Knowledge, action, and the frame problem

Reasoning about Actions

Francesco Petri, Charlotte Primiceri and Flavio Maiorana

May 29, 2024

SAPIENZA
UNIVERSITÀ DI ROMA

Table of Contents

- Scherl and Levesque's Approach
  - Introduction
  - Knowledge as a fluent
  - Defining a successor state axiom for K
- Theorems and Regression
  - Theorems
  - Regression
- Example: The Gardening Robot
  - Axiomatization
  - Instantiation

Outline

Situation calculus provides a framework for reasoning about actions.
This work presents an expansion to handle the *knowledge* possessed or acquired by the agent, and allow it to shape the agent's decisions.

- Knowledge is represented by one additional fluent
- Uniform axiomatization with the rest of sitcalc
- Ordinary actions and knowledge-producing ones are strictly separated
- Easy expansion of regression as defined in [Reiter2001]
- Desirable properties are direct consequences of the axiomatization (e.g. knowledge persistence / memory)

The K fluent

$$\mathrm{K}(s', s)$$

Defines an accessibility relation between situations.

(Informal) definition

$\mathrm{K}(s', s)$ is true if an agent in situation s could mistake the current situation for the other s', given its current knowledge.

Knowledge

Definition of knowledge

A fluent is known to be true (false) in a situation s if and only if it is true (false) in all situations accessible from s.

Shorthand notation: $\textbf{Knows}(\phi, s) \stackrel{\text{def}}{=} \forall s'\, \mathrm{K}(s', s) \rightarrow \phi(s')$

Knowledge-producing actions

Actions that have an effect on the agent's knowledge

SENSE actions

Learn the truth value of a formula. Example: check if a door is open or closed.

$$\textbf{Knows}(\mathrm{P}, \textsc{do}(\textsc{sense}_{\mathrm{P}}, s)) \vee \textbf{Knows}(\neg \mathrm{P}, \textsc{do}(\textsc{sense}_{\mathrm{P}}, s))$$

READ actions

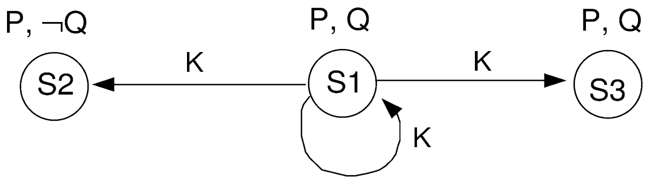
Learn the value of a term. Example: read a number on a sheet of paper.

$$\exists x\, \textbf{Knows}(\tau = x, \textsc{do}(\textsc{read}_{\tau}, s))$$

Knowledge effects

In order to complete the specification of the K fluent, we need to define its successor state axiom, determining how ordinary actions and knowledge-producing actions affect it.

Consider this case study with three accessible situations. The agent is in S1.



$$\textbf{Knows}(\mathrm{P}, S1) \land \neg\textbf{Knows}(\mathrm{Q}, S1)$$

Ordinary actions

Assume the agent performs a DROP action.

Informal idea

The agent cannot distinguish the current situation s from all the other s' accessible from it. Therefore, after executing the action, the agent may believe to be in any situation resulting from any s' after executing DROP.

Axiomatization

$$\mathrm{K}(s'', \textsc{do}(\textsc{drop}, s)) \equiv \exists s' \, (\textsc{Poss}(\textsc{drop}, s') \wedge \mathrm{K}(s', s) \wedge s'' = \textsc{do}(\textsc{drop}, s'))$$

Ordinary actions

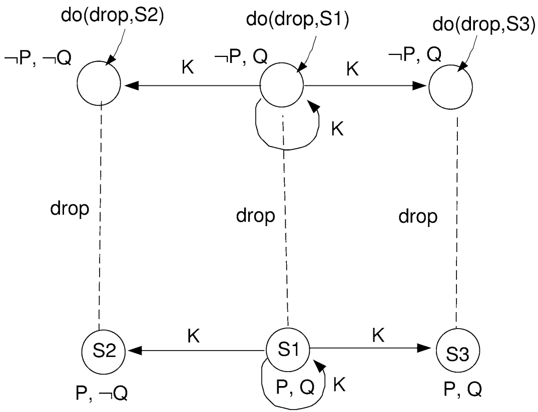
The **only knowledge gained is that the** $\textsc{drop}$ **action has been performed**, as well as anything that can be derived from the action effects.
For example, if $\textsc{drop}$ makes P false:

$$\mathrm{P}(\textsc{do}(a, s)) \equiv a \neq \textsc{drop} \wedge \mathrm{P}(s)$$

then

$$\textbf{Knows}(\neg \mathrm{P}, \textsc{do}(\textsc{drop}, S1))$$

but no extra knowledge is gained about Q.



Knowledge-producing actions

Consider an action SENSE_{Q} that provides information on whether Q is true or false. We define a **sensing result function** to represent the signal received by the agent in response:

Sensing result function

$$\text{SR}(\text{SENSE}_{\text{Q}}, s) = r \equiv (r = \text{``YES''} \land \text{Q}(s)) \lor (r = \text{``NO''} \land \neg\text{Q}(s))$$

Knowledge-producing actions

When the agent executes $\textsc{sense}_{\mathrm{Q}}$, what are the accessible situations afterwards?

Informal definition

Initially, the agent cannot distinguish the current situation s from all the other s' accessible from it. Therefore, after executing $\textsc{sense}_{\mathrm{Q}}$, the agent may believe to be in any situation that:

- results from any s' after executing the action,
- **AND** would yield the same sensing result as the one that has been observed.

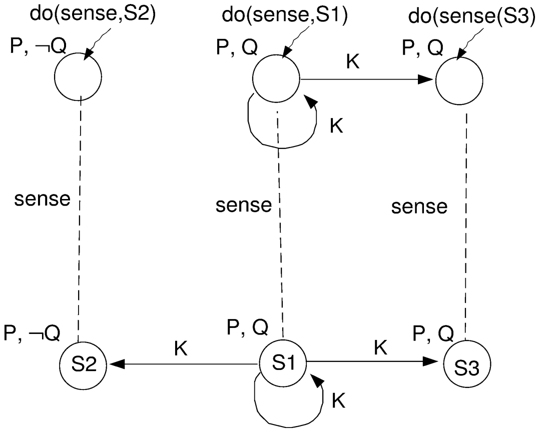
Axiomatization

$$\mathrm{K}(s'', \textsc{do}(\textsc{sense}_{\mathrm{Q}}, s)) \equiv \exists s' \, (\textsc{Poss}(\textsc{sense}_{\mathrm{Q}}, s') \wedge \mathrm{K}(s', s) \wedge \\ s'' = \textsc{do}(\textsc{sense}_{\mathrm{Q}}, s') \wedge \textsc{sr}(\textsc{sense}_{\mathrm{Q}}, s) = \textsc{sr}(\textsc{sense}_{\mathrm{Q}}, s'))$$

Knowledge-producing actions

After executing $\textsc{sense}_Q$, **only situations with the same truth value for Q are accessible**.
Thus, in addition to knowing that $\textsc{sense}_Q$ has been performed, the agent now knows the truth value of Q as well, by definition.

$$\textbf{Knows}(Q, \textsc{do}(\textsc{sense}_Q, S1))$$



Sensing results in general

The concept of sensing result extends to all types of action, allowing for a uniform axiomatization.

Ordinary actions

$$\mathrm{SR}(\mathrm{DROP}, s) = r \equiv r = \text{“OK”}$$

SENSE-type knowledge-producing actions

$$\mathrm{SR}(\mathrm{SENSE}_{\mathrm{Q}}, s) = r \equiv (r = \text{“YES”} \wedge \mathrm{Q}(s)) \vee (r = \text{“NO”} \wedge \neg \mathrm{Q}(s))$$

READ-type knowledge-producing actions

$$\mathrm{SR}(\mathrm{SENSE}_{\tau}, s) = r \equiv r = \tau(s)$$

The successor state axiom for K

Putting it all together, the definitive form is as follows:

Successor state axiom for the K fluent

$$\mathrm{K}(s'', \textsc{do}(a, s)) \equiv \exists s' \; (\textsc{Poss}(a, s') \wedge \mathrm{K}(s', s) \wedge$$
$$s'' = \textsc{do}(a, s') \wedge \textsc{sr}(a, s) = \textsc{sr}(a, s'))$$

What about…

- **…mixing ordinary and knowledge effects?**
  We assume that ordinary and knowledge actions are disjoint: each action is going to be axiomatized as either affecting *only* the K fluent or as not affecting it at all. This does not cause loss of generality.
- **…knowledge of arbitrary formulae?**
  They already work within this system.
  Example: **Knows**$(\forall x(\textsc{man}(x) \rightarrow \textsc{mortal}(x)) \wedge \textsc{man}(Socrates))$

Table of Contents

- Scherl and Levesque's Approach
  - Introduction
  - Knowledge as a fluent
  - Defining a successor state axiom for K
- Theorems and Regression
  - Theorems
  - Regression
- Example: The Gardening Robot
  - Axiomatization
  - Instantiation

Theorems

Theorem 1 (Knowledge-producing effects). *For all situations s, all fluents* P *(other than* K*) and knowledge-producing action terms α, if* $\mathrm{P}(s)$ *then* $\mathrm{P}(\textsc{do}(\alpha, s))$.

Theorem 2 (Default persistence of ignorance). *For an action α and a situation s, if* $\neg\mathbf{Knows}(\mathrm{P}, s)$ *holds and the axiomatization entails*

$$\forall s\ \mathrm{P}(s) \equiv \mathrm{P}\big(\textsc{do}(\alpha, s)\big)$$

and

$$\forall y\ \neg\mathbf{Knows}\big(\big(\textsc{Poss}(\alpha) \wedge \textsc{sr}(\alpha) = y\big) \rightarrow \mathrm{P}, s\big)$$

then

$$\neg\mathbf{Knows}\big(\mathrm{P}, \textsc{do}(\alpha, s)\big)$$

holds as well.

Theorems

Theorem 3 (Knowledge incorporation). *For a knowledge-producing action* α, *a fluent or the negation of a fluent* F, *a fluent or the negation of a fluent* P, *and a situation* s, *if the axiomatization entails*

$$\exists y\ \mathbf{Knows}\big(\mathrm{F} \equiv \textsc{sr}(\alpha) = y, s\big)$$

and also

$$\mathrm{F}(s), \quad \textsc{Poss}(\alpha, s),$$

and

$$\mathbf{Knows}(\mathrm{F} \rightarrow \mathrm{P}, s)$$

hold, then

$$\mathbf{Knows}\big(\mathrm{P}, \textsc{do}(\alpha, s)\big)$$

holds as well.

Theorems

Theorem 4 (Memory). *For all fluents* P *and situations* s, *if* **Knows**(P, s) *holds then* **Knows**(P, $\text{DO}(\alpha, s)$) *holds as long as the axiomatization entails*

$$\forall s\ \mathrm{P}(s) \equiv \mathrm{P}(\text{DO}(\alpha, s))$$

Theorem 5 (Knowledge of effects of actions). *If* α *is an ordinary (not a knowledge-producing) action, and if the axiomatization entails*

$$\forall s\ \phi[s] \rightarrow \mathrm{P}(\text{DO}(\alpha, s))$$

where ϕ *is an arbitrary formula with situation terms suppressed*[8] *and* P *is a fluent or its negation, then the following is also entailed*:

$$\textbf{Knows}((\text{POSS}(\alpha) \wedge \phi), s) \rightarrow \textbf{Knows}(\mathrm{P}, \text{DO}(\alpha, s))$$

Reasoning Tasks

Projection Task

Determing if a sentence G is true in the situation resulting from the execution of an action sequence is represented by the query:

$$F \models G(\mathrm{DO}([\alpha_1, \ldots, \alpha_n], S_0))$$

Reasoning Tasks

Projection Task

Determing if a sentence G is true in the situation resulting from the execution of an action sequence is represented by the query:

$$F \models G(\mathsf{DO}([\alpha_1, \ldots, \alpha_n], S_0))$$

Legality Task

The situation term

$$\mathsf{DO}(\alpha_m, \mathsf{DO}(\alpha_{m-1} \ldots \mathsf{DO}(\alpha_1, S_0) \ldots))$$

is a legal situation iff

$$[\alpha_1, \ldots, \alpha_m]$$

is a legal action sequence

Regression

- Reduce Reasoning about future situations to reasoning about the initial situation;
- Sound and Complete Reasoner.

Regression

Definition of Regressor Operator through:

- Ordinary Actions
- Knowledge-Producing Actions

Regression Theorem

Theorem

For any ground situation term

$$F \models G(s_{gr}) \quad iff \quad F - F_{SS} \models R^*_\theta[G(s_{gr})]$$

Proof.

It suffices to show the preservation of logical equivalence

$$F \models G(s_{gr}) \;\equiv\; \mathsf{R}^*_\theta[G(s_{gr})]$$

□

Regression Theorem

Theorem (Regression-Ordinary Actions)

$$R^*_\theta[\textbf{\textit{Knows}}(W, DO(a, s))] \equiv$$

$$\textbf{\textit{Knows}}(POSS(a) \to R^*_\theta[W[DO(a, s')]], s)$$

Regression-Ordinary Actions

Proof.

1. **Knows**(W, DO(a, s))
2. by the definition of **Knows**:

$$\forall s'' \ \mathsf{K}(s'', \mathsf{DO}(a, s)) \ \equiv \rightarrow \mathsf{W}[s'']$$

□

Regression-Ordinary Actions

Proof.

3. by Successor State Axiom:

$$\mathsf{K}(s'', \mathsf{DO}(a, s)) \equiv \quad \exists s'(\mathsf{K}(s', s) \wedge \mathsf{POSS}(a, s) \wedge s'' = \mathsf{DO}(a, s'))$$

rephrase as:

$$\forall s''(\exists s'(\mathsf{K}(s', s) \wedge \mathsf{POSS}(a, s) \wedge s'' = \mathsf{DO}(a, s')) \rightarrow \mathsf{W}[s''])$$

4. the axiomatization entails:

$$\forall s, s' \mathsf{SR}(a, s) = \mathsf{SR}(a, s')$$

so:

Regression-Ordinary Actions

Proof.

5. Inductive Hypothesis:

$$\forall s'(\mathsf{K}(s', s) \land (\mathsf{POSS}(a, s'))) \rightarrow \mathsf{R}^*_\theta[W[\mathsf{DO}(a, s')]]$$

$$\forall s'(\mathsf{K}(s', s) \land (\mathsf{POSS}(a, s'))) \rightarrow \mathsf{R}^*_\theta[W[\mathsf{DO}(a, s')]]^{-1}$$

6. again by definition of **Knows**:

$$\textbf{Knows}(\mathsf{POSS}(a) \rightarrow \mathsf{R}^*_\theta[W[\mathsf{DO}(a, s')]]^{-1}, s)$$

$\square$

Regression Theorem

Theorem (Regression- Knowledge Producing Actions)

$$\mathcal{R}^*_\theta[\textit{\textbf{Knows}}(W, \mathit{DO}(\mathit{SENSE}_i, s))] \equiv$$

$$\exists y \mathit{SR}(\mathit{SENSE}_i, s) = y \wedge$$

$$\textit{\textbf{Knows}}((\mathit{POSS}(a) \wedge \mathit{SR}(\mathit{SENSE}_i) = y) \to \mathcal{R}^*_\theta[W[\mathit{DO}(\mathit{SENSE}_i)]]^{-}1, s)$$

Regression- Knowledge Producing Actions

(vi) $\quad \mathbf{Knows}\big(W, \text{DO}(\text{SENSE}_i, s)\big)$

by the definition of **Knows**

$$\forall s''\ \mathrm{K}\big(s'', \text{DO}(\text{SENSE}_i, s)\big) \rightarrow W[s'']$$

by the successor state axiom for K (sentence (18)), and also and the inductive hypothesis

$$\begin{aligned}&\forall s'\ \big(\mathrm{K}(s', s) \wedge \\ &\qquad \text{POSS}(\text{SENSE}_i, s') \wedge \text{SR}(\text{SENSE}_i, s) = \text{SR}(\text{SENSE}_i, s')\big) \rightarrow \\ &\qquad\quad \mathcal{R}_{\Theta}\big[W[\text{DO}(\text{SENSE}_i, s')]\big]\end{aligned}$$

by the definition of equality and the existential quantifier

$$\begin{aligned}&\forall s'\ \big(\mathrm{K}(s', s) \wedge \\ &\quad \text{POSS}(\text{SENSE}_i, s') \wedge \exists y\ \text{SR}(\text{SENSE}_i, s) = y \\ &\qquad \wedge\ \text{SR}(\text{SENSE}_i, s') = y\big) \rightarrow\ \mathcal{R}_{\Theta}\big[W[\text{DO}(\text{SENSE}_i, s')]\big]\end{aligned}$$

Regression- Knowledge Producing Actions

by the definition of the connectives and quantifiers

$$\begin{aligned}&\forall y\ \textsc{sr}(\textsc{sense}_i, s) = y \ \rightarrow \\ &\quad \forall s' \left(\mathrm{K}(s', s) \wedge \textsc{Poss}(\textsc{sense}_i, s') \wedge \textsc{sr}(\textsc{sense}_i, s') = y\right) \rightarrow \\ &\qquad \mathcal{R}_\Theta\big[W[\textsc{do}(\textsc{sense}_i, s')]\big]\end{aligned}$$

by the definition of the connectives, quantifiers, and the fact that there can only be one denotation of $\textsc{sr}(\textsc{sense}_i, s)$

$$\begin{aligned}&\exists y\ \textsc{sr}(\textsc{sense}_i, s) = y \ \wedge\ \forall s' \left(\mathrm{K}(s', s) \rightarrow \right. \\ &\qquad \left.\textsc{Poss}(\textsc{sense}_i, s') \ \wedge\ \textsc{sr}(\textsc{sense}_i, s') = y\right) \rightarrow \\ &\qquad\quad \mathcal{R}_\Theta\big[W[\textsc{do}(\textsc{sense}_i, s')]\big]\end{aligned}$$

by the definitions of **Knows**

$$\begin{aligned}&\exists y\ \textsc{sr}(\textsc{sense}_i, s) = y \wedge \\ &\qquad \mathbf{Knows}\Big(\big(\textsc{Poss}(\textsc{sense}_i) \wedge \textsc{sr}(\textsc{sense}_i) = y\big) \\ &\qquad\quad \rightarrow \mathcal{R}_\Theta\big[W[\textsc{do}(\textsc{sense}_i, s')]\big]^{-1}, s\Big) \qquad \square\end{aligned}$$

Legality Testing

Method

Given *n* actions, the action precondition axioms are:

$$\forall x_n \ \mathsf{POSS}(A_n(x_n), s) \equiv \Pi A_n(x_n, s)$$

A sequence is a legal action sequence iff:

$$F \models \Pi(\alpha_1)[s \rightarrow S_0] \wedge \Pi(\alpha_2)[s \rightarrow \mathsf{DO}(\alpha_1, S_0)] \wedge \ldots$$

$$\wedge \Pi(\alpha_m)[s \rightarrow \mathsf{DO}([\alpha_1, \ldots, \alpha_{m-1}], S_0)]$$

So iff, by Regression:

$$F - F_{SS} \models \Pi(\alpha_1)[s \rightarrow S_0] \wedge \mathsf{R}^*_\theta[\Pi(\alpha_2)[s \rightarrow \mathsf{DO}(\alpha_1, S_0)]] \wedge \ldots$$

$$\wedge \mathsf{R}^*_\theta[\Pi(\alpha_m)[s \rightarrow \mathsf{DO}([\alpha_1, \ldots, \alpha_{m-1}], S_0)]]$$

Table of Contents

- Scherl and Levesque's Approach
  - Introduction
  - Knowledge as a fluent
  - Defining a successor state axiom for K
- Theorems and Regression
  - Theorems
  - Regression
- Example: The Gardening Robot
  - Axiomatization
  - Instantiation

The Problem

- A robot has to manage **n plants** in a garden.
- The robot performs an action on one plant at a time.
- **A plant can be watered only if it is dry and the temperature is known.**
- The robot has access to a watering can with unlimited capacity.
- The robot performs action on an object only if it is near it.
- The robot can hold only one object at a time.

(Non)Fluents

Fluents

- $\textsc{Near}(x, s) \rightarrow$ Robot is near object x in situation s
- $\textsc{Holding}(x, s) \rightarrow$ Robot is holding object x in situation s
- $\textsc{Moist}(p, s) \rightarrow$ Plant p is moist in situation s
- $\textsc{Temperature}(p, s) \rightarrow$ Temperature value of the spot near plant p
- $\textsc{MoistPlants}(s) \rightarrow$ Number of moist plants

Non-Fluents

- $\textsc{WateringCan}(x) \rightarrow$ Object x is a watering can
- $\textsc{Thermometer}(x) \rightarrow$ Object x is a thermometer
- $\textsc{MoistureMeter}(x) \rightarrow$ Object x is a moisturemeter

Actions

All actions are to be axiomatized as affecting only either the K fluent or other fluents.

Normal

- $\textsc{goto}(x)$ $\rightarrow$ Go to object x
- $\textsc{water}(p)$ $\rightarrow$ Water plant p
- $\textsc{pickup}(x)$ $\rightarrow$ Pick up object x
- $\textsc{putdown}(x)$ $\rightarrow$ Put down object x

Knowledge

- $\textsc{checkmoisture}(p)$ $\rightarrow$ Check moisture of plant p
- $\textsc{checktemperature}(p)$ $\rightarrow$ Check the temperature of the spot near plant p

Effects

Sensing Result Axioms

- $\textsc{sr}(\textsc{goto}(x), s) = r \equiv r = \text{“}\textsc{ok}\text{”}$
- $\textsc{sr}(\textsc{checkMoisture}(p), s) = r \equiv (r = \text{“}\textsc{yes}\text{”} \wedge \textsc{Moist}(p, s)) \vee (r = \text{“}\textsc{no}\text{”} \wedge \neg\textsc{Moist}(p, s))$
- $\textsc{sr}(\textsc{checkTemperature}(p), s) = r \equiv r = \textsc{Temperature}(p, s)$

Knowledge Action Effects

- $\textbf{Kwhether}(\textsc{Moist}(p), \textsc{do}(\textsc{checkMoisture}(p), s))$
- $\textbf{Kref}(\textsc{Temperature}(p), \textsc{do}(\textsc{checkTemperature}(p), s))$

Preconditions

- $\textsc{Poss}\,(\textsc{water}(p), s) \equiv \textsc{Near}(p, s) \land \textsc{Holding}(x, s) \land \textsc{WateringCan}(x) \land$ **Knows**$(\neg\textsc{Moist}(p), s) \land$ **Kref**$(\textsc{Temperature}(p), s)$
- $\textsc{Poss}\,(\textsc{pickup}(x), s) \equiv \textsc{Near}(x, s) \land \neg\exists y.\textsc{Holding}(y, s)$
- $\textsc{Poss}\,(\textsc{putdown}(x), s) \equiv \textsc{Holding}(x, s)$
- $\textsc{Poss}\,(\textsc{checkmoisture}(p), s) \equiv$
  $\textsc{Near}(p, s) \land \textsc{Holding}(x, s) \land \textsc{MoistureMeter}(x)$
- $\textsc{Poss}\,(\textsc{checktemperature}(p), s) \equiv$
  $\textsc{Near}(p, s) \land \textsc{Holding}(x, s) \land \textsc{Thermometer}(x)$

Successor State Axioms

In general $F(x, \textsc{do}(\alpha, s)) \equiv \Phi^+_F(x, a, s) \vee (F(x, s) \wedge \neg\Phi^-_F(x, a, s))$

- $\textsc{Near}(x, \textsc{do}(\alpha, s)) \equiv \alpha = \textsc{goto}(x) \vee (\textsc{Near}(x, s) \wedge \neg\exists y.\alpha = \textsc{goto}(y))$
- $\textsc{Holding}(x, \textsc{do}(\alpha, s)) \equiv \alpha =$ $\textsc{pickup}(x) \vee (Holding(x, s) \wedge \neg\exists r.\alpha = \textsc{putdown}(x))$
- $\textsc{Moist}(p, \textsc{do}(\alpha, s)) \equiv \alpha = \textsc{water}(p) \vee (\textsc{Moist}(p, s) \wedge \neg\exists r.\alpha = \textsc{water}(p))$
- $\textsc{Temperature}(p, \textsc{do}(\alpha, s)) \equiv \textsc{Temperature}(p, s)$
- $\textsc{MoistPlants}(p, \textsc{do}(\alpha, s)) = n \equiv$ $(\textsc{MoistPlants}(p, s) = n - 1 \wedge \alpha = \textsc{water}(p)) \vee (\textsc{MoistPlants}(p, s) = n)$

Theorems

1. Knowledge-producing actions do not change the state of the world. If $\mathrm{P}(s)$ then $\mathrm{P}(\textsc{do}(\alpha, s))$, given that α is knowledge-producing and P is not K
2. Nothing is learned about P by doing action α, as long as α does not affect P.
3. Agents know the consequences of knowledge acquired through knowledge-producing actions.
4. If the agent knows P at s, then P is also known at $\textsc{do}(\alpha, s)$ as long as the effect of α is not to make P false.
5. Agents know the effects of (ordinary) actions.

Initial Situation

- $\textsc{WateringCan}(c)$
- $\textsc{Thermometer}(t)$
- $\textsc{MoistureMeter}(m)$
- $\textsc{Plant}(p_1)$

  $\vdots$

- $\textsc{Plant}(p_n)$
- $\textsc{Near}(c)$

Goal

Have n moist plants, namely $\textsc{MoistPlants}(p, s) = n$

Solution

For every plant p, with the following situations

$$\mathrm{S}_1 = \textsc{do}(\textsc{putdown}(m), \textsc{do}(\textsc{checkmoisture}(p), \textsc{do}(\textsc{goto}(p), \textsc{do}(\textsc{pickup}(m), \textsc{do}(\textsc{goto}(m), \mathrm{S}_0)))))$$

$$\mathrm{S}_2 = \textsc{do}(\textsc{putdown}(t), \textsc{do}(\textsc{checktemperature}(p), \textsc{do}(\textsc{pickup}(t), \textsc{do}(\textsc{goto}(t), \mathrm{S}_1))))$$

$$\mathrm{S}_3 = \textsc{do}(\textsc{putdown}(c), \textsc{do}(\textsc{water}(p), \textsc{do}(\textsc{pickup}(c), \textsc{do}(\textsc{goto}(c), \mathrm{S}_2))))$$

In the end, we need to entail that $\textsc{MoistPlants}(p, s) = n$
We can use the successor state axioms for that

Legality testing (with a different S_0)

$$\textsc{do}(\textsc{water}(p), \textsc{do}(\textsc{pickup}(c), \textsc{do}(\textsc{goto}(c), \mathrm{S}_0)))$$

$$\Downarrow$$

We need to entail

$$\textsc{Near}(p, \mathrm{S}_0) \wedge \textsc{Holding}(t, \mathrm{S}_0) \wedge \textsc{Thermometer}(t, \mathrm{S}_0) \wedge \textbf{Knows}(\textsc{Moist}(p, \mathrm{S}_0)) \wedge$$
$$\textbf{Kref}(\textsc{Temperature}(p), \textsc{do}(\textsc{checktemperature}(p), \mathrm{S}_0))$$

$$\Downarrow$$

$$\exists y.y = \textsc{Temperature}(p, s)$$

Golog

```
while ¬Knows(HealthyPlants = n) do
    (Πp) goto(p);
        checkMoisture(p);
        if Knows(¬Moist(p))
            goto(t);
                pickup(t);
                goto(p);
                checkTemperature(p);
                putdown(t);
            goto(c);
                pickup(c);
                goto(p);
                water(p);
                putdown(c);
end while
```

Knowledge, action, and the frame problem

Thank you for listening!

Any questions?