Fast approximate planning in POMDPs

Geoff Gordon

ggordon@cs.cmu.edu

Joelle Pineau, Geoff Gordon, Sebastian Thrun. *Point-based value iteration: an anytime algorithm for POMDPs*

Overview

POMDPs are too slow

Overview

~~POMDPs are too slow~~

Overview

Review of POMDPs

Review of POMDP value iteration algorithms

Point-based value iteration

Theoretical results

Actual results

POMDP overview

Planning in an uncertain world

Actions have random effects

Don't observe full world state

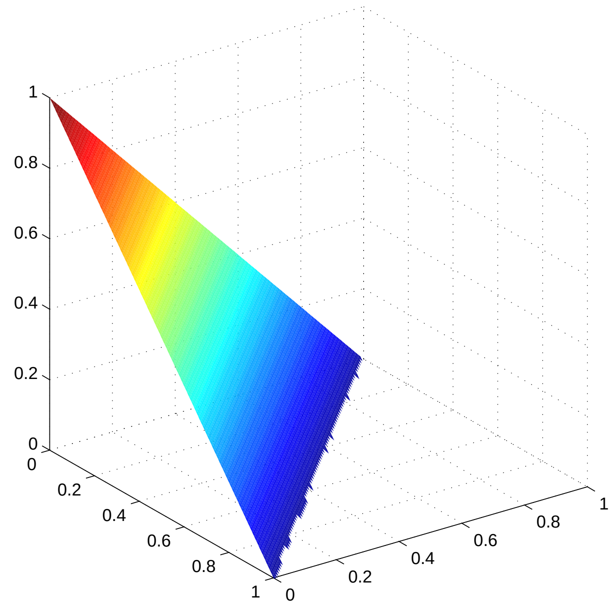
POMDP definition

State $x \in X$, actions $a \in A$, observations $z \in Z$

Rewards r_a (column vectors), discount $\gamma \in [0, 1)$

Belief $b \in P(X)$ (row vectors)

Starting belief b_0



POMDP definition cont'd

Transitions $b \to bT_a$ (T_a stochastic)

Observation likelihoods w_z (row vectors)

$$\sum_z w_z = \mathbf{1}$$

Observation update:

$$b \leftarrow w_z \times b \cdot \eta$$

where $\times$ is pointwise multiplication

Value functions

Just like MDP value function (but bigger)

$V(b)$ = expected total discounted future reward starting from b

Knowing V means planning is 1-step lookahead

If we discretize belief simplex, we are “done”

From b get to $b_{z_1}, b_{z_2}, \ldots$ according to $P(z \mid b, a)$

Value functions

Additional structure: convexity

Consider beliefs $b_1, b_2, b_3 = \frac{b_1+b_2}{2}$

b_3: flip a coin, then start in b_1 if heads, b_2 if tails

b_3 is always worse than average of b_1, b_2

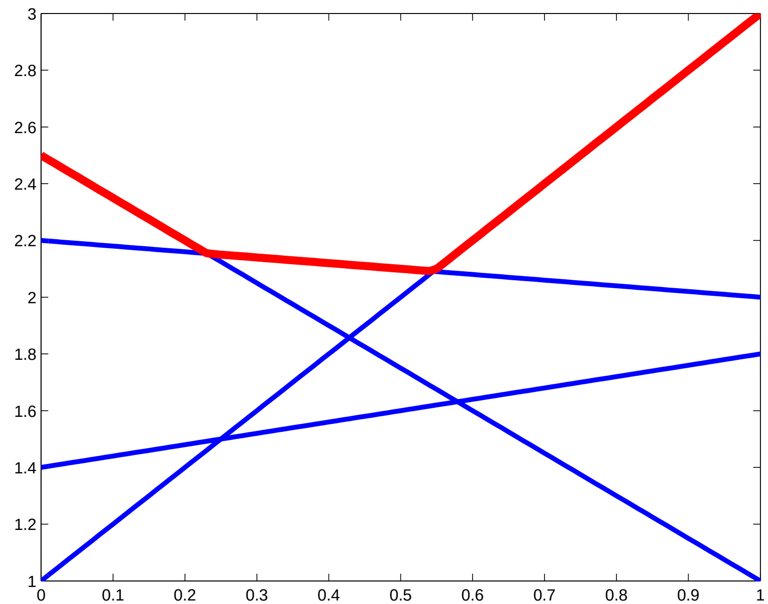
Representation

Represent V as the upper surface of a (possibly infinite) set of hyperplanes

$\mathcal{V}$ is set of hyperplanes

Hyperplanes represented by normals v (column vectors)

$$V(b) = \max_{v \in \mathcal{V}} b \cdot v$$



Value iteration

Bellman's equation:

$$V(b) = \max_a Q(b, a)$$

$$Q(b, a) = r_a + \gamma \sum_z P(z \mid b, a) V(b_{az})$$

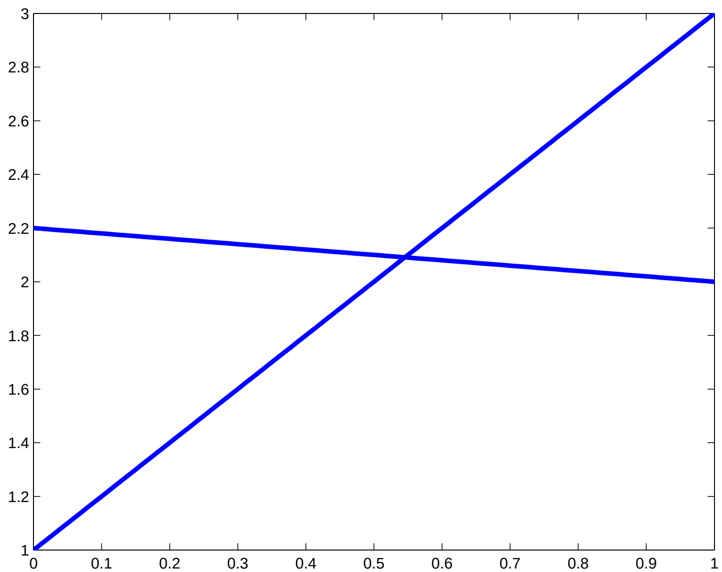
where $b_{az} = \eta(bT_a) \times w_z$

Convergence

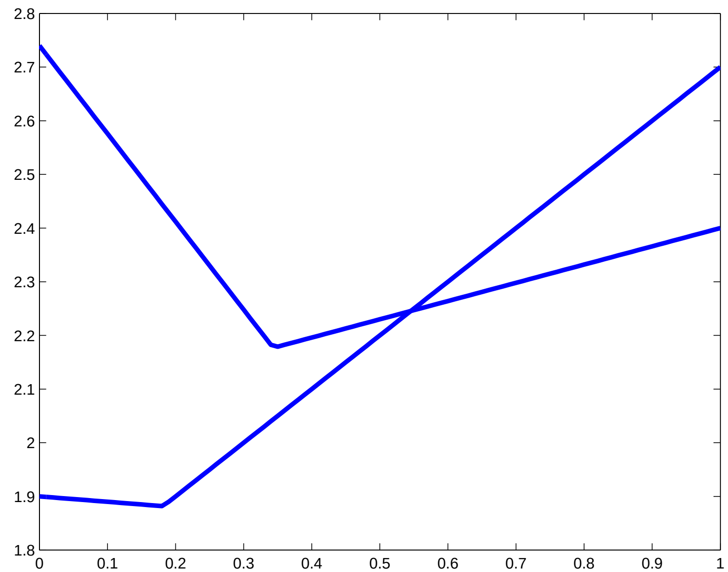
Backup operator T: $V \leftarrow TV$

T is a contraction on $P(X) \mapsto \mathbb{R}$

$\|b - b'\| = \max_x |b(x) - b'(x)|$



$\longmapsto$



Sondik's algorithm (1972)

Rearrange Bellman equation to make it linear:

$\eta^{-1} = P(z \mid b, a)$, and $V(\eta b) = \eta V(b)$, so

$$\begin{aligned} Q(b,a) &= r_a + \gamma \sum_z V((bT_a) \times w_z) \\ &= r_a + \gamma \sum_z \max_{v \in \mathcal{V}} ((bT_a) \times w_z) \cdot v \\ &= r_a + \gamma \sum_z \max_{v \in \mathcal{V}} b \cdot T_a (w_z \times v) \end{aligned}$$

Evaluate from inside out

Suppose $V_t(b) = b \cdot v$

$v_z = w_z \times v$

$v_{az} = \gamma T_a v_z$

$v_a = v_{az_1} + v_{az_2} + \ldots$

$\mathcal{V}' = \{v_{a_1}, v_{a_2}, \ldots\}$

Now $V_{t+1}(b) = \max_{v \in \mathcal{V}'} b \cdot v$

More than 1 hyperplane

Suppose $V_t(b) = \max_{v \in \mathcal{V}} b \cdot v$

$\mathcal{V}_z = w_z \times \mathcal{V}$ set ops are elementwise

$\mathcal{V}_{az} = \gamma T_a \mathcal{V}_z$

$\mathcal{V}_a = r_a + \mathcal{V}_{az_1} \oplus \mathcal{V}_{az_2} \oplus \ldots$ *expensive!*

$\mathcal{V}' = \mathcal{V}_{a_1} \cup \mathcal{V}_{a_2} \cup \ldots$

Now $V_{t+1}(b) = \max_{v \in \mathcal{V}'} b \cdot v$

above representation due to [Cassandra et al]

A note on complexity

Or, some very large numbers

Set	Comment	Total size	Time/element
$\mathcal{V}_z$	same size as $\mathcal{V}$	$\lvert Z\rvert\,\lvert\mathcal{V}\rvert$	$O(\lvert X\rvert)$
$\mathcal{V}_{az}$	still same size	$\lvert A\rvert\,\lvert Z\rvert\,\lvert\mathcal{V}\rvert$	$O(\lvert X\rvert^2)$
$\mathcal{V}_a$	big!	$\lvert A\rvert\,\lvert\mathcal{V}\rvert^{\lvert Z\rvert}$	$O(\lvert X\rvert)$

For example, w/ 5 actions, 5 observations:

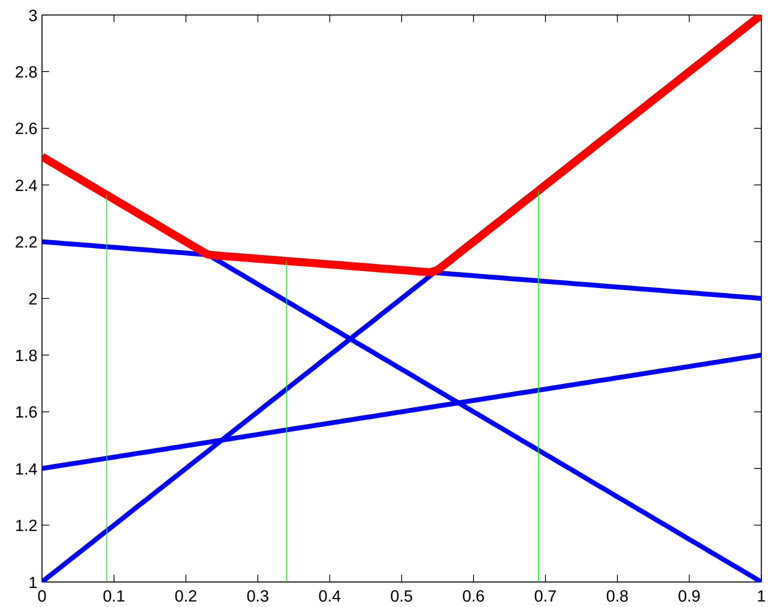
$$1,\ 5,\ 15625,\ 4.6566 \times 10^{21},\ 1.0948 \times 10^{109},\ \ldots$$

Witnesses (Littman 1994)

Don't need all elements of $\mathcal{V}$

Just those which are $\arg\max b \cdot v$ for some b

If we have the b (a *witness*), fast to check that v is indeed $\arg\max$



Witness details

Linear feasibility problem (size about $|\mathcal{V}| \times |X|$)

$$\begin{aligned} b \cdot v &\geq b \cdot v_i \qquad \forall i \\ b \cdot \mathbf{1} &= 1 \\ b &\geq 0 \end{aligned}$$

Solve one LF per element of $\mathcal{V}$—expensive, but well worth it

Can add margin $\epsilon > 0$ for approximate solution

- don't have to have all witnesses

Incremental pruning

(Cassandra, Littman, Zhang 1997)

Prune $\mathcal{V}_z$, $\mathcal{V}_{az}$, and $\mathcal{V}_a$ as they are constructed

Another big win in runtime

We are now up to 16-state POMDPs

Summary so far

Solve POMDPs by repeatedly applying backup T

Represent V with set of hyperplanes $\mathcal{V}$

$\mathcal{V}$ grows fast

Can prune $\mathcal{V}$ using witnesses

Plan for rest of talk

Better use of witnesses: point backups

Better way to find witnesses: exploration

PBVI = point backups + exploration for witnesses

PBVI examples

Backups at a point

Computing witnesses is expensive

What if we knew a witness b already?

Fast to compute both $V(b)$ and $\frac{d}{db}V(b)$

Intuitive, then formal derivation

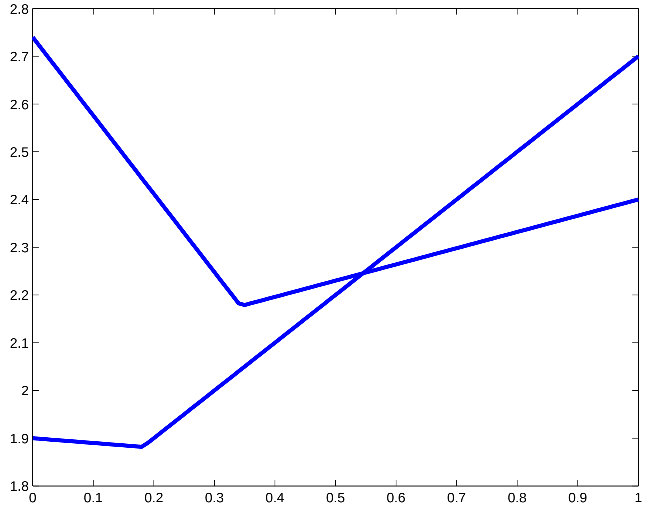
Point backup—intuition

$V(b')$ depends on $P(z \mid b, a)b_{az}$ for all a, z

$P(z \mid b, a)b_{az}$ are linear functions of b

$V(P(z \mid b, a)b_{az})$ is scaled/shifted copy of V

Adding these copies: hard over $P(X)$, easy at b



Point backup—math

When $\mathcal{V} \to \mathcal{V}'$, we want $\max_{v \in \mathcal{V}'} b \cdot v$

That's $\max_a \max_{v \in \mathcal{V}_a} b \cdot v$, since $\mathcal{V}' = \mathcal{V}_{a_1} \cup \mathcal{V}_{a_2} \ldots$

But $\max_{v \in \mathcal{V}_a} b \cdot v$ is

$$\max_{v_1 \in \mathcal{V}_{az_1}} b \cdot v_1 + \max_{v_2 \in \mathcal{V}_{az_2}} b \cdot v_2 + \ldots$$

since any $v \in \mathcal{V}_a$ is $v_1 + v_2 + \ldots$

$\ldots$ and $\mathcal{V}_{az}$ is quick to compute.

Advantage of point-based backups

- Suppose we have a set B of witnesses and $\mathcal{V}$ of hyperplanes
- Pruning $\mathcal{V}$ takes time $O(|B|\,|\mathcal{V}|\,|X|)$ (w/ small constant)
- Without knowing witnesses, solve $|\mathcal{V}|$ LFs, each $|\mathcal{V}| \times |X|$
- Higher order, worse constants

Where do witnesses come from?

Grids (note difference to discretizing belief simplex)

Random (Poon 2001)

Interleave point-based with incremental pruning (Zhang & Zhang 2000)

We are now up to 90-state POMDPs

New theorem

Bound error of the point-based backup operator

Bound depends on how densely we sample reachable beliefs

Probably exists an extension to “easily reachable” beliefs

Error bound on one step + contraction of value iteration = overall error bound

First result of this sort for POMDP VI

Definitions

Let Δ be the set of reachable beliefs

Let B be a set of witnesses

Let $\epsilon(B)$ be the worst-case density of B in Δ:

$$\epsilon(B) = \max_{b' \in \Delta} \min_{b \in B} \|b - b'\|_1$$

Theorem

A single point-based backup's error is

$$\frac{\epsilon(B)(R_{\max} - R_{\min})}{1-\gamma}$$

That means the error after value iteration is

$$\frac{\epsilon(B)(R_{\max} - R_{\min})}{(1-\gamma)^2}$$

plus a bit for stopping at finite horizon

Policy error

We therefore have that policy error is:

$$\frac{\epsilon(B)(R_{\max} - R_{\min})}{(1-\gamma)^3}$$

$(1-\gamma)^3$, ouch! But it does go to 0 as $\epsilon(B) \longrightarrow 0$

Exploration

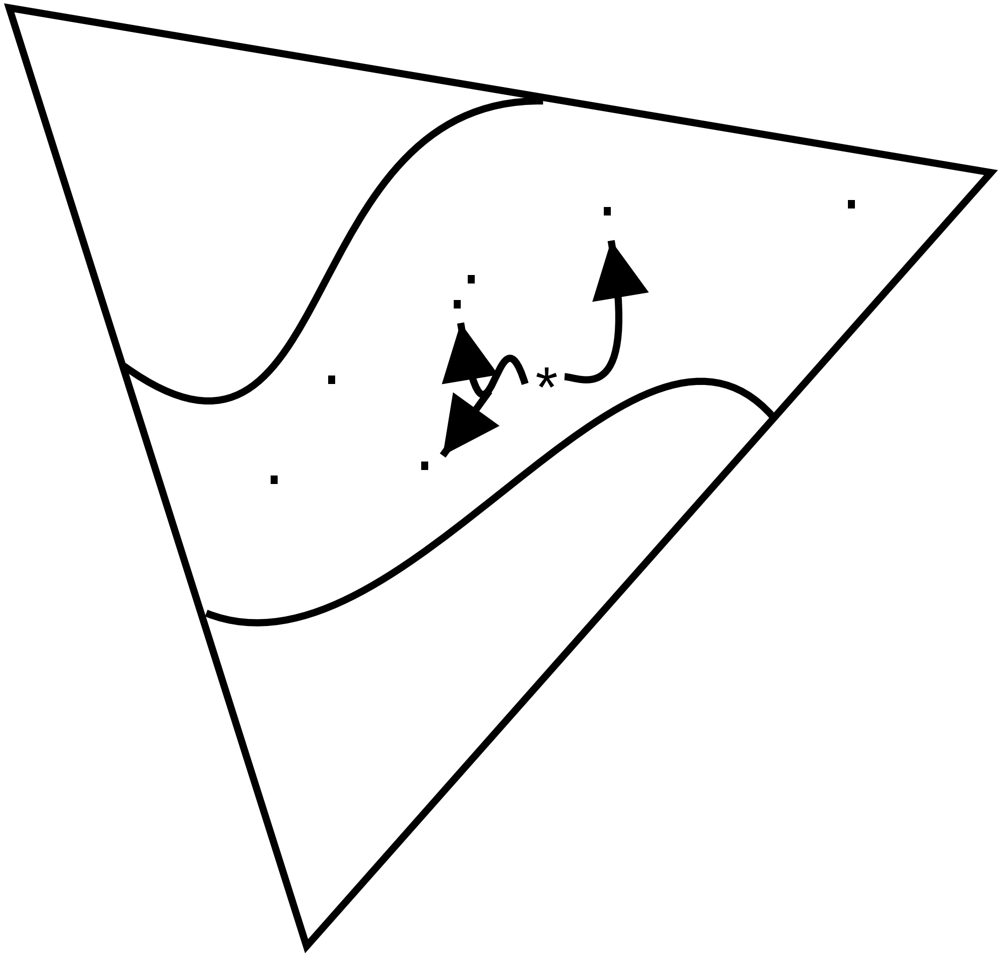
Theorem tells us we want to sample reachable beliefs with high worst-case 1-norm density

We can do this by simulating forward from b_0

Generate a set of candidate witnesses

Accept those which are farthest (1-norm) from current set

Selecting new witnesses



Summary of algorithm

$B \leftarrow \{b_0\}$

$\mathcal{V} = \{0\}$ (or whatever—e.g., use QMDP)

Do some point-based backups on $\mathcal{V}$ using B

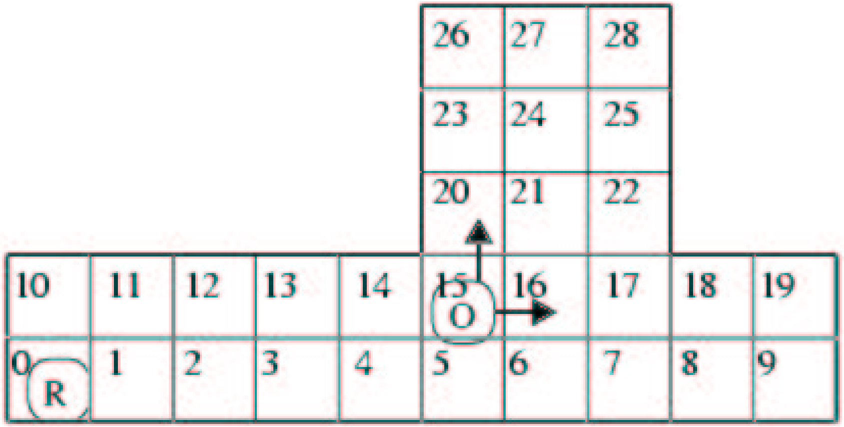
- we backup k times, where γ^k is small

Add more beliefs to B

- we double the size of B each time

Repeat

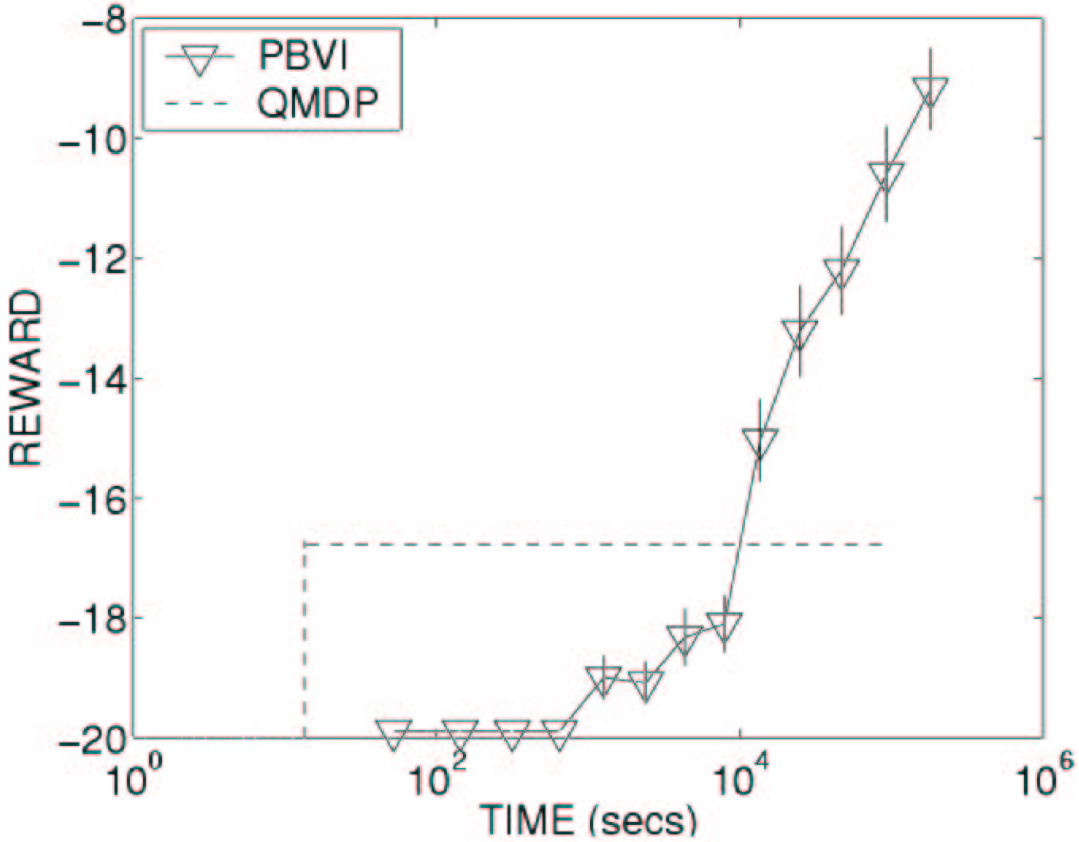
Tag problem



870 states, 2$\times$29 observations, 5 actions

fixed opponent policy

Results



Results

Catches opponent 60% of time

Don’t know of another value iteration algorithm which could do this well

On smaller problems, gets policies as good as other algorithms

But uses a small fraction of the compute time

Contributions and Conclusion

Others have used point-based backups

- mostly in combination with other, more expensive ops

Others have tried to select witnesses quickly

- on small problems, random & grid are good heuristics

Pushed to $10\times$ larger problems with efficient algorithm and intelligent search for witnesses

Our theorem is the strongest of its type