RL 14: Simplifications of POMDPs

Michael Herrmann

University of Edinburgh, School of Informatics

04/03/2016

POMDPs: Points to remember

- Belief states are probability distributions over states
- Even if computationally complex, POMDPs can be useful as a modelling approach (consider simplification of the implementation in a second stage)
- POMDPs enable agents to deal with uncertainty efficiently
- POMDPs are Markovian w.r.t. belief states
- Beliefs tend to blur as consequence of the state dynamics, but can refocus by incorporating observations via Bayes' rule.
- Policy trees take all possible realisations of the sequence of future observations into account, i.e. the choice of the current action depends on the average over many futures.
- This causes exponential complexity unless the time horizon is truncated (standard) or approximations are used (e.g. $\mathcal{Q}$MDP, AMPD, and sample-based methods).
- Often some states are fully observable and these may be the states where decisions are critical (e.g. a robot turning when observing a doorway)

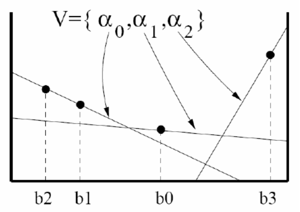
Simplifications of POMDPs

Often simiplifications and approximations are used:

- PBVI: Point-based value iteration
- α vectors
- $\mathcal{Q}$MDPs
- AMDPs: Augmented MPDs
- Monte Carlo POMDPs (last time)

Point Based Value Iteration

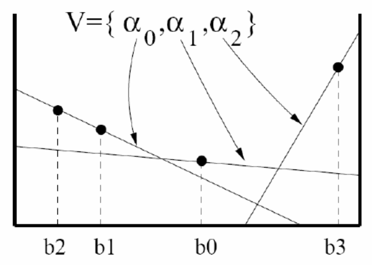
- Maintains a set of example beliefs
- Only considers constraints that maximise value function for at least one of the examples



- Solve POMDP for finite set of belief points
  - Initialise linear segment for each belief point and iterate
- Occasionally add new belief points
  - Add point after a fixed horizon
  - Add points when improvements fall below a threshold
  - Add points implied by belief update if sufficiently different from present set

Point Based Value Iteration

- Solve POMDP for finite set of belief points



- Can do point updates in polynomial time
  - Modify belief update so that one vector is maintained per point
  - Simplified by finite number of belief points
- Does not require pruning!
  - Only need to check for redundant vectors

J. Pineau, G. Gordon, and S. Thrun, Point-based value iteration: An anytime algorithm for POMDPs. International joint conference on artificial intelligence. Vol. 18. Lawrence Erlbaum Associates Ltd, 2003.

Value iteration ($\gamma = 1$) for α vectors

$$\begin{aligned}
V_t(b) &= \max_{a\in\mathcal{A}} \left(\sum_{s\in\mathcal{S}} b(s) \sum_{s'\in\mathcal{S}} T(s'|s,a) \sum_{o\in\Omega} \Omega(o|s',a) \left(R^a_{ss'o} + V_{t-1}\left(b^o_a(s')\right)\right)\right) \\
&= \max_{a\in\mathcal{A}} \left(\sum_{s\in\mathcal{S}} b(s) R(s,a) + \sum_{s\in\mathcal{S}} b(s) \sum_{s'\in\mathcal{S}} T(s'|s,a) \sum_{o\in\Omega} \Omega(o|s',a) V_{t-1}\left(b^o_a(s')\right)\right) \\
&= \max_{a\in\mathcal{A}} \left(\sum_{s\in\mathcal{S}} b(s) R(s,a) + \sum_{o\in\Omega} \max_k \sum_{s\in\mathcal{S}} b(s) \sum_{s'\in\mathcal{S}} T(s'|s,a) \Omega(o|s',a) \alpha^k_{t-1}(s')\right) \\
&= \max_{a\in\mathcal{A}} \left(\sum_{s\in\mathcal{S}} b(s) \left(\underbrace{R(s,a) + \sum_{o\in\Omega} \sum_{s'\in\mathcal{S}} T(s'|s,a) \Omega(o|s',a) \alpha^{l(b,a,o)}_{t-1}(s')}_{\alpha^k_t(s)} \right)\right)
\end{aligned}$$

Notes: t is the iteration index, the current state is s, the next state is s'. Ω is the likelihood of an observation, T is the transition probability due to an action, $b = b(s)$ is the current belief state, b^o_a is the belief after the next action and observation. The αs are meant to provide a more compact representation.

Algorithm POMDP(T) (based on a set of points x_i)

```
Υ = {(0, ..., 0)}, U = ∅
for τ = 1 to T do
    Υ' = ∅
    for all (U; α_1^k, ..., α_N^k) in Υ do
        for all control actions u do
            for all measurements z do
                for j = 1 to N do
                    α_{j,u,z}^k = Σ_{i=1}^N α_i^k p(z|x_i) p(x_i|u, x_j)
                endfor
            endfor
        endfor
    endfor
    for all control actions u do
        for all k = 1 to |Υ| do
                for i = 1 to N do
                    α'_i = r(x_i, u) + γ Σ_z α_{i,u,z}^k
                endfor
            add u to U and (U; α'_1, ..., α'_N) to Υ'
            endfor
        endfor
    optional: prune Υ'
    Υ=Υ'
endfor
return Υ
```

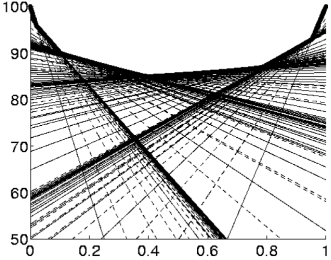
$\Upsilon = \{(0,\ldots,0)\}$, $\mathcal{U} = \emptyset$
for $\tau = 1$ to T do
  $\Upsilon' = \emptyset$
  for all $(\mathcal{U};\ \alpha_1^k,\ldots,\alpha_N^k)$ in Υ do
    for all control actions u do
      for all measurements z do
        for $j = 1$ to N do
          $\alpha_{j,u,z}^k = \sum_{i=1}^N \alpha_i^k\, p(z|x_i)\, p(x_i|u, x_j)$
        endfor
      endfor
    endfor
  endfor
  for all control actions u do
    for all $k = 1$ to $|\Upsilon|$ do
        for $i = 1$ to N do
          $\alpha_i' = r(x_i, u) + \gamma \sum_z \alpha_{i,u,z}^k$
        endfor
      add u to $\mathcal{U}$ and $(\mathcal{U}; \alpha_1',\ldots,\alpha_N')$ to Υ'
      endfor
    endfor
  optional: prune Υ'
  $\Upsilon = \Upsilon'$
endfor
return Υ

Remarks on the algorithm

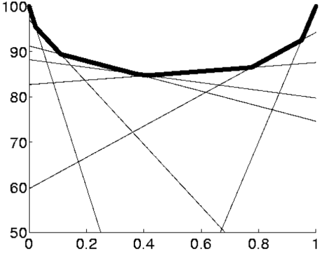
- Without pruning $|\Upsilon|$ increases exponentially with T
- The algorithm describes the determination of the value function. Value iteration, actual observations and actions are not entering.
- Further steps in algorithm
    - Find value function on policy trees up to a given T
    - Determine maximum over branches and perform first action
    - Recalculate policy taking into account observations and rewards
    - Update observation model, transition model and reward model
- Many variants exist.

Point-based Value Iteration

- Value functions for $T = 30$



Exact value function



PBVI

$\mathcal{Q}$MDPs

- $\mathcal{Q}$MDPs only consider state uncertainty in the first step (in a sense, similar to $\mathcal{Q}$-learning:)
- After that, the world is assumed to become fully observable.

Algorithm $\mathcal{Q}\mathrm{MDP}(b = (p_1, \ldots, p_N))$

$\hat{V}$ = MDP_DiscreteValueIteration()

for all control actions u and states x_i do

$$\mathcal{Q}(x_i, u) = r(x_i, u) + \sum_{j=1}^{N} \hat{V}(x_j)\, p(x_j | u, x_i)$$

end for

return $u' = \arg\max_u \sum_{i=1}^{N} p_i \mathcal{Q}(x_i, u)$

Augmented MDPs

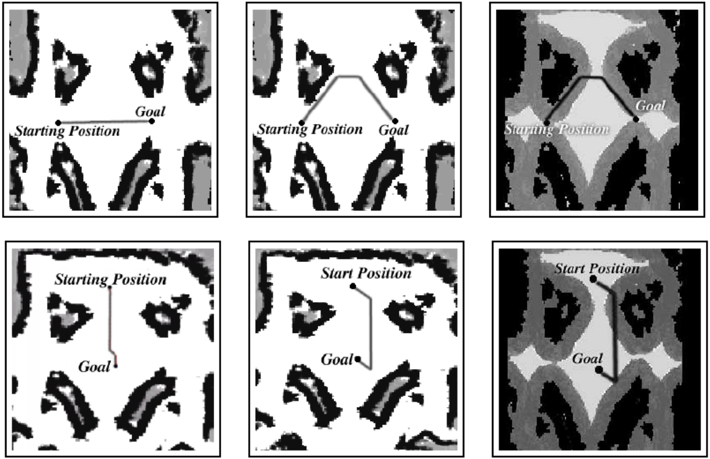
- Augmentation adds uncertainty component to state space, e.g.,

$$\bar{b} = \left(\begin{array}{c} \arg\max_x b(x) \\ H_b(x) \end{array} \right) \text{ with } H_{b(x)} = -\int b(x) \log b(x) \, dx$$

- Planning is performed by MDP in augmented state space
- Transition, observation and payoff models have to be learnt

N. Roy and S. Thrun, Coastal navigation with mobile robots. In NIPS 12, 1999.

Coastal Navigation by AMDPs (museum environment)



(a) Conventional (b) Coastal (c) Sensor Map

see: Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. MIT press.

What is Missing in POMDPs?

- POMDPs do not describe natural metrics in environment
  - When driving, we know both global and local distances
- POMDPs do not natively recognise differences between scales
  - Uncertainty in control is entirely different from uncertainty in routing
- POMDPs conflate properties of the environment with properties of the agent
  - Roads and buildings behave differently from cars and pedestrians: we need to generalise over them differently
- POMDPs are defined in a global coordinate frame, often discrete
  - We may need many different representations in real problems

References

- Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. MIT press. Chapters 15 and 16. (text book)
- Milos Hausknecht (2000) Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research* **13**, 33-94. (detailled paper)
- Joelle Pineau (2013) A POMDP Tutorial. *European Workshop on Reinforcement Learning. (review on recent research)*
- The POMDP Page (www.pomdp.org)
- Tony's POMDP Page)cs.brown.edu/research/ai/pomdp)