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), \(^c\)Istitució 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 \(P = (F, O, I, G, M)\) 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 \(b + \text{invariants}\)

**Translation into Classical Planning**

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

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

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

**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 \(K(P)[s]\) do
3. Compute classical plan \(\pi\) for classical problem \(K(P)[s]\)
4. Set \(\pi' := \text{executable prefix of } \pi\)
5. Set \(s' := \text{result of applying } \pi' \text{ to } K(P)[s]\)
6. Add observations to \(s'\) and close \(s'\) with invariants
7. Set \(s := s'\)

- The executable prefix of \(\pi\) is shortest prefix of \(\pi\) 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**

<table>
<thead>
<tr>
<th>Domain</th>
<th>#states</th>
<th>length</th>
<th>#calls</th>
<th>avg. srch. time</th>
<th>avg. #calls</th>
<th>avg. #solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freespace</td>
<td>140</td>
<td>134</td>
<td>26.49</td>
<td>8.31</td>
<td>8.02</td>
<td></td>
</tr>
<tr>
<td>Doors</td>
<td>90</td>
<td>69</td>
<td>132.96</td>
<td>85.17</td>
<td>11.98</td>
<td></td>
</tr>
<tr>
<td>Wumpus</td>
<td>70</td>
<td>70</td>
<td>34.60</td>
<td>2.39</td>
<td>1.11</td>
<td></td>
</tr>
<tr>
<td>Kill-Wumpus</td>
<td>30</td>
<td>30</td>
<td>49.20</td>
<td>31.10</td>
<td>1.92</td>
<td></td>
</tr>
<tr>
<td>Colored-Balls</td>
<td>75</td>
<td>26</td>
<td>115.81</td>
<td>27.54</td>
<td>26.51</td>
<td></td>
</tr>
<tr>
<td>Trail</td>
<td>140</td>
<td>133</td>
<td>22.77</td>
<td>8.18</td>
<td>7.54</td>
<td></td>
</tr>
</tbody>
</table>

**K-Replanner using FF**

<table>
<thead>
<tr>
<th>Domain</th>
<th>#inst.</th>
<th>#solved</th>
<th>avg. length</th>
<th>avg. #calls</th>
<th>avg. srch. time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freespace</td>
<td>140</td>
<td>134</td>
<td>26.49</td>
<td>8.31</td>
<td>8.02</td>
</tr>
<tr>
<td>Doors</td>
<td>90</td>
<td>69</td>
<td>132.96</td>
<td>85.17</td>
<td>11.98</td>
</tr>
<tr>
<td>Wumpus</td>
<td>70</td>
<td>70</td>
<td>34.60</td>
<td>2.39</td>
<td>1.11</td>
</tr>
<tr>
<td>Kill-Wumpus</td>
<td>30</td>
<td>30</td>
<td>49.20</td>
<td>31.10</td>
<td>1.92</td>
</tr>
<tr>
<td>Colored-Balls</td>
<td>75</td>
<td>26</td>
<td>115.81</td>
<td>27.54</td>
<td>26.51</td>
</tr>
<tr>
<td>Trail</td>
<td>140</td>
<td>133</td>
<td>22.77</td>
<td>8.18</td>
<td>7.54</td>
</tr>
</tbody>
</table>

**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. 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 till 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 adjacent cells.

**K-Replanner using MP on colored-balls**

<table>
<thead>
<tr>
<th>Domain</th>
<th>#states</th>
<th>length</th>
<th>#calls</th>
<th>avg. srch. time</th>
<th>avg. #solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>9x9-15balls-#2</td>
<td>3.68e39</td>
<td>298</td>
<td>65</td>
<td>190.4</td>
<td>428.9</td>
</tr>
<tr>
<td>9x9-16balls-#2</td>
<td>3.68e39</td>
<td>369</td>
<td>73</td>
<td>816.1</td>
<td>1,164.8</td>
</tr>
<tr>
<td>9x9-16balls-#1</td>
<td>1.19e42</td>
<td>283</td>
<td>64</td>
<td>454.9</td>
<td>743.6</td>
</tr>
<tr>
<td>9x9-17balls-#2</td>
<td>3.97e44</td>
<td>339</td>
<td>69</td>
<td>721.7</td>
<td>1,082.2</td>
</tr>
<tr>
<td>9x9-18balls-#2</td>
<td>1.25e47</td>
<td>406</td>
<td>69</td>
<td>533.6</td>
<td>857.0</td>
</tr>
</tbody>
</table>