KNOWLEDGE REPRESENTATION, REASONING AND DECLARATIVE PROBLEM SOLVING

By Jicheng Zhao
based on the notes of others

Class 15: specification in reasoning about actions and planning in AnsProlog*

Review: temporal projection with complete initial state

- The language A being used for discussion consists of a set of fluents F, and a set of actions A.
  - The domain description language D of the language A consists of effect propositions,
  - whereas the observation language consists of value propositions.
  - The Query language is used to query the descriptions, and contain value propositions.
  - Formulations of the language A in which the domain description language D is given by:

$$a \textbf{ causes } f \textbf{ if } p_1, p_2, \cdots p_m, \neg q_1, \neg q_2, \cdots, \neg q_n.$$
$$a \textbf{ causes } \neg g \textbf{ if } p_1, p_2, \cdots p_m, \neg q_1, \neg q_2, \cdots, \neg q_n.$$

– The observations are of the form

$$\textbf{initially } f.$$
$$\textbf{initially } \neg g.$$

– The query language would be

$$f \textbf{ after } a_1, a_2, \cdots a_n.$$
$$\neg g \textbf{ after } a_1, a_2, \cdots a_n.$$

The formulations in the previous lecture were done in AnsProlog, without the use of the classical negation. The closed world assumption was followed, in which negation was considered by default in case of failure to prove the fact.

Translate to AnsProlog

- The AnsProlog$^{\neg}$ formulations use the classical negation to encode the languages described above. The translation of the above sub languages in AnsProlog$^{\neg}$ would be as follows:
  - D:

$$\begin{aligned}
holds(f, res(a, S)) &\leftarrow holds(p_1, S), holds(p_2, S), \cdots holds(p_m, S),\\
&\neg holds(q_1, S), \neg holds(q_2, S), \cdots, \neg holds(q_n, S).\\
ab(f, a, S) &\leftarrow holds(p_1, S), holds(p_2, S), \cdots holds(p_m, S),\\
&\neg holds(q_1, S), \neg holds(q_2, S), \cdots, \neg holds(q_n, S).\\
\neg holds(g, res(a, S)) &\leftarrow holds(p_1, S), holds(p_2, S), \cdots holds(p_m, S),\\
&\neg holds(q_1, S), \neg holds(q_2, S), \cdots, \neg holds(q_n, S).\\
ab(f, a, S) &\leftarrow holds(p_1, S), holds(p_2, S), \cdots holds(p_m, S),\\
&\neg holds(q_1, S), \neg holds(q_2, S), \cdots, \neg holds(q_n, S).
\end{aligned}$$

– O:

$$holds(f, S_0) \leftarrow .$$
$$\neg holds(g, S_0) \leftarrow .$$

– The rules for inertia:

$$holds(F, res(A, S)) \leftarrow holds(F, S), \textbf{not } ab(F, A, S).$$
$$\neg holds(F, res(A, S)) \leftarrow \neg holds(F, S), \textbf{not } ab(F, A, S).$$

Note:

1. The $ab(f, a, S)$ and the $ab(g, a, S)$ rules have the same head 'ab' rather than '$ab1$' and '$ab2$' or '$ab()$' and '$\neg ab()$' because they are rules to check for the inertia of the fluents, and would mean the same if used in either of the forms. The rules are to block the inference of the fluent involved by inertia.
2. Also, we use ¬holds (f) instead of holds (¬f) because we want the negation to be explicitly specified in the rule, which would make reasoning easier.

3. We may delete the rules about ab and replace the rules about inertia with:

$$holds(F, res(A, S)) \leftarrow holds(F, S), \textbf{not } \neg holds(F, res(A, S)).$$
$$\neg holds(F, res(A, S)) \leftarrow \neg holds(F, S), holds(F, res(A, S)).$$

Example 1 *The Yale Shooting example*

– *D:*

$$load\ \boldsymbol{causes}\ loaded.$$
$$shoot\ \boldsymbol{causes}\ \neg alive\ \boldsymbol{if}\ loaded.$$

– *O:*

$$\boldsymbol{initially}\ \neg loaded.$$
$$\boldsymbol{initially}\ alive.$$

The Translation contains the following parts:

– D:

$$holds(loaded, res(load, S)) \leftarrow .$$
$$\neg holds(alive, res(shoot, S)) \leftarrow holds(loaded, S).$$
$$ab(alive, shoot, S) \leftarrow holds(loaded, S).$$
$$ab(loaded, load, S) \leftarrow .$$

– O:

$$holds(alive, S_0) \leftarrow .$$
$$\neg holds(loaded, S_0) \leftarrow .$$

– The general rules:

$$holds(F, res(A, S)) \leftarrow holds(F, S), \textbf{not } ab(F, A, S).$$
$$\neg holds(F, res(A, S)) \leftarrow \neg holds(F, S), \textbf{not } ab(F, A, S).$$

- The above example was in the case where the set of observations were complete, i.e., when the observation O had predicates to explain each of the fluents. In case of incomplete observations, the reasoning processes with the above rules do not suffice.

For example, if O did not have any information about loaded, with the above rules, we will be able to infer

$$holds(alive, res(shoot, S_0))$$

This is because $\neg holds(alive, res(shoot, S_0))$ does not fire since of the body of the rule is false. This conclusion is not true.

So, in case of incomplete information, the rules have to be modified in such a way that the states are considered to be abnormal even in the 'possibility' of the action firing.

Hence, the rules for inertia are modified as

$$ab(alive, shoot, S) \leftarrow \textbf{not } \neg holds(loaded, S).$$

In the general case, each rule

$$\begin{aligned} ab(f, a, S) \leftarrow\ & holds(p_1, S), holds(p_2, S), \cdots holds(p_m, S), \\ & \neg holds(q_1, S), \neg holds(q_2, S), \cdots, \neg holds(q_n, S). \end{aligned}$$

is replaced by

$$ab(f, a, S) \leftarrow \textbf{not } \neg holds(p_1, S), \textbf{not } \neg holds(p_2, S), \cdots \textbf{not } \neg holds(p_m, S), \\ \textbf{not } holds(q_1, S), \textbf{not } holds(q_2, S), \cdots, \textbf{not } holds(q_n, S).$$

The advantage of this approach is that it can handle some kinds of incompleteness. Though the above rules would entail a correct reasoning, it is not complete. To illustrate, consider the example

Example 2

$$a \textbf{ causes } f \textbf{ if } p.$$
$$a \textbf{ causes } f \textbf{ if } \neg p.$$
$$\textbf{initially } \neg f.$$

By intuition, this should translate that f is true. But the reasoning with the rules would arrive at the conclusion that f is false. The translation of the example in AnsProlog would be,

$$holds(f, res(a, S)) \leftarrow holds(p, S), \textbf{not } ab(f, a, S).$$

$$
\begin{aligned}
holds(f, res(a, S)) &\leftarrow \quad \neg holds(p, S), \textbf{not } ab(f, a, S). \\
ab(f, a, S) &\leftarrow \textbf{not } \neg holds(p, S). \\
ab(f, a, S) &\leftarrow \textbf{not } holds(p, S). \\
\neg holds(f, S_0) &\quad \leftarrow . \\
holds(F, res(A, S)) &\leftarrow holds(F, S), \textbf{not } ab(F, A, S). \\
\neg holds(F, res(A, S)) &\leftarrow \quad \neg holds(F, S), \textbf{not } ab(F, A, S).
\end{aligned}
$$

Since we do not have any information about p, we would be able to infer both the cases of $ab(f, a, S)$. Hence, our reasoning would be $\neg holds(f, res(a, S))$.

- This is the case where precision is high , and the recall is low. To increase the recall so our reasoning becomes complete and correct would increase the complexity of the planner.

  Non-deterministic effect of actions will lead to the notion of multiple models for completeness.