Explainable AI Planning: Overview and the Case of Contrastive Explanation

Part 2: Explaining the Space of Plans

Jörg Hoffmann and Daniele Magazzeni

September 23, 2019
Agenda

1. Introduction
2. Oversubscription Planning
3. Explanation Framework
4. Computing Explanations
5. Compilations
6. NoGood Learning in State Space
The Traditional View of AI Planning

(Figure from [Smith (2012)])
A Typical Reality: Interactive Decision Making

(Figure from [Smith (2012)])
The Problem

Why this plan and not another plan?

Human

AI
Our Approach: Plan-Property Dependencies

Why does this plan not have property A?

Because all plans with property A have property B!

Human

AI
Our Approach in Interactive AI Planning

(Figure adapted from [Smith (2012)])
Classical Planning

FDR Planning: Syntax

A finite-domain representation (FDR) task is a tuple \( \tau = (V, A, c, I, G) \):

- **V variables**, each \( v \in V \) with a finite domain \( D_v \); a **state** is a complete assignment to \( V \);
- **A actions**, each \( a \in A \) has precondition \( \text{pre}_a \) and effect \( \text{eff}_a \), both partial assignments to \( V \);
- **c** : \( A \rightarrow \mathbb{R}^+ \) action-cost function;
- **I initial state**; **G goal** partial assignment to \( V \);

FDR Planning: Semantics

Action \( a \) applicable in state \( s \) if \( \text{pre}_a \subseteq s \). Outcome state \( s[[a]] \) like \( s \) except that \( s[[a]](v) = \text{eff}_a(v) \) for those \( v \) on which \( \text{eff}_a \) is defined. Outcome state of iteratively applicable action sequence \( \pi \) denoted \( s[[\pi]] \).

Sequence \( \pi \) applicable in \( I \) is **plan** if \( G \subseteq I[[\pi]] \).
(Toy) Example: IPC Rovers

- \( \text{drive}(R_i, L_x, L_y) \)
- \( \text{takeImage}(I_x, R_y) \)

**goal:**

\[
\begin{align*}
I_1 & \quad I_2 & \quad I_3 \\
L_1 & \quad L_2 & \quad L_3 & \quad L_4 & \quad L_5 & \quad L_6
\end{align*}
\]
### Oversubscription Planning

<table>
<thead>
<tr>
<th>An OSP task is a tuple $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- $V$ variables, $A$ actions, $c$ action-cost function, $I$ initial state;</td>
</tr>
<tr>
<td>- $G^{\text{hard}}$ hard goal;</td>
</tr>
<tr>
<td>- $G^{\text{soft}}$ soft goal;</td>
</tr>
<tr>
<td>- $b \in \mathbb{R}_0^+$ cost budget.</td>
</tr>
</tbody>
</table>

$\pi = \langle a_1, \ldots, a_n \rangle$ is plan if $\sum_{i=1}^{n} c(a_i) \leq b$ and $G^{\text{hard}} \subseteq I[[\pi]]$.

**Plan quality:** Usually additive soft-goal rewards. Here:

- User preferences hard to specify/elicitate. Iterative planning instead.
- Goal-exclusion dependencies to support that process.
OSP and Our Explanation Problem in Rovers

planning task

soft goals

plan

\[
\text{drive}(R_1, L_1, L_2) \\
\text{takeImage}(I_1, R_1) \\
\text{drive}(R_2, L_4, L_5) \\
\text{takeImage}(I_3, R_2)
\]
Plan Properties

→ Plan properties over soft goals in OSP:

Plan Properties

OSP task $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$, $\Pi$ its set of plans.

- **Plan property**: propositional formula $\phi$ over atoms $g \in G^{\text{soft}}$
- **Conjunctive plan property**: $\phi$ has form $\bigwedge_{g \in A} g$ or $\neg \bigwedge_{g \in B} g$

Simple special case: In general, any function $\Pi \rightarrow \{true, false\}$

- e. g. temporal plan trajectory constraints.
- e. g. deadlines, resource bounds.

Compilation: Into (additional variables/actions and) goal facts!

- e. g. LTL formulas
- Here: action-set properties, easy special case of LTL
Π-Entailment

→ Π in the role of a knowledge base:

OSP task \( \tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b) \), \( \Pi \) its set of plans \( \pi \).

- \( \pi \) satisfies \( \phi \), \( \pi \models \phi \): if \( \phi \) true given truth value assignment to \( g \in G^{\text{soft}} \) defined by \( g \in I[[\pi]] \) ? \( g \mapsto \text{true} \) : \( g \mapsto \text{false} \)
- \( M_{\Pi}(\phi) := \{ \pi | \pi \in \Pi, \pi \models \phi \} \)
- \( \phi \) Π-entails \( \psi \), written \( \Pi \models \phi \Rightarrow \psi \): if \( M_{\Pi}(\phi) \subseteq M_{\Pi}(\psi) \)

→ Special case focus here:

Goal Exclusions

- Goal exclusion: entailment of form \( \Pi \models \bigwedge_{g \in A} g \Rightarrow \neg \bigwedge_{g \in B} g \)
- Non-dominated: \( \not\exists (A', B') \neq (A, B): \ A' \subseteq A, \ B' \subseteq B \), \( \Pi \models \bigwedge_{g \in A'} g \Rightarrow \neg \bigwedge_{g \in B'} g \)
Π-entailment:

\[ \text{Dominated } \Pi\text{-entailment:} \]

\[ \Rightarrow \]

\[ \begin{array}{c}
L_1 \\
L_2 \\
L_3 \\
L_4 \\
L_5 \\
L_6
\end{array} \]

\[ \text{goal:} \]

\[ \begin{array}{c}
I_1 \\
I_2 \\
I_3
\end{array} \]
Global Explanations

→ All entailment relations over plan properties in the task:

Global Explanation (GE)

OSP task $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$, $\Pi$ its set of plans.

- $[\phi]_\Pi$: equivalence class, i.e. set of $\psi$ with $M_\Pi(\phi) = M_\Pi(\psi)$

- Global explanation (GE) for $\tau$: strict partial order over equivalence classes, $[\phi]_\Pi < [\psi]_\Pi$ iff $[\phi]_\Pi \neq [\psi]_\Pi$ and $\Pi \models \phi \Rightarrow \psi$

→ More practical variant for goal exclusions:

Goal-Exclusion GE

OSP task $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$, $\Pi$ its set of plans.

- Goal-exclusion GE for $\tau$: strict partial order over conjunctive plan properties induced by the non-dominated goal exclusions

$$\Pi \models \bigwedge_{g \in A} g \Rightarrow \neg \bigwedge_{g \in B} g$$
Goal-Exclusion GE: Rovers Example

All non-dominated goal exclusions:
Local Explanations

→ In response to user question “Why not property $\phi$?":

Local Explanation (LE)

OSP task $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$, $\Pi$ its set of plans.

- Local explanation (LE) for $\phi$: $\{\psi | \Pi \models \phi \Rightarrow \psi\}$
- Goal-exclusion LE for $\phi = \bigwedge_{g \in A} g$:
  $\{\psi | \psi = \neg \bigwedge_{g \in B} g, \Pi \models \phi \Rightarrow \psi \text{ is non-rhs-dominated}\}$
- Non-rhs-dominated: $\not\exists B': B' \subsetneq B, \Pi \models \bigwedge_{g \in A} g \Rightarrow \neg \bigwedge_{g \in B'} g$

Remarks:

- Smaller and easier to compute than GE (see next section).
- Relative to current plan $\pi$ in iterative planning: only those $\psi$ where $\pi \not\models \psi$ (i.e. new properties entailed by $\phi$).
Goal-Exclusion LE: Rovers Example

Why does this plan not achieve ?

Because
Another Example: Transportation (IPC “NoMystery”)

Variables $V$: $T_0, f_0, T_1, f_1, P_0, P_1, P_2$

Actions $A$: $\text{drive}(T_i, L_x, L_y)$, $\text{load}(T_i, P_j, L_x)$, $\text{unload}(T_i, P_j, L_x)$

Driving consumes fuel as indicated

Initial state $I$: as shown;

$\textbf{Goal } G^{\text{soft}}$: $\text{at}(P_0, L_4), \text{at}(P_1, L_3), \text{at}(P_2, L_5)$

Non-dominated goal exclusions:

- $\Pi \models P_0 \Rightarrow \neg(P_1 \land P_2)$
- $\Pi \models P_1 \Rightarrow \neg(P_0 \land P_2)$
- $\Pi \models P_2 \Rightarrow \neg(P_0 \land P_1)$

$\rightarrow$ In other words: “$G^{\text{soft}}$ is not solvable as a whole, but each of its subsets is”.

Hoffmann and Magazzeni Explainable AI Planning 15th Reasoning Web Summer School (RW 2019) 22/55
Discussion/Literature/Challenges

**Framework intention:**
- Plan properties = language for (or finite set of) properties relevant to user preferences.
- Elucidate plan-property dependencies for interactive planning (instead of just fixing an optimization objective).
- Goal exclusions merely a simple starting point, yet powerful through compilation (see later).

**Positioning in literature:**
- "Does $\Pi \models \phi \Rightarrow \psi$?" = model checking of planning task.
  \[\Rightarrow\text{Framework} = \text{exhaustive model checking of entailments within a set } P \text{ of plan properties.}\]
- Working hypothesis: Meaningful concept/special case.
  Computation: Exploit relatedness across individual checks.
- Very little prior work on model checking for planning models [Vaquero et al. (2013)].
Why does this plan not achieve $\phi \Rightarrow \psi$?

Because $\phi \Rightarrow \psi$ is not true.

But why $\phi \Rightarrow \psi$ is not true?

"Why does $\Pi \models \phi \Rightarrow \psi$?"

- Idea 1: Extend set $P$ of plan properties to elucidate “the causal chain between” $\phi$ and $\psi$.
  - One instance of problem how to identify the relevant set $P$.

- Idea 2: Find minimal relaxation (superset) of $\Pi$ in which $\Pi \models \phi \Rightarrow \psi$ no longer holds.
  - $\phi \Rightarrow \psi$ no longer holds.
  - Drop hard goals, increase cost budget, ...
Non-Dominated Goal Exclusions from MUGS

Minimal Unsolvable Goal Subset (MUGS)

OSP task $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$, $\Pi$ its set of plans.

- Minimal unsolvable goal subset (MUGS): unsolvable $G \subseteq G^{\text{soft}}$, every $G' \varsubsetneq G$ solvable

Proposition (Non-dominated Goal Exclusions from MUGS)

Non-dominated $\Pi \models \bigwedge_{g \in A} g \Rightarrow \neg \bigwedge_{g \in B} g \iff A \cup B$ MUGS

Proof.

A $\Pi$-entailment $\Pi \models \bigwedge_{g \in A} g \Rightarrow \neg \bigwedge_{g \in B} g$ clearly holds iff $A \cup B$ is unsolvable. Non-dominated entailments result from set-inclusion minimal $A$ and $B$, corresponding to the set-inclusion minimality of MUGS.

→ Compute and represent goal-exclusion GE via MUGs.
Systematic Weakening

1. Start with $G^{\text{soft}}$
2. Select open node $G$, call planner to test solvability, cache result, expand $G$ if unsolvable
3. Children of $G$: $G' \subset G$ where $|G'| = |G| - 1$
## Experiments: Global Explanations

<table>
<thead>
<tr>
<th>domain</th>
<th>Reference Coverage</th>
<th>SysS/W Coverage</th>
<th>Search Fraction</th>
<th>#MUGS, $x =$ average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$h_{\text{LM-cut}}$ OSP 0.25 0.5 0.75</td>
<td>0.25 S W</td>
<td>0.5 S W</td>
<td>0.75 S W</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>agricola (20)</td>
<td>0 0 0 0</td>
<td>20 20 13 13 1 1</td>
<td>1 0.5</td>
<td>1 0.5</td>
</tr>
<tr>
<td>airport (50)</td>
<td>28 28 24 22</td>
<td>35 34 21 21 16 16</td>
<td>0.6 0.81 1 0.61</td>
<td>3.8 2 1.4</td>
</tr>
<tr>
<td>barman (34)</td>
<td>4 18 11 4</td>
<td>18 18 4 4 3 4</td>
<td>0.57 0.94 1 0.5</td>
<td>6.9 4.2 2.5</td>
</tr>
<tr>
<td>blocks (35)</td>
<td>28 35 28 21</td>
<td>35 35 29 29 26 26</td>
<td>0.15 0.97 0.8 0.64</td>
<td>11 12.4 13.7</td>
</tr>
<tr>
<td>childcare (20)</td>
<td>0 2 0 0</td>
<td>4 4 0 0 0 0</td>
<td>0.34 0.98 - -</td>
<td>16.8 - -</td>
</tr>
<tr>
<td>data-network (20)</td>
<td>12 13 13 13</td>
<td>20 20 18 18 17 15</td>
<td>0.72 0.73 0.91 0.66</td>
<td>2.1 1.8 1.5</td>
</tr>
<tr>
<td>depot (22)</td>
<td>7 16 11 7</td>
<td>16 16 9 10 3 3</td>
<td>0.24 0.96 0.89 0.68</td>
<td>8.3 7 6.5</td>
</tr>
<tr>
<td>driverlog (20)</td>
<td>13 15 13 10</td>
<td>15 15 12 12 10 10</td>
<td>0.17 0.98 0.87 0.5</td>
<td>8.1 16.1 11.1</td>
</tr>
<tr>
<td>elevators (50)</td>
<td>40 22 22 22</td>
<td>47 48 38 37 27 27</td>
<td>0.35 0.94 0.9 0.67</td>
<td>4.6 5.1 5.9</td>
</tr>
<tr>
<td>floorite (36)</td>
<td>13 18 6 2</td>
<td>8 8 2 2 2 2</td>
<td>0.1 0.99 0.96 0.3</td>
<td>316.2 137 45.5</td>
</tr>
<tr>
<td>freecell (80)</td>
<td>15 77 30 21</td>
<td>76 76 30 30 18 18</td>
<td>0.31 0.94 0.88 0.76</td>
<td>4 3.4 3.3</td>
</tr>
<tr>
<td>ged (20)</td>
<td>15 20 20 20</td>
<td>16 20 10 12 7 7</td>
<td>0.25 0.9 0.58 0.7</td>
<td>13.3 38.7 12.5</td>
</tr>
<tr>
<td>grid (5)</td>
<td>2 5 3 2</td>
<td>5 5 3 4 3 3</td>
<td>0.54 0.84 1 0.54</td>
<td>4 2.5 1</td>
</tr>
<tr>
<td>gripper (20)</td>
<td>7 11 8 8</td>
<td>5 5 4 4 3 3</td>
<td>0.21 0.98 0.96 0.46</td>
<td>783.5 228 156</td>
</tr>
<tr>
<td>hiking (20)</td>
<td>9 19 14 13</td>
<td>20 20 16 17 11 10</td>
<td>0.81 0.69 1 0.63</td>
<td>1.8 1.7 1</td>
</tr>
<tr>
<td>logistics (60)</td>
<td>26 27 20 16</td>
<td>15 15 6 6 3 4</td>
<td>0.35 0.95 0.98 0.73</td>
<td>7.2 7.3 2.8</td>
</tr>
<tr>
<td>miconic (150)</td>
<td>141 97 66 55</td>
<td>66 64 42 43 35 36</td>
<td>0.3 0.92 0.95 0.61</td>
<td>81.3 38.2 18.8</td>
</tr>
<tr>
<td>mprime (35)</td>
<td>22 35 27 24</td>
<td>35 35 35 35 35 35</td>
<td>0.9 0.59 0.94 0.59</td>
<td>1.3 1.2 1.1</td>
</tr>
<tr>
<td>mystery (30)</td>
<td>12 29 27 21</td>
<td>30 30 30 30 30 30</td>
<td>0.89 0.61 0.93 0.61</td>
<td>1.3 1.2 1.1</td>
</tr>
<tr>
<td>nomystery (20)</td>
<td>14 20 14 10</td>
<td>20 20 12 12 8 8</td>
<td>0.15 0.98 0.87 0.61</td>
<td>20.2 18.5 5.8</td>
</tr>
<tr>
<td>openstacks (77)</td>
<td>47 63 56 52</td>
<td>49 43 45 39 42 35</td>
<td>0.03 0.99 0.12 0.98</td>
<td>15.3 14.9 10.3</td>
</tr>
<tr>
<td>organic-syn-s (13)</td>
<td>10 8 8 8</td>
<td>8 8 8 8 6 6</td>
<td>0.19 0.96 0.28 0.91</td>
<td>5.3 7.3 8.3</td>
</tr>
<tr>
<td>parcprinter (26)</td>
<td>24 26 22 18</td>
<td>10 14 10 14 10 12</td>
<td>0.44 0.98 0.73 0.85</td>
<td>5.6 7.5 4.1</td>
</tr>
<tr>
<td>parking (40)</td>
<td>5 25 5 0</td>
<td>17 12 1 1 0 0</td>
<td>0.02 1 - -</td>
<td>63.9 31</td>
</tr>
<tr>
<td>pathways (30)</td>
<td>5 5 4 4</td>
<td>7 7 5 5 4 4</td>
<td>0.41 0.86 0.91 0.7</td>
<td>11.3 3.8 1.8</td>
</tr>
<tr>
<td>pegsol (2)</td>
<td>2 2 2 2</td>
<td>0 2 0 2 0 2</td>
<td>- - - -</td>
<td>7 23.5 64</td>
</tr>
<tr>
<td>pipesworld-nt (50)</td>
<td>17 45 30 23</td>
<td>46 46 25 26 15 15</td>
<td>0.31 0.94 0.88 0.66</td>
<td>5 5.6 4.3</td>
</tr>
<tr>
<td>pipesworld-t (50)</td>
<td>12 33 20 16</td>
<td>39 40 18 17 13 11</td>
<td>0.35 0.95 0.88 0.65</td>
<td>4 4.2 3.2</td>
</tr>
<tr>
<td>Sum (1517)</td>
<td>828 1088 828 705</td>
<td>1026 1005 690 694 522 528</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Local Explanations from MUGS

→ In response to user question “Why not property $\bigwedge_{g \in A} g$?”:

Proposition (Non-rhs-dominated Goal Exclusions from MUGS)

OSP task $\tau = (V, A, c, I, G^{hard}, G^{soft}, b)$, $\Pi$ its set of plans.
Modified task $\tau' := (V, A, c, I, G^{hard} \cup A, G^{soft} \setminus A, b)$.
Then: Non-rhs-dominated $\Pi \models \bigwedge_{g \in A} g \Rightarrow \neg \bigwedge_{g \in B} g \Leftrightarrow B$ MUGS in $\tau'$.

This is easier & smaller because: Less soft goals!

- Search tree size worst-case exponential in $|G^{soft}|$
- $\#MUGS$ worst-case exponential in $|G^{soft}|$
Experiments: Local Explanations

→ Performance and \( \# \text{MUGS} \) as function of question size \( |A| \) in “Why not property \( \bigwedge_{g \in A} g \)”: 

![Graph showing search time and coverage vs. question size](image1)

![Graph showing number of MUGS vs. question size](image2)
Discussion/Literature/Challenges

Many related computations: (for different purposes)

- Minimal unsatisfiable cores [e.g. Chinneck (2007); Laborie (2014)].
- Solvability borderline within a lattice of problem variants [de Kleer (1986); Reiter (1987)]
- MUGS = special case of preferred diagnoses [Grastien et al. (2011, 2012)], transfer pruning methods?
- Suggesting goals to drop in oversubscribed situations [Yu et al. (2017); Lauffer and Topcu (2019)].

Alternative algorithms to try:

- Run a single search in state space finding all maximal solvable goal subsets. Adapt pruning methods from oversubscription planning [Domshlak and Mirkis (2015)]?
- Represent the plan set $\Pi$ symbolically (e.g. BDD), use that representation to identify all entailment relations?
We Want More General Plan Properties!

\[
\begin{align*}
\text{drive}(R_1, L_1, L_2) \\
\text{takeImage}(I_1, R_1) \\
\text{drive}(R_2, L_4, L_5) \\
\text{takeImage}(I_3, R_2) \\
\text{drive}(R_2, L_5, L_6) \\
\text{takeImage}(I_2, R_2)
\end{align*}
\]
→ Compile more general plan properties into (additional variables/actions and) goal facts! For example:

- Precondition and goal formulas, conditional effects [Gazen and Knoblock (1997); Nebel (2000)]
- LTL formulas over plan trajectory [Edelkamp (2006); Baier et al. (2009)]
- Initial state uncertainty [Palacios and Geffner (2009)]

→ Here: effective-to-compile special case of LTL

### Action-Set Properties

OSP task $\tau = (V, A, c, I, G^{\text{hard}}, G^{\text{soft}}, b)$, $\Pi$ its set of plans, $A_1, \ldots, A_n \subseteq A$.

- **Action-set property**: propositional formula $\phi$ over atoms $A_1, \ldots, A_n$
- $\pi \models \phi$: if $\phi$ true given truth value assignment $A_i \cap \{a_1, \ldots, a_k\} \neq \emptyset$?
  - $A_i \mapsto \text{true} : A_i \mapsto \text{false}$ where $\pi = \langle a_1, \ldots, a_k \rangle$
Rovers Action-Set Properties

1. Specific rover
2. Use connection
3. Don’t use connection
4. Same rover
5. Energy limit > 50%
6. Order
Rovers Action-Set Properties: “Same Rover”

Action sets:

\[
A_1 = \{\text{takeImage}(I_1, R_1), \text{takeImage}(I_2, R_1)\}
\]

\[
A_2 = \{\text{takeImage}(I_1, R_2), \text{takeImage}(I_2, R_2)\}
\]

Test formula:

\[
A_1 \otimes A_2
\]
Action-Set Property Compilation

**Given:** $\tau$, $\Pi$, and $A_1, \ldots, A_n$

**Construct:** $\tau'$

- Booleans $isUsed_i$, initially $false$, set to $true$ by any action from $A_i$;
- formula-evaluation state variables and actions evaluating each $p_\phi$ based on these, setting Boolean flags $isTrue_\phi$;
- separate 1. planning phase vs. 2. formula-evaluation phase, switch action from 1. to 2. enabled when $G^{hard}$ is satisfied.

$\Rightarrow$ planning-phase prefixes in $\tau'$ one-to-one $\Pi$; given such prefix $\pi$, evaluation phase in $\tau'$ can achieve $isTrue_\phi$ iff $\pi \models \phi$. 
Rovers “Same Rover” Compilation: Illustration

\[ A_1 = \{ \text{takeImage}(I_1, R_1), \text{takeImage}(I_2, R_1) \} \]

\[ A_2 = \{ \text{takeImage}(I_1, R_2), \text{takeImage}(I_2, R_2) \} \]

\[ A_1 \otimes A_2 \]

\[ \begin{array}{c|c|c}
\text{used} & A_1 & A_2 \\
\hline
A_1 & \checkmark & \times \\
A_2 & \checkmark \\
\end{array} \]
MUGS

L₁ don’t use connection

⇒

specific rover

don’t use connection

specific rover

Hoffmann and Magazzeni Explainable AI Planning 15th Reasoning Web Summer School (RW 2019)
Transportation Example (NoMystery)

Variables $V$: $T_1, f_1, T_2, f_2, P_0, P_1, P_2$

Actions $A$: $\text{drive}(T_i, L_x, L_y)$, $\text{load}(T_i, P_j, L_x)$, $\text{unload}(T_i, P_j, L_x)$

Driving consumes fuel as indicated

Initial state $I$: as shown; $I(f_1) = 16, I(f_2) = 7$

Goal $G^{\text{soft}}$: at($P_0, L_4$), at($P_1, L_3$), at($P_2, L_5$)

Example action-set property analysis:

1. uses $T_0$ ($L_2, L_3$); 2. same truck $P_1, P_2$; 3. uses $T_0$ ($L_4, L_3$); 4. same truck $P_2, P_0$; 5. doesn’t use $T_0$ ($L_0, L_5$); 6. uses $T_1$ ($L_5, L_4$).

MUGS: 7, each of size 3, including $\{5, 2, 4\}$.

$\rightarrow$ User question “Why do you not avoid the road $L_0 - L_5$ (which has a lot of traffic at the moment)?” “Because if you don’t use that road, then you cannot deliver all packages with the same truck.”
Experiments: Global Explanations

→ Blocksworld, NoMystery, Rovers, TPP (top left to bottom right):
Conflict Based Learning: Is all over the place!

But: “State-space”

- Conflict-based learning is ubiquitous in constraint reasoning.
- Planning/reachability checking: Limited to bounded-length reachability (which is a form of constraint reasoning, easily encoded into e.g. SAT).
- Can we learn from conflicts in unbounded-length state space search?
Conflicts in State Space Search

What is a “conflict” in state space search?

→ Conflict = dead-end state from which the goal is unreachable.

- Planning: took bad decisions (ran out of resources, etc).
- Model checking safety properties: error can’t be reached from here.
Learning from Dead-End States

**Constraint reasoning:** For unsolvable partial assignment $\alpha$ that does not violate the constraints, add a new constraint discarding $\alpha$.

**Basic idea:** Constraints $\approx$ sound but incomplete dead-end detector $\Delta$.

$\rightarrow$ For unsolvable state $s$ not detected by $\Delta$, refine $\Delta$ to detect $s$.

**What are suitable $\Delta$?** E.g. $\Delta^C$, set $C$ of atomic conjunctions:

$$
\Delta^C(s, g) = \begin{cases} 
0 \\
\min_{a: \text{Regress}(g, a) \neq \bot} \Delta^C(s, \text{Regress}(g, a)) \\
\max_{g' \subseteq g, g' \in C} \Delta^C(s, g')
\end{cases}
$$

$\rightarrow \Delta^C(s, G) = \infty$: “Goal unreachable even when breaking up conjunctive subgoals into elements of $C$.” For suitable $C$, $\Delta^C$ detects all dead-ends.

**Conflict-Learning State Space Search:** [Steinmetz and Hoffmann (2016, 2017c)]

- Start with $C$ containing the singleton conjunctions.
- On dead-end $s$ where $\Delta^C(s, G) \neq \infty$, refine $\Delta^C$ by adding new atomic conjunctions, i.e., by extending $C$ such that $\Delta^C(s, G) = \infty$.
- Further, learn a clause $\phi$ where $s' \not\models \phi$ implies $\Delta^C(s', G) = \infty$. 
A Simple Transportation Example

Classical Planning Task:

- \( V = \{t, f, p_1, p_2\} \) with \( D_t = \{A, B, C\} \), \( D_f = \{0, 1, 2\} \), \( D_{p_i} = \{A, B, C, T\} \).
- \( A = \{load(p_i, x), unload(p_i, x), drive(x, x', n)\} \), where e. g.:
  - \( \text{pre}_{\text{drive}}(x, x', y) = \{(t, x), (f, n)\} \) and \( \text{eff}_{\text{drive}}(x, x', y) = \{(t, x'), (f, n - 1)\} \).
- \( I = \{(t, A), (f, 2), (p_1, B), (p_2, C)\} \). \( G = \{(p_1, C'), (p_2, B)\} \).

Conflict-Learning State Space Search

1. Forward state space search.
2. Identify dead-end states \( s \).
3. Refine \( C \) so that \( \Delta^C(s, G) = \infty \).
4. Learn a clause \( \phi \) s. t. \( s' \not\models \phi \) implies \( \Delta^C(s', G) = \infty \).
$\Delta^C$ in the Example

**State:** $tB, f1, p_1B, p_2C$

$C := C_1$ containing the singleton conjunctions:

1. $tB, f1, p_1B, p_2C$
2. $drive(B, A, 1) \rightarrow tA, load(p_1, B) \rightarrow p_1t$
3. $drive(A, C, 1) [pre : tB, f1] \rightarrow tC$
4. $unload(p_1, C) \rightarrow p_1C, load(p_2, C) \rightarrow p_2t$
5. $unload(p_2, B) \rightarrow p_2B$

$C := C_1 \cup \{tA \land f1\}$:

1. $tB, f1, p_1B, p_2C$
2. $drive(B, A, 1) \rightarrow tA [but \not\rightarrow tA \land f1], load(p_1, B) \rightarrow p_1t$
3. $\underline{drive(A, C, 1)} [pre : tA \land f1]$
Conflict-Learning State Space Search in the Example

3. Refine $C$ so that $\Delta^C(s, G) = \infty$.

4. Learn a clause $\phi$ s.t. $s' \not\models \phi$ implies $\Delta^C(s', G) = \infty$.

$\phi_1 = p_2 B \lor tB \lor f1 \lor f2$

$\phi_2 = p_2 B \lor f2 \lor tA$

$\Rightarrow$ Dead-End

$s_0 \rightarrow tA, f2, p_1 B, p_2 C$

$s_1 \rightarrow tB, f1, p_1 B, p_2 C$

$s_2 \rightarrow tC, f1, p_1 B, p_2 C$

$s_3 \rightarrow tB, f1, p_1 t, p_2 C$

$s_4 \rightarrow tA, f0, p_1 B, p_2 C$

$s_5 \rightarrow tA, f0, p_1 t, p_2 C$
Mature exploration of variants:

- Trap learning [Steinmetz and Hoffmann (2017b)]
- Refining more powerful LP-based dead-end detectors [Steinmetz and Hoffmann (2018)]
- Offline nogood computation [Steinmetz and Hoffmann (2017a)]

Open questions:

- Can we reason over the learned clauses, deducing new knowledge from the already derived one?
- Combine with property-directed reachability (PDR) [Bradley (2011); Suda (2014)]: Combine PDR clauses with clauses from different $\Delta$; use lower-bound heuristic functions for additional pruning in PDR; use heuristic functions for node selection in PDR.
- Apply these methods to model checking and game playing . . .
Thanks for your attention. Questions?
References I


References V
