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
- Regression
- Example: The Gardening Robot
  - Axiomatization
  - Golog

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)

...

Opzionale
Un paio di azioni ordinarie e un paio di azioni di conoscenza di esempio, giusto per inquadrare il discorso

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

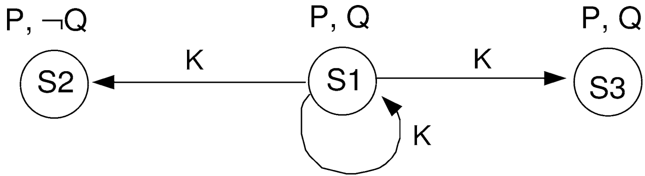
Learn the value of a term. Example: read a number on a sheet of paper.

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

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.



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

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

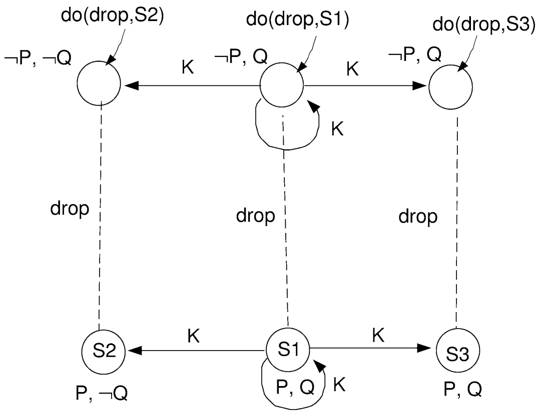
The **only knowledge gained is that the** DROP **action has been performed**, as well as anything that can be derived from the action effects.
For example, if DROP makes P false:

$$\mathrm{P}(\mathrm{DO}(a, s)) \equiv a \neq \mathrm{DROP} \wedge \mathrm{P}(s)$$

then

$$\textbf{Knows}(\neg\mathrm{P}, \mathrm{DO}(\mathrm{DROP}, S1))$$

but no extra knowledge is gained about Q.



Knowledge-producing actions

Consider an action $\textsc{sense}_{\mathrm{Q}}$ 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

$$\textsc{sr}(\textsc{sense}_{\mathrm{Q}}, s) = r \equiv (r = \text{``}\textsc{yes}\text{''} \wedge \mathrm{Q}(s)) \vee (r = \text{``}\textsc{no}\text{''} \wedge \neg \mathrm{Q}(s))$$

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.

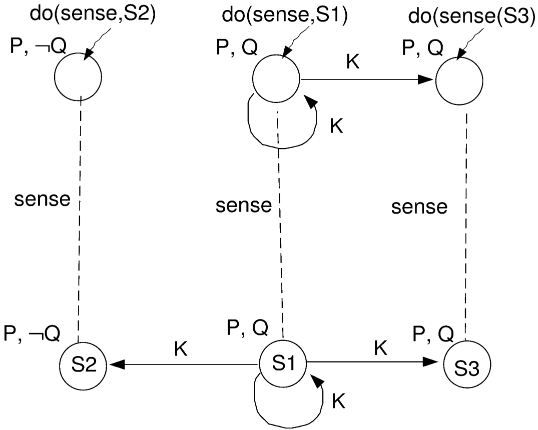
Axiomatization

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

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... (opzionale)

- **...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**$(\forall x(\textsc{man}(x) \rightarrow \textsc{mortal}(x)) \wedge \textsc{man}(Socrates))$

<varie ed eventuali>

boh

Table of Contents

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

Table of Contents

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

The Problem

- A robot has to manage **n plants** in a garden.
- The robot performs an action on one plant at a time and only if it's near the plant.
- A plant can be watered only if it is dry and the temperature is known.
- The robot has access to a watering can that is full and has unlimited capacity.
- 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
- $\textsc{Temperature}(p) \rightarrow$ Temperature value of the spot near plant p
- $\textsc{HealthyPlants}(s) \rightarrow$ Number of healthy 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

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

- $\textsc{sr}(\textsc{goto}(x), s) = r \equiv r = "OK"$
- $\textsc{sr}(\textsc{checkmoisture}(p), s) = r \equiv (r = "YES" \land \textsc{Moist}(p, s)) \lor (r = "NO" \land \neg\textsc{Moist}(p, s))$
- $\textsc{sr}(\textsc{checktemperature}(p), s) = r \equiv r = \textsc{Temperature}(p, s)$

Knowledge Action Effects

- **Kwhether**$(\textsc{Moist}(p, s), \textsc{do}(\textsc{checkmoisture}(p), s))$
- **Kref**$(\textsc{Temperature}(p), \textsc{do}(\textsc{checktemperature}(p), s))$

Preconditions

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

Successor State Axioms

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

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

Initial Situation

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

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

Encoding

while $\neg\mathbf{Knows}(\textsc{HealthyPlants}(s) = n)$ **do** $\textsc{goto}(\mathsf{p})$ Ok
end while

Example Run

Knowledge, action, and the frame problem

Thank you for listening!

Any questions?