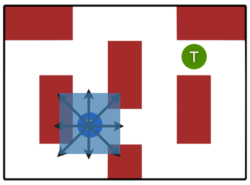
Towards Large-Scale POMDP Planning for Robotic Tasks

David Hsu
National University of Singapore

Partially observable Markov decision process (POMDP)

- A discrete POMDP model
  - States (configurations)
  - Actions — Unknown
  - Observations
  - Rewards
  - State transition function
  - Observation function — Known
- A belief state is a probability distribution over the states.
- A policy is a mapping from a belief to an action. An optimal policy maximizes the expected total reward.



Some history: 1978

OPERATIONS RESEARCH
Vol. 26, No. 2, March–April 1978
0030-364X/78/2602-0282 $01.25

The Optimal Control of Partially Observable Markov Processes over the Infinite Horizon: Discounted Costs

EDWARD J. SONDIK
Stanford University, Stanford, California

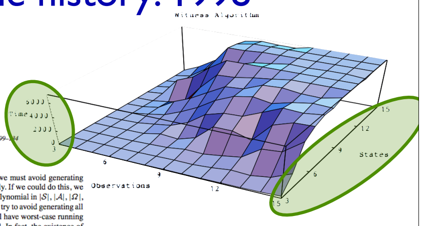
(Received July 1973; accepted May 1977)

This paper treats the discounted cost, optimal control problem for Markov processes with incomplete state information. The optimization approach for these partially observable Markov processes is a generalization of the well-known policy iteration technique for finding optimal stationary policies for completely observable Markov processes. The state space for the problem is the space of state occupancy probability distributions (the unit simplex). The development of the algorithm introduces several new ideas, including the class of finitely transient policies, which are shown to possess piecewise linear cost functions. The paper develops easily implemented approximations to stationary policies based on these finitely transient policies and shows that the concave hull of an approximation can be included in the well-known Howard policy improvement algorithm with subsequent convergence. The paper closes with a detailed example illustrating the application of the algorithm to the two-state partially observable Markov process.

THIS PAPER studies the control of Markov processes for which only

Drake (1962)
Astrom (1965)
Aoki (1965)
Smallwoord & Sondik (1971)

Some history: 1998



4.4. The witness algorithm

To improve the complexity of the value-iteration algorithm, we must avoid generating $\mathcal{V}_t^+$; instead, we would like to generate the elements of $\mathcal{V}_t$ directly. If we could do this, we might be able to reach a computation time per iteration that is polynomial in $|\mathcal{S}|$, $|\mathcal{A}|$, $|\Omega|$, $|\mathcal{V}_{t-1}|$, and $|\mathcal{V}_t|$. Cheng [10] and Smallwood and Sondik [56] also try to avoid generating all of $\mathcal{V}_t^+$ by constructing $\mathcal{V}_t$ directly. However, their algorithms still have worst-case running times exponential in at least one of the problem parameters [34]. In fact, the existence of an algorithm that runs in time polynomial in $|\mathcal{S}|$, $|\mathcal{A}|$, $|\Omega|$, $|\mathcal{V}_{t-1}|$, and $|\mathcal{V}_t|$ would settle the long-standing complexity-theoretic question "Does NP = RP?" in the affirmative [34], so we will pursue a slightly different approach.

Instead of computing $\mathcal{V}_t$ directly, we will compute, for each action a, a set $\mathcal{Q}_t^a$ of t-step policy trees that have action a at their root. We can compute $\mathcal{V}_t$ by taking the union of the $\mathcal{Q}_t^a$ sets for all actions and pruning as described in the previous section. The *witness* algorithm is a method for computing $\mathcal{Q}_t^a$ in time polynomial in $|\mathcal{S}|$, $|\mathcal{A}|$, $|\Omega|$, $|\mathcal{V}_{t-1}|$, and $|\mathcal{Q}_t^a|$ (specifically, run time is polynomial in the size of the inputs, the outputs, and an important intermediate result). It is possible that the $\mathcal{Q}_t^a$ are exponentially larger than $\mathcal{V}_t$, but this seems to be rarely the case in practice.

In what sense is the witness algorithm superior to previous algorithms for solving POMDPs, then? Experiments indicate that the witness algorithm is faster in practice over a wide range of problem sizes [34]. The primary complexity-theoretic difference is that the witness algorithm runs in polynomial time in the number of policy trees in $\mathcal{Q}_t^a$. There are example problems that cause the other algorithms, although they never construct the $\mathcal{Q}_t^a$'s

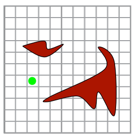
Kaelbling, Littman & Cassandra (1998)

Complexity of solving POMDPs

- Solving POMDPs exactly is computationally intractable:
  - Finite-horizon POMDPs are PSPACE-complete [Papadimitriou & Tsisiklis, 87].
  - Infinite-horizon POMDPs are undecidable [Madani et al., 99].

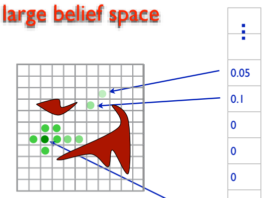
“Curse of dimensionality”

large state space



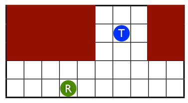
exponential in the dimensionality of the state space

large belief space

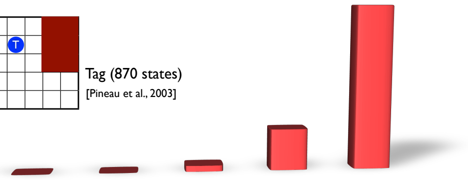


exponential in the number of states; doubly exponential in the dimensionality of the state space

Point-based POMDP algorithms

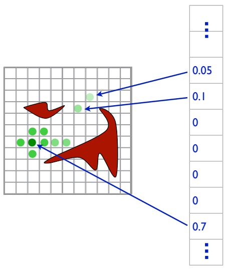


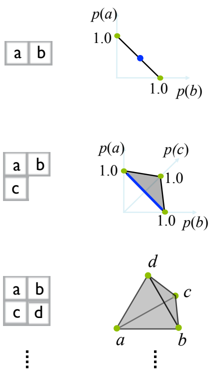
Tag (870 states)
[Pineau et al., 2003]



	PBVI 2003 Pineau et al.	HSVI 2003 Smith & Simmons	Perseus 2005 Spaan & Vlassis	HSVI2 2005 Smith & Simmons	SARSOP 2008 Hsu et al.	....
Time (sec.)	180,880	10,113	1,670	24	**6**	....
Reward	-9.18	-6.17	-6.17	-6.36	**-6.13**	

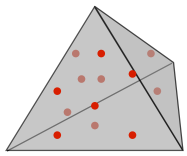
Belief space



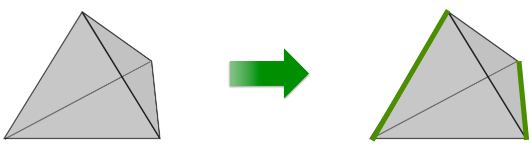


Sampling the belief space

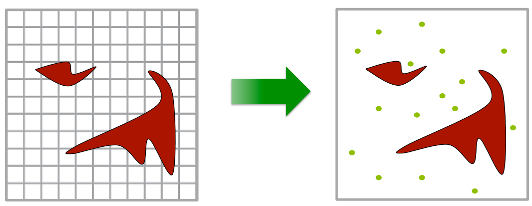
- Point-based POMDP algorithms



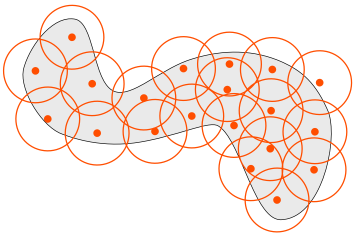
Factoring



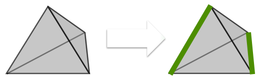
Sampling the state space



Covering number



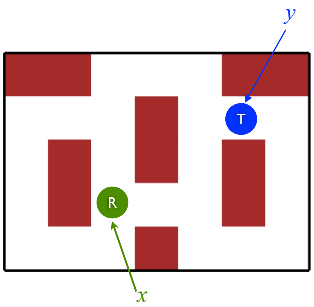
Mixed Observability Markov Decision Process (MOMDP)



S.C.W. Ong, S.W. Png, D. Hsu, and W.S. Lee. POMDPs for robotic tasks with mixed observability. *Int. J. Robotics Research*, 29(8), 2010.

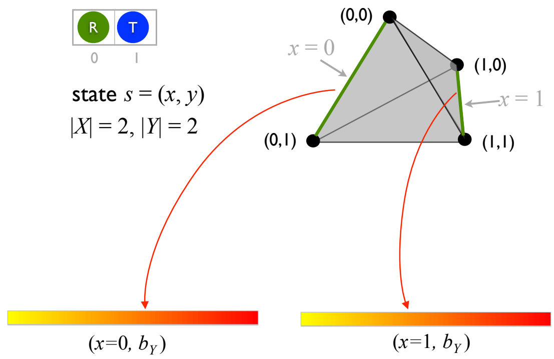
Mixed observability Markov decision process (MOMDP)

- state $s = (x, y)$
- POMDP belief $b(s)$
- MOMDP belief $(x, b(y))$



Factoring the belief space

state $s = (x, y)$
$|X| = 2$, $|Y| = 2$

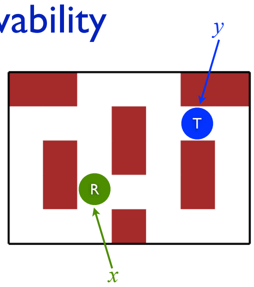


Mixed observability

- state $s = (x, y)$
  number of states 50 x 50 = 2,500
- belief $b(s)$
  2,500-dimensional space
- belief $(x, b(y))$
  union of 50-dimensional subspaces

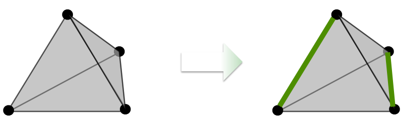
50x

- Potential efficiency gain

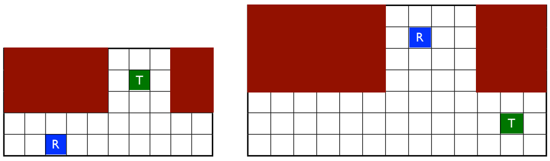


Computational efficiency

- POMDP operates in a single $|S|$-dimensional space.
- MOMDP operates in a union of $|X|$ sets of $|Y|$-dimensional spaces, where $|S| = |X||Y|$.
- Computational efficiency gain from MOMDP representation
  - Point-based approximation algorithms $\propto |X|$

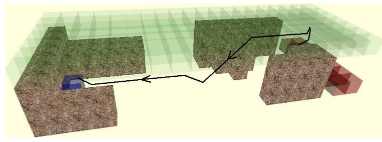


Tag
(Pineau, Gordon & Thrun 2003)



	Tag (29-position map)		Tag (55-position map)	
	$\lvert S\rvert = 870$		$\lvert S\rvert = 3{,}080$	
	$\lvert X\rvert = 30, \lvert Y\rvert = 29$		$\lvert X\rvert = 56, \lvert Y\rvert = 55$	
MOMDP	**5** sec		**19** sec	
	-6.03		-9.90	
SARSOP	17 sec	3.4x	736 sec	38.7x
(Hsu et. al. 08)	-6.03		-9.90	

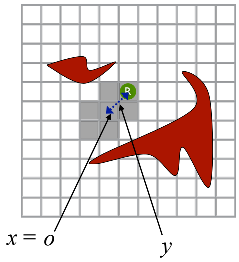
AUV navigation in 3D



	AUV Navigation (140 hpos x 4 depth x 24 orien)	
	$\lvert S\rvert = 13{,}536$	
	$\lvert X\rvert = 96, \lvert Y\rvert = 141$	
MOMDP	**124** sec	
	1020	
SARSOP	409 sec	3.3x
	1020	

Reparameterized full observability

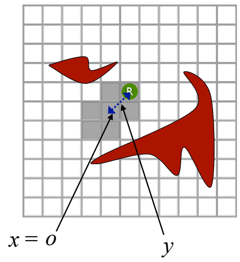
- Reparameterize the state space
- $x = o$
- $h(o)$: the set of states that have non-zero probability of emitting o
- y = offset from o, indicating the exact state in $h(o)$



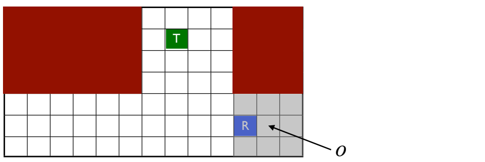
Reparameterized full observability

Theorem

The POMDP and the reparameterized MOMDP (with $x = o$) are equivalent.



Noisy Tag
Reparameterized full observability

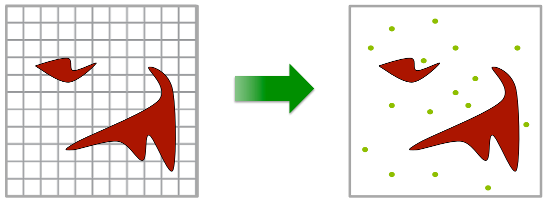


	$\lvert S\rvert = 3{,}080$	
	$\lvert X\rvert = 56, \lvert Y\rvert = 495$	
MOMDP	**32** sec	
	-10.6	
SARSOP	**927** sec	29.0x
	-10.6	

Continuous-state POMDPs

H.Y. Bai, D. Hsu, and W.S. Lee. Monte Carlo Value Iteration for Continuous-State POMDPs. In *Proc. Int. Workshop on the Algorithmic Foundations of Robotics*, 2010.

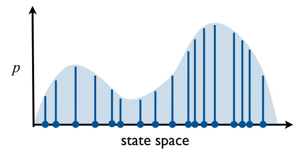
Sampling the state space



PRM (Kavraki et al. 1996)
EST (Hsu et al. 1999)
RRT (LaValle & Kuffner 2001)

Continuous-state POMDPs

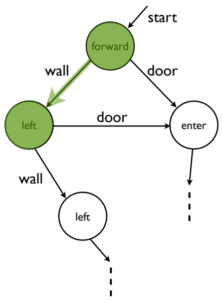
- Belief over continuous state space



Particle filter
MC-POMDP (Thrun, 2000)
Perseus (Porta et al., 2006)

Continuous-state POMDPs

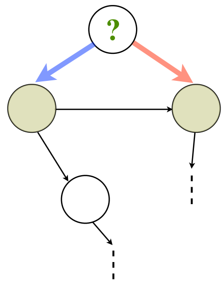
- Policy graph (finite-state controller)



(Hansen, 2000)

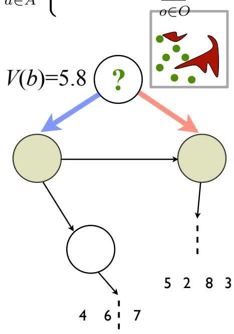
Value iteration on a policy graph

$$V_{t+1}(b) = \max_{a \in A} \left\{ R(b,a) + \gamma \sum_{o \in O} p(o|b,a) V_t(b') \right\}$$

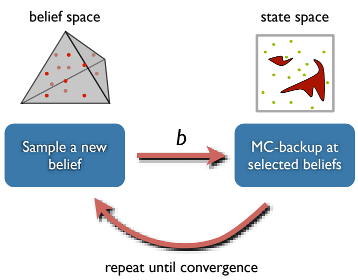


Monte Carlo backup

$$V_{t+1}(b) = \max_{a \in A} \left\{ R(b,a) + \gamma \sum_{o \in O} p(o|b,a) V_t(b') \right\}$$



Monte Carlo Value Iteration (MCVI)



Computational efficiency

Theorem

$$|V^*(b) - V_t(b)| \leq \sqrt{\frac{|O| + \ln|A| + \ln(1/\tau)}{N}} + \delta_B + \gamma^t$$

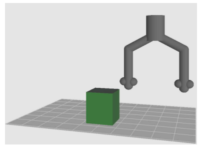
with probability at least $1 - \tau$.

- N : number of samples from the state space (Monte Carlo simulations)
- δ_b : covering of the belief space
- t : number of iterations

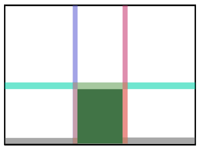
Grasping

(Hsiao, Kaelbling & Lozano-Perez 2007)

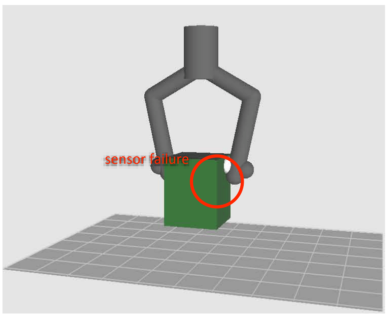
- Uncertain initial position
- Noisy touch sensors on fingers



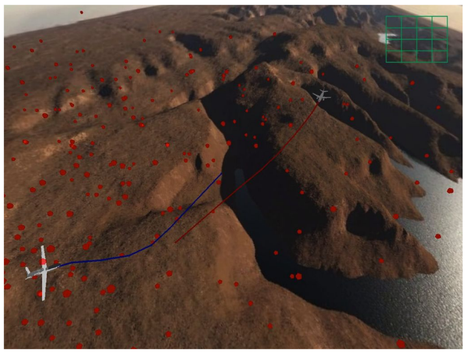
- Manual discretization



Grasping



Aircraft collision avoidance



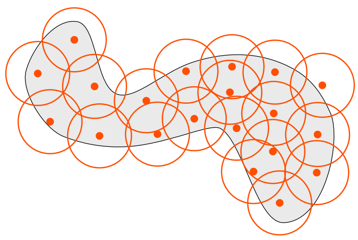
Simulation results

Model	Algorithm	Risk ratio
3D continuous-state POMDP	MCVI	0.00066
2D continuous-state POMDP	MCVI	0.017
2D discrete POMDP (Temizer et al. 2010)	SARSOP	0.035
TCAS	TCAS	0.061
Nominal		1.0

53x

- MIT Lincoln Laboratory CASSATT simulator
- 15,000 encounters from 9 months of radar data in US airspace

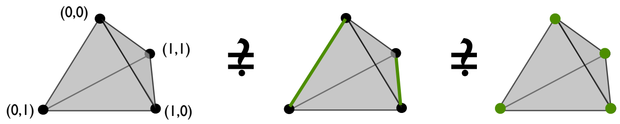
What makes some POMDPs easier than others?



D. Hsu, W.S. Lee, and N. Rong. What makes some POMDP problems easy to approximate? In Advances in Neural Information Processing Systems (NIPS), 2007.

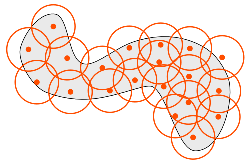
Common complexity measures

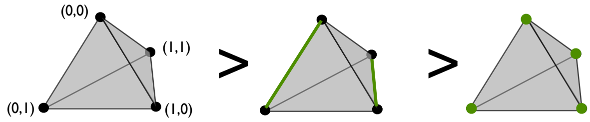
- Number of states = dimensionality of belief space



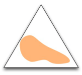
An alternative complexity measure

- The **covering number** $C(\delta)$ of a set S is the number of balls of radius δ needed to cover S completely.





Reachable space



- Find an approximately optimal **action** at b_0 (on-line action selection)

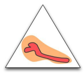
Theorem

An approximately optimal value $V(b_0)$ with regret no more than ε can be found in time

$$O\left(\mathcal{C}\left(\frac{(1-\gamma)^2\varepsilon}{4\gamma R_{\max}}\right)^2 \log_\gamma \frac{\varepsilon(1-\gamma)}{2R_{\max}}\right)$$

- The problem is easy if the covering number (“volume”) of the reachable space is small.

Optimally reachable space



Theorem

Finding the optimal action is NP-hard, even if the optimally reachable space has a polynomial-size cover.

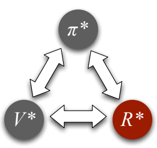
Theorem

Given a suitable cover C of the optimally reachable space, an approximately optimal value $V(b_0)$ with regret no more than ε can be found in time

$$O\left(|C|^2 + |C| \log_\gamma \frac{(1-\gamma)\varepsilon}{2R_{\max}}\right)$$

Implications

- Together, the positive and negative results indicate that finding a suitable cover of the optimally reachable space is a key difficulty.
- The covering number better characterizes the complexity of the problem by capturing the sparsity of the space.



Bounding the covering number

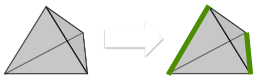
- Several properties, often encountered in practice, reduce the size of covering numbers.
  - Fully observable state variables

$$\left(\frac{k^d}{\delta}\right)^{k^d} \Rightarrow k^{d'}\left(\frac{k^{d-d'}}{\delta}\right)^{k^{d-d'}}$$

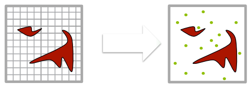
  - Sparse beliefs
  - Smooth beliefs
  - Circulant state-transition matrices
  - ...

Summary

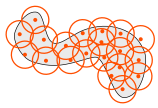
- Large belief space: MOMDP



- Large state space: MCVI



- Covering number

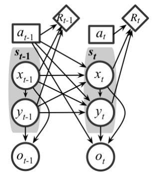


POMDP software

Approximate POMDP Planning (APPL) Toolkit

http://bigbird.comp.nus.edu.sg/pmwiki/farm/appl/index.php?n=Main.HomePage

```
<Variable>
     <StateVar vnamePrev="rover_0" vnameCurr="rover_1"
      fullyObs="true">
         <NumValues>3</NumValues>
     </StateVar>
     <StateVar vnamePrev="rock_0" vnameCurr="rock_1>"
         <ValueEnum>good bad</ValueEnum>
     </StateVar>
     <ObsVar vname="obs sensor">
         <ValueEnum>ogood obad</ValueEnum>
     </ObsVar>
     <ActionVar vname="action_rover">
         <ValueEnum>amw ame ac as</ValueEnum>
     </ActionVar>
     <RewardVar vname="reward rover" />
</Variable>
```



Future Work

Acknowledgments

- Wee Sun Lee, NUS
- Mykel Kochenderfer
  MIT Lincoln Laboratory
- Hanna Kurniawati, SMART
- Sylvie Ong, McGill
- Haoyu Bai, NUS
- Yanzhu Du, Stanford
- Shaowei Png, McGill
- Nan Rong, Cornell

Acknowledgments

- MDA Gambit Game Lab
- Ministry of Education, Singapore
- School of Computing, National University of Singapore