KNOWLEDGE REPRESENTATION, REASONING AND DECLARATIVE PROBLEM SOLVING

By Jicheng Zhao
based on the notes of others

Class 14: Dealing with incomplete initial states

Specification

- The domain D has rules of the form

$$a \textbf{ causes } f \textbf{ if } p_1, p_2, \ldots p_n, \neg q_1, \neg q_2, \ldots \neg q_m.$$

- The observations O have the form

$$\textbf{initially } f$$

- The query Q is of the form

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

$$(D, O) \models Q$$

iff

$$\Pi_1(D, O) \models_{smodels} tr(Q).$$

$tr(Q)$ is some transformation of the query Q. We now need to define such a transformation.

- We start with the transition function $\phi : state \times action \rightarrow state$.

  Note that P is the set of fluents added by positive effect propositions, while N is the set of fluents subtracted by negative effect propositions.

  $\phi(\sigma, a) = (\sigma \cup P) - N$ if $P \cap N = \emptyset$, otherwise undefined.

$$\begin{aligned} P &= \{f : \text{``}a \textbf{ causes } f \textbf{ if } p_1, p_2, \dots p_n, \neg q_1, \neg q_2, \dots, \neg q''_m \in D, \\ &\quad \{p_1, p_2, \dots, p_n\} \subseteq \sigma, \{q_1, q_2, \dots, q_m\} \cap \sigma = \emptyset\} \\ N &= \{f : \text{``}a \textbf{ causes } \neg f \textbf{ if } p_1, p_2, \dots p_n, \neg q_1, \neg q_2, \dots \neg q''_m \in D, \\ &\quad \{p_1, p_2, \dots p_n\} \subseteq \sigma, \{q_1, q_2, \dots q_m \cap \sigma = \emptyset\} \end{aligned}$$

- σ_0 is initial state for (D, O) if:
  1. for each "**initially** f" in O, $f \in \sigma_0$ and
  2. for each "**initially** $\neg f$" in O, $f \notin \sigma_0$.
- $(D, O) \models f$ **after** $a_1 \cdots a_n$ iff for all initially states σ_0 of (D, O), f is true in $\phi(\cdots \phi(\phi(\sigma_0, a_1), a_2) \cdots, a_n)$.

Example

Assume the observation set O is complete. We make the closed world assumption. We want to convert the following to AnsProlog.

Example 1

$$\begin{aligned} load\ \textbf{\textit{causes}}\ loaded.\\ shoot\ \textbf{\textit{causes}}\ \neg alive\ \textbf{\textit{if}}\ loaded.\\ \textbf{\textit{Initially}}\ alive.\\ \textbf{\textit{Initially}}\ \neg loaded.\end{aligned}$$

- The initial condition is converted to

$$holds(alive, s_0).$$

  s_0 is the initial situation.

  $\neg alive$ is proved by the closed world assumption, because $alive$ can not be proved in S after shoot.

- $res(A, S)$ is the situation that results from executing action A in situation S.

  The two actions can be presented as

$$\begin{aligned} holds(loaded, res(load, S)) &\leftarrow . \\ holds(F, res(A, S)) &\leftarrow holds(F, S), \textbf{not } ab(F, A, S). \\ ab(alive, shoot, S) &\leftarrow holds(loaded, S). \end{aligned}$$

- Suppose we want determine whether the query "$\neg alive$ **after** $load, shoot$" is entailed. Then we would use the following AnsProlog query.

$$holds(alive, res(shoot, res(load, s_0)))$$