_a P(a \mid q) \left[ R(s,a) + \gamma \sum_{s'} P(s' \mid s,a) \sum_o P(o \mid s',a) \sum_{q'} P(q' \mid q,a,o) V(q',s') \right]$$
**Controller example**

- Stochastic controller
  - 2 nodes, 2 actions, 2 obs
  - Parameters
    - \( P(a|q) \)
    - \( P(q'|q,a,o) \)

**Optimal controllers**

- How do we set the parameters of the controller?
  - Deterministic controllers - traditional methods such as branch and bound (Meuleau et al. 99)
  - Stochastic controllers - continuous optimization

**Gradient ascent**

- Gradient ascent (GA) - Meuleau et al. 99
  - Create cross-product MDP from POMDP and controller
  - Matrix operations then allow a gradient to be calculated

**Problems with GA**

- Incomplete gradient calculation
- Computationally challenging
- Locally optimal
**BPI**

- Bounded Policy Iteration (BPI) - Poupart & Boutilier 03
- Alternates between improvement and evaluation until convergence
- Improvement: For each node, find a probability distribution over one-step lookahead values that is greater than the current node’s value for all states
- Evaluation: Finds values of all nodes in all states

**BPI - Linear program**

For a given node, q
Variables: $x(a) = P(a|q)$, $x(q',a,o) = P(q',a|q,o)$
Objective: Maximize $\epsilon$
Improvement Constraints: $\forall s \in S$

$$V(q,s) + \epsilon \leq \sum_a x(a)R(s,a) + \gamma \sum_{s'} P(s'|s,a) \sum_q O(s,q,a) \sum_{x(q',a,o)} x(q',a,o) V(q',s')$$

Probability constraints: $a \in A$ $\sum_q x(q',a,o) = x(a)$

Also, all probabilities must sum to 1 and be greater than 0

**Problems with BPI**

- Difficult to improve value for all states
- May require more nodes for a given start state
- Linear program (one step lookahead) results in local optimality
- Must add nodes when stuck

**QCLP optimization**

- Quadratically constrained linear program (QCLP)
- Consider node value as a variable
- Improvement and evaluation all in one step
- Add constraints to maintain valid values
QCLP intuition

- Value variable allows improvement and evaluation at the same time (infinite lookahead)
- While iterative process of BPI can “get stuck” the QCLP provides the globally optimal solution

QCLP representation

Variables: \( x(q', a, q, o) = P(q', a|q, o) \), \( y(q, s) = V(q, s) \)
Objective: Maximize \( \sum_p y(q, s) \)
Value Constraints: \( \forall s \in S, q \in Q \)
\[
y(q, s) = \sum_{s'} \left[ \sum_{q'} x(q', a, q, o) \right] P(s', a) + \sum_{s'} P(s'|s, a) \sum_{o} O(o|s', a) \sum_{q'} x(q', a, q, o) y(q', s')
\]
Probability constraints: \( \forall q \in Q, a \in A, o \in \Omega \)
\[
\sum_{s} x(q', a, q, o) = \sum_{s'} x(q', a, q, o)
\]
Also, all probabilities must sum to 1 and be greater than 0

Optimality

Theorem: An optimal solution of the QCLP results in an optimal stochastic controller for the given size and initial state distribution.

Pros and cons of QCLP

- Pros
  - Retains fixed memory and efficient policy representation
  - Represents optimal policy for given size
  - Takes advantage of known start state
- Cons
  - Difficult to solve optimally
Experiments
- Nonlinear programming algorithm (snopt) - sequential quadratic programming (SQP)
- Guarantees locally optimal solution
- NEOS server
- 10 random initial controllers for a range of sizes
- Compare the QCLP with BPI

Results
- Computation time is comparable to BPI
- Increase as controller size grows offset by better performance

<table>
<thead>
<tr>
<th># nodes</th>
<th>QCLP</th>
<th>BPI</th>
<th># nodes</th>
<th>QCLP</th>
<th>BPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&lt; 1 min</td>
<td>&lt; 1 min</td>
<td>1</td>
<td>&lt; 1 min</td>
<td>1.3 min</td>
</tr>
<tr>
<td>2</td>
<td>&lt; 1 min</td>
<td>&lt; 1 min</td>
<td>2</td>
<td>&lt; 1 min</td>
<td>4.6 min</td>
</tr>
<tr>
<td>4</td>
<td>&lt; 1 min</td>
<td>&lt; 1 min</td>
<td>4</td>
<td>7.9 min</td>
<td>14.1 min</td>
</tr>
<tr>
<td>6</td>
<td>1.4 mins</td>
<td>1.8 mins</td>
<td>6</td>
<td>42.4 mins</td>
<td>25.5 mins</td>
</tr>
<tr>
<td>8</td>
<td>6.9 mins</td>
<td>2.9 mins</td>
<td>8</td>
<td>57.5 mins</td>
<td>42.9 mins</td>
</tr>
<tr>
<td>10</td>
<td>9.1 mins</td>
<td>4 mins</td>
<td>10</td>
<td>130.8 mins</td>
<td>62.8 mins</td>
</tr>
</tbody>
</table>

(a) best and (b) mean results of the QCLP and BPI on the hallway domain (57 states, 21 obs, 5 acts)

(a) best and (b) mean results of the QCLP and BPI on the machine maintenance domain (256 states, 16 obs, 4 acts)
Conclusion

- Introduced new fixed-size optimal representation
- Showed consistent improvement over BPI with a locally optimal solver
- In general, the QCLP may allow small optimal controllers to be found
- Also, may provide concise near-optimal approximations of large controllers

Future Work

- Investigate more specialized solution techniques for QCLP formulation
- Greater experimentation and comparison with other methods
- Extension to the multiagent case