Planning under Partial Observability by Classical Replanning: Theory and Experiments

Blai Bonet[a] and Héctor Geffner[b,c]

[a]Universidad Simón Bolívar, Caracas (Venezuela), [b]Universitat Pompeu Fabra, Barcelona (Spain), [c]Institució Catalana de Recerca i Estudis Avançats, Barcelona (Spain)

Problem

Solve: planning problems with partial observability

Difficulties

- tracking belief states instead of states
- search for policy trees instead of linear plans

Options

- better belief representations and heuristics
- **restrictions that permit more efficient solution methods**

Contribution:

- framework for planning for partial observability that spells these restrictions and exploits efficient classical planning methods

Related Work and Ideas

- Freespace assumption [Koenig, 2005]
- Preferred values of variables [Likhachev & Stentz, 2009]
- Safe assumptions in LTL checking [Albore & Bertoli, 2006]
- Model knowledge at propositional level [Petrick & Bacchus, 2002]
- Compilation [Palacios & Geffner, 2009; Albore et al., 2009]
- **Planning under optimism** and **optimism in the face of uncertainty**

Language and Restrictions

Extension of STRIPS with conditional effects, negation, uncertain initial situation, and **sensing**. A problem is tuple $P = \langle F, O, I, G, M \rangle$ where

- F is set of fluent symbols
- O is set of actions; each action a has precondition and conditional effects of the form $a : C \rightarrow L$
- I is set of F-clauses defining initial situation
- G is set of F-literals defining the goal
- M is set of **sensors** (C, L): if C becomes true, value of L is observed

Restrictions: Simple Partially Observable Problems

- non-unary clauses in I are **invariant**
- no hidden fluent appears in the body of a conditional effect

Theorem: reachable beliefs b for simple problems represented by just the literals true in b + invariants

Translation into Classical Planning

For simple problem $P = \langle F, O, I, G, M \rangle$, the translation $K(P)$ is classical problem $K(P) = \langle F', O', I', G' \rangle$ where

- $F' = \{KL, K\neg L : L \in F\}$ (knowledge literals)
- $I' = \{KL : I \models L\}$
- $G' = \{KL : L \in G\}$
- $O' = O_{\text{sup}} \cup O_{\text{can}} \cup O_{\text{inv}} \cup O_{\text{sen}}$ where
  - $O_{\text{sup}} = \{a : KC \rightarrow KL : a : C \rightarrow L \in O\}$ (supports)
  - $O_{\text{can}} = \{a : \neg K\neg C \rightarrow \neg KL : a : C \rightarrow L \in O\}$ (cancellations)
  - $O_{\text{inv}} = \{KC \rightarrow KL : \neg C \vee L \in I\}$ (invariants)
  - $O_{\text{sen}} = \{A(C, L), A(C, \neg L) : \text{sensor } (C, L) \in M\}$ (assumptions)

where $A(C, L)$ is action w/ precondition $KC, \neg KL, \neg K\neg L$ and effect KL
(Note: for $C = L_1 \wedge L_2 \cdots$, $KC = KL_1 \wedge KL_2 \cdots$ and $\neg K\neg C = \neg K\neg L_1 \wedge \neg K\neg L_2 \cdots$)

Translation $K(P)[s]$

For knowledge state s, $K(P)[s]$ is equal to $K(P)$ with $I' = s$

- A sensor (C, L) **gets activated** in a plan for $K(P)[s]$ when it makes KC true when KL and $K\neg L$ are false
- A plan for $K(P)[s]$ always **activates a sensor** or **achieves goal**

K-Replanner

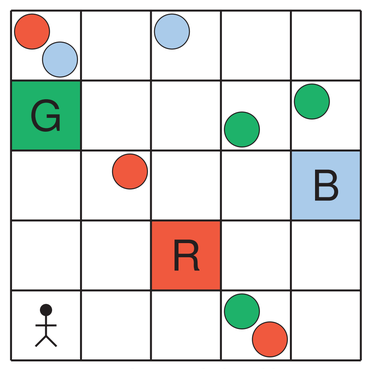
1. **Let** s be the initial state in $K(P)$
2. **While** s is not a goal in $K(P)$ **do**
3. Compute classical plan π for classical problem $K(P)[s]$
4. **Set** $\pi' :=$ **executable prefix** of π
5. **Set** $s' :=$ result of applying π' to $K(P)[s]$
6. Add observations to s' and **close s' with invariants**
7. **Set** $s := s'$

- The executable prefix of π is shortest prefix of π that achieves goal or activates a sensor
- Optimized to shortest prefix that achieves goal or yields observation $\neg L$ that **refutes** an action-assumption $A(C, L)$ in the rest of the plan

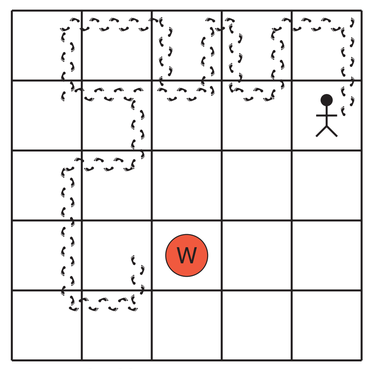
Theorem (Soundness): for simple problems P, if the K-Replanner terminates, it terminates in a goal state of P

Theorem (Completeness): if P is a simple problem with a **connected space** and the non-unary clauses in I are in **prime implicate form**, then the K-Replanner is guaranteed to terminate in a goal state of P

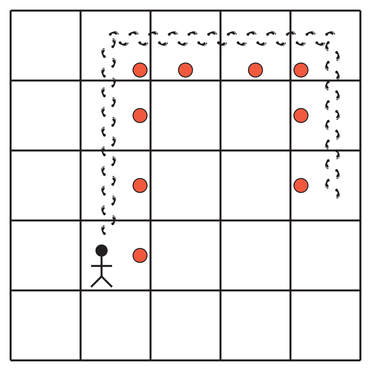
Examples and Results



colored-balls



kill-wumpus


trail

Benchmark

- **freespace**: the agent needs to reach a final position by moving through unknown terrain. Each cell of the grid is either blocked or clear. As the agent moves, it senses the status of adjacent cells.
- **doors**: the agent moves in a grid to reach a final position, yet it has to cross doors whose positions are only partially known.
- **wumpus**: the agent moves in a grid, with deadly *wumpuses*. to reach a goal position. As the agent moves, it smells the stench of nearby wumpuses.
- **kill-wumpus**: a variation of the above in which the agent must locate and kill the *single* wumpus in the grid.
- **colored-balls**: the agent navigates a grid to pick up and deliver balls of different colors. The positions and colors of the balls are unknown, but when the agent is at a position, he observes if there are balls, and if so, their colors.
- **trail**: the agent must follow a trail of stones til the end. The shape and length of the trail are not known. The agent cannot get off trail but can observe the presence of stones in adjancent cells.

K-Replanner using FF

domain	#inst.	#solved	avg. length	avg. #calls	avg. srch. time
freespace	140	134	26.49	8.31	8.02
doors	90	69	132.96	85.17	11.98
wumpus	70	70	34.60	2.39	1.11
kill-wumpus	30	30	49.20	31.10	1.92
colored-balls	75	26	115.81	27.54	26.51
trail	140	133	22.77	8.18	7.54

K-Replanner using Mp on colored-balls

instance	#states	length	#calls	srch. time	total time
9×9-15balls-#2	3.68e39	298	65	190.4	428.9
9×9-15balls-#4	3.68e39	369	73	816.1	1,164.8
9×9-16balls-#1	1.19e42	283	64	454.9	743.6
9×9-16balls-#2	1.19e42	293	71	554.7	998.8
9×9-17balls-#2	3.87e44	339	69	721.7	1,082.2
9×9-18balls-#4	1.25e47	406	69	533.6	857.0