Action Language $\mathcal{L}$ and Diagnosis

Tran Cao Son

Department of Computer Science
New Mexico State University
Las Cruces, NM 88011, USA

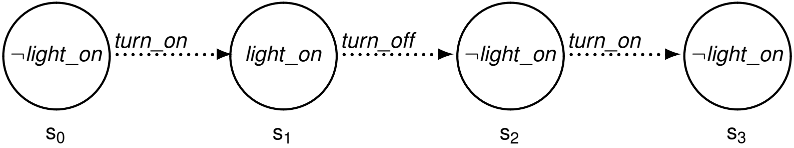
2007

Example $N_0 = (D_0, \Gamma_0)$

$$D_0 = \left\{ \begin{array}{l} turn\_on \textbf{ causes } light\_on \textbf{ if } \neg ab(bulb) \\ turn\_off \textbf{ causes } \neg light\_on \\ \neg light\_on \textbf{ if } ab(bulb) \\ burn\_out(bulb) \textbf{ causes } ab(bulb) \\ \textbf{impossible } burn\_out(bulb) \textbf{ if } ab(bulb) \end{array} \right.$$

$$\Gamma_0 = \left\{ \begin{array}{l} turn\_on \textbf{ occurs\_at } s_0 \\ turn\_off \textbf{ occurs\_at } s_1 \\ turn\_on \textbf{ occurs\_at } s_2 \\ s_1 \textbf{ precedes } s_2 \\ s_2 \textbf{ precedes } s_3 \\ \neg light\_on \textbf{ at } s_0 \\ light\_on \textbf{ at } s_1 \\ \neg light\_on \textbf{ at } s_2 \\ \neg light\_on \textbf{ at } s_3 \end{array} \right.$$

Illustration of N_0



Causal Interpretation

- A *causal interpretation* of (D, Γ) is a partial function from action sequences to interpretations of *Lang*(**F**), whose domain is nonempty and prefix-closed.
- A set X of action sequences is prefix-closed if for every sequence $\alpha \in X$, every prefix of α is also in X.
- By $Dom(\Psi)$ we denote the domain of a causal interpretation Ψ.
- $[] \in Dom(\Psi)$ for every causal interpretation Ψ, where $[]$ is the empty sequence of actions.

Causal Model

A *causal model* of D is a causal interpretation Ψ such that:

(i) $\Psi([])$ is a state of D; and

(ii) for every $\alpha \circ a \in Dom(\Psi)$, $\Psi(\alpha \circ a) \in \Phi(a, \Psi(\alpha))$.

A *situation assignment* of **S** with respect to D is a mapping Σ from **S** into the set of action sequences of D that satisfy the following properties:

(i) $\Sigma(s_0) = []$;

(ii) for every $s \in \mathbf{S}$, $\Sigma(s)$ is a prefix of $\Sigma(s_c)$.

Example — Models of $N_0 = (D_0, \Gamma_0)$

There are four interpretations for the set of fluent formulas of D_0:
$I_0 = \emptyset$,
$I_1 = \{light\_on\}$,
$I_2 = \{ab(bulb)\}$, and
$I_3 = \{ab(bulb), light\_on\}$.

Since $Cn(I_3)$ is not closed under the static causal law $\neg light\_on$ **if** $ab(bulb)$ of D_0 ($ab(bulb) \in Cn(I_3)$ and this law implies that $light\_on \notin Cn(I_3)$), I_3 is not a state of D_0. Thus, D_0 has only three states $s_0 = \emptyset$, $s_1 = \{light\_on\}$, and $s_2 = \{ab(bulb)\}$.

Transition Function

$s_0 = \{\neg ab(bulb), \neg light\_on\}$, $s_1 = \{\neg ab(bulb), light\_on\}$, and
$s_2 = \{ab(bulb), \neg light\_on\}$.
The transition function of D_0 is given by

$$\begin{array}{ll}
\Phi(turn_on, s_0) = \{s_1\} & \Phi(turn_on, s_1) = \{s_1\} \\
\Phi(turn_on, s_2) = \{s_2\} & \Phi(turn_off, s_0) = \{s_0\} \\
\Phi(turn_off, s_1) = \{s_0\} & \Phi(turn_off, s_2) = \{s_2\} \\
\Phi(burn_out(bulb), s_0) = \{s_2\} & \Phi(burn_out(bulb), s_1) = \{s_2\} \\
\Phi(burn_out(bulb), s_2) = \emptyset &
\end{array}$$

Φ can be represented graphically as follows.

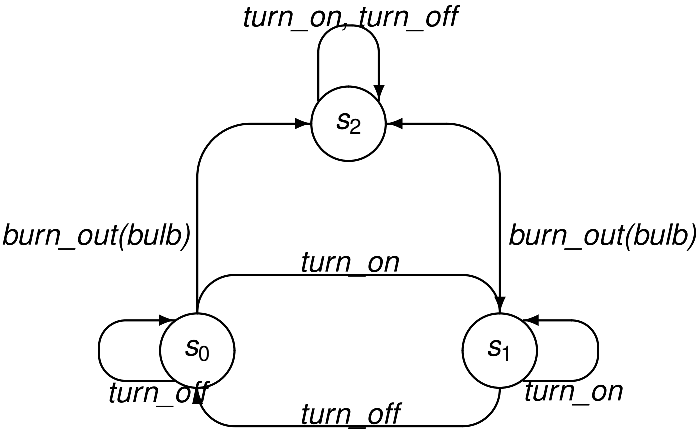


Figure: Transition graph of narrative N_0

Three Models of N_0

$M_1 = (\Psi_1, \Sigma_1)$, $M_2 = (\Psi_2, \Sigma_2)$, and $M_3 = (\Psi_3, \Sigma_3)$, where
$\Psi_1([]) = \Psi_2([]) = \Psi_3([]) = s_0$, and

$\Sigma_1(s_0) = []$,
$\Sigma_1(s_1) = turn\_on$,
$\Sigma_1(s_2) = turn\_on \circ turn\_off$, and
$\Sigma_1(s_3) = \Sigma_1(s_c) = turn\_on \circ turn\_off \circ turn\_on \circ turn\_off$,

$\Sigma_2(s_0) = []$,
$\Sigma_2(s_1) = turn\_on$,
$\Sigma_2(s_2) = turn\_on \circ turn\_off \circ burn\_out(bulb)$, and
$\Sigma_2(s_3) = \Sigma_2(s_c) = turn\_on \circ turn\_off \circ burn\_out(bulb) \circ turn\_on$.

$\Sigma_3(s_0) = []$,
$\Sigma_3(s_1) = turn\_on$,
$\Sigma_3(s_2) = turn\_on \circ turn\_off$, and
$\Sigma_3(s_3) = \Sigma_3(s_c) = turn\_on \circ turn\_off \circ turn\_on \circ burn\_out(bulb)$.

Observation

The difference between M_1 and M_2: In M_1, the unobserved action (*turn_off*) occurs after the last observed action whereas in M_2 the unobserved action (*burn_out*(*bulb*)) occurs prior to it.

$\Sigma_1(s_3) = \Sigma_1(s_c) = turn\_on \circ turn\_off \circ turn\_on \circ turn\_off$,
$\Sigma_2(s_3) = \Sigma_2(s_c) = turn\_on \circ turn\_off \circ burn\_out(bulb) \circ turn\_on$.
$\Sigma_3(s_3) = \Sigma_3(s_c) = turn\_on \circ turn\_off \circ turn\_on \circ burn\_out(bulb)$.

Diagnostic Reasoning — Example

$Sys_3 = (SD_3 \cup SD_{ab}, \{bulb, switch\}, OBS_3)$ where

$$SD_3 = \left\{ \begin{array}{ll}
(r1) & turn_on \textbf{ causes } light_on \\
 & \quad \textbf{if } \neg ab(bulb), \neg ab(switch), \\
 & \quad connected(bulb, switch) \\
(r2) & turn_off \textbf{ causes } \neg light_on \\
 & \quad \textbf{if } connected(bulb, switch) \\
(r3) & disconnect(bulb, switch) \textbf{ causes} \\
 & \quad \neg connected(bulb, switch) \\
 & \quad \textbf{if } connected(bulb, switch) \\
(r4) & connect(bulb, test_device) \textbf{ causes} \\
 & \quad connected(bulb, test_device) \\
 & \quad \textbf{if } \neg connected(bulb, switch) \\
(r5) & light_on \textbf{ if } connected(bulb, test_device), \neg ab(bulb)
\end{array} \right.$$

$$OBS_3 = \left\{ \begin{array}{l} \textit{turn\_on} \ \textbf{occurs\_at} \ s_0 \\ \textit{turn\_off} \ \textbf{occurs\_at} \ s_1 \\ \textit{turn\_on} \ \textbf{between} \ s_2, s_3 \\ s_0 \ \textbf{precedes} \ s_1 \\ s_1 \ \textbf{precedes} \ s_2 \\ s_2 \ \textbf{precedes} \ s_3 \\ \neg \textit{light\_on} \ \textbf{at} \ s_0 \\ \textit{connected}(\textit{bulb}, \textit{switch}) \ \textbf{at} \ s_0 \\ \textit{light\_on} \ \textbf{at} \ s_1 \\ \neg \textit{connected}(\textit{bulb}, \textit{test\_device}) \ \textbf{at} \ s_0 \\ \neg \textit{light\_on} \ \textbf{at} \ s_2 \\ \neg \textit{light\_on} \ \textbf{at} \ s_3 \end{array} \right.$$

Diagnostic Reasoning — Example

and

$$SD_{ab} = \left\{ \begin{array}{l} burn\_out(bulb) \textbf{ causes } ab(bulb) \\ break(switch) \textbf{ causes } ab(switch) \\ miracle(bulb) \textbf{ causes } ab(bulb) \\ miracle(switch) \textbf{ causes } ab(switch) \end{array} \right\}$$

Here, the only observable fluent is *light_on*.

Diagnoses

It is easy to see that there are two current fluent diagnoses for Sys_3: $\Delta_1 = \{bulb\}$ and $\Delta_2 = \{switch\}$ which correspond to the models (Ψ, Σ_1) and (Ψ, Σ_2) of $(SD_3 \cup SD_{ab}, OBS_3 \cup OK_0)$ where $OK_0 = \{\neg ab(c) \ \textbf{at} \ s_0 \mid c \in COMPS\}$

$\Psi([]) = \{connected(bulb, switch), \neg light\_on\} \cup \{\neg ab(c) \mid c \in COMPS\}$, and

$\Sigma_1(s_0) = []$,
$\Sigma_1(s_1) = turn\_on$,
$\Sigma_1(s_2) = turn\_on \circ turn\_off \circ burn\_out(bulb)$,
$\Sigma_1(s_3) = \Sigma_1(s_c) = turn\_on \circ turn\_off \circ burn\_out(bulb) \circ turn\_on$, and

$\Sigma_2(s_0) = []$,
$\Sigma_2(s_1) = turn\_on$,
$\Sigma_2(s_2) = turn\_on \circ turn\_off \circ break(switch)$,
$\Sigma_2(s_3) = \Sigma_2(s_c) = turn\_on \circ turn\_off \circ break(switch) \circ turn\_on$.

Diangostic Plan

$Sys = (SD \cup SD_{ab}, COMPS, OBS \cup OK_0)$ and P is a conditional plan consisting of only actions that the agent can execute.
s_C — current state of the world
P is a diagnostic plan for Sys iff for every $c \in COMPS$

$$(SD, OBS) \models \textbf{ knows } ab(c) \textbf{ after } P \textbf{ at } s_C$$

and

$$(SD, OBS) \models ab(c) \textbf{ during } P \textbf{ at } s_C$$

This says that after the execution of P in the current state, we know (the agent knows) the truth value of $ab(c)$, i.e., whether the component c is defected or not. Furthermore, we require that the value of $ab(c)$ does not change during the execution of P.

Example of Diagnostic Plan

Sys_3: the empty plan is not a diagnostic plan.
$s_1 = \{connected(bulb, switch), ab(bulb), \neg ab(switch), \neg light\_on\}$
$s_2 = \{connected(bulb, switch), \neg ab(bulb), ab(switch), \neg light\_on\}$.
Let $\mathcal{S} = \{s_1, s_2\}$.
$\hat{\Phi}([], \langle s_1, \mathcal{S} \rangle) = \{\langle s_1, \mathcal{S} \rangle, \langle s_2, \mathcal{S} \rangle\}$ and

$$\hat{\Phi}([], \langle s_1, \mathcal{S} \rangle) \not\models \textbf{knows } ab(bulb)$$

and

$$\hat{\Phi}([], \langle s_1, \mathcal{S} \rangle) \not\models \textbf{knows } ab(switch)$$

We conclude that the empty plan is not a purely diagnostic plan for Sys_3.

Example of Diagnostic Plan

Sys_3:
$P = turn\_off \circ disconnect(bulb, switch) \circ connect(bulb, test\_device)$ is a diagnostic plan.
$\hat{\Phi}(P, \langle s_1, \mathcal{S}\rangle) = \langle s_1', \{s_1'\}\rangle$ and $\hat{\Phi}(P, \langle s_2, \mathcal{S}\rangle) = \langle s_2', \{s_2'\}\rangle$ where
$s_1' = \{connected(bulb, test\_device), ab(bulb), \neg ab(switch), \neg light\_on\}$
and
$s_2' = \{connected(bulb, test\_device), ab(switch), light\_on, \neg ab(bulb)\}$.

In both cases, the values of $ab(bulb)$ and $ab(switch)$ do not change while P is executed.

$\hat{\Phi}(P, \langle s_1, \mathcal{S}\rangle) \models$ **knows** $ab(bulb)$ $\hat{\Phi}(P, \langle s_1, \mathcal{S}\rangle) \models$ **knows** $\neg ab(switch)$

and

$\hat{\Phi}(P, \langle s_2, \mathcal{S}\rangle) \models$ **knows** $\neg ab(bulb)$ $\hat{\Phi}(P, \langle s_2, \mathcal{S}\rangle) \models$ **knows** $ab(switch)$

So, P is a diagnostic plan of Sys_3.

Complex Action Theories

Consider the action of driving a car for 10 units of time with a velocity v. We represent this action *of fixed duration* by $drive_{0,10}(v)$. Now, let *gas_in_tank* denote the amount of gasoline available in the tank.

The executability condition saying that the action $drive_{0,10}(v)$ is executable if $gas\_in\_tank \geq 20$ can be expressed in $\mathcal{ADC}$ as:

executable $drive_{0,10}(v)$ **if** $gas\_in\_tank \geq 20$.

The effect of the action $drive_{0,10}(v)$ on the fluent *loc* (encoding how far from the initial position the car is) is expressed as:

$drive_{0,10}(v)$ **causes** $locn = locn + v * t$ **from** $0 \rightarrow 10$.

We can record the information that the action $drive_{0,10}(v)$ is under execution by the following propositions:

$drive_{0,10}(v)$ **causes** *driving* **from** $0 \rightarrow 10$.
$drive_{0,10}(v)$ **causes** $\neg driving$ **from** $10 \rightarrow 10$.
where *driving* is a Boolean-fluent.

We can express that driving at a velocity v consumes $c(v)$ unit of gasoline per unit time as