What Observations Really Tell Us

Gero Iwan and **Gerhard Lakemeyer**

Department of Computer Science V
Aachen University of Technology
52056 Aachen, Germany
{iwan,gerhard}@cs.rwth-aachen.de

Abstract

When agents like mobile robots make observations while carrying out a course of actions, a formalization of the observed information is needed in order to reason about it. When doing so in the situation calculus, a seemingly straightforward approach turns out to be inappropriate since it leads to unintended results and has an unfortunate sensitivity with respect to different forms of successor state axioms. In this paper we suggest how to properly encode observed information in order to avoid both of these problems.

1 Introduction

When agents like mobile robots make observations while carrying out a course of actions, this information is of use when reasoning about the future as in planning, but also when reasoning about the past, which is necessary, for example, when diagnosing execution failures. Here we are considering formalizing actions and observations in the situation calculus (McCarthy 1963; Levesque, Pirri, & Reiter 1998; Reiter & Pirri 1999).

At first glance, it seems to be quite clear what knowledge observations provide the agent with, namely that a certain statement about the world is true at a given point during its course of actions. However, when we began extending Iwan's work on the diagnosis of plan execution failures (Iwan 2002) to the case that observations can be made during a course of actions, we realized that a seemingly straightforward formalization of the information provided by the observations may lead to unintended and/or unintuitive results when drawing conclusions from these observations. Moreover, in this case different forms of the so-called successor state axioms which describe the effect and non-effects of actions yield different results although switching between these different forms was thought to be innocuous. We will suggest how to properly formalize the information provided by observations in order to avoid both of these problems.

We will illustrate the problem and our solution by way of the following simple scenario. Suppose an autonomous robot is acting in a typical office environment with rooms $R1, R2, R3, \ldots, Rm$ each of which is connected by a door to a hallway H. Starting from the hall in the initial situation, as the first action in a course of actions, the robot wants to enter room $R1$ and initiates navigating towards room $R1$.[1] But then, after arriving at the room, it finds out that it is not $R1$. An reasonable diagnosis of this plan execution failure would be that the robot entered a room different from $R1$. In order to figure out what actually happened it would be helpful to determine the rooms for which it was possible to enter them in the initial situation. Assume that it is possible to enter a room from the hall iff the door of the room is open. Let us consider three cases:

1. Suppose that nothing is known about the state of the doors in the initial situation except that the door of room $R1$ was open. Without any further information, one can only infer that it was possible to enter room $R1$ and that it was at best *potentially* possible to enter any other room.
2. If the robot is capable of inquiring about the state of the doors (e. g., from a door control system) it can obtain new information. For example, let the answer to the request be: rooms $R1, R2, R3$ are open, all other rooms are closed. From this information one should be able to infer that it was possible to enter rooms $R2, R3$ instead of $R1$, but no other room.
3. Instead of requesting the door states from an outside source, suppose the robot uses its own sensors and observes, for example, that it is in a room whose door is open. In a sense, this seems like a rather trivial observation because having just entered a room should already imply that the door to this room is open. Thus from this redundant observation one should not be able to infer more than in the case where no observation was made.

As we will see below, there are surprisingly subtle issues that arise when attempting to formalize such scenarios.

The rest of the paper is organized as follows. In the next section, we briefly introduce the situation calculus, followed by a first attempt at formalizing and reasoning about observations using the robot example. In Section 4, we analyze the problems with this approach and discuss our solution. In Section 5, we look at the more general picture of projecting both forward and backward in time. Finally, we end the paper with a section on related work and a brief summary.

[1] We assume a navigation software as in (Burgard *et al.* 1999) which usually, but not always, leads to successful navigation from one location to another.

2 The Situation Calculus

In this section we briefly go over the situation-as-histories variant (Levesque, Pirri, & Reiter 1998; Reiter & Pirri 1999; Reiter 2001) of the *situation calculus* (McCarthy 1963; McCarthy & Hayes 1969) and, as we go along, give a formal account of our robot scenario in this framework. We adopt the convention that free variables are implicitly universally quantified unless otherwise stated. $\Phi[x_1, \ldots, x_n]$ indicates that the free variables of formula Φ are among $x_1, \ldots, x_n$.

The language of the situation calculus has three disjoint sorts: *situation*, *action*, *object*. There are two function symbols of sort *situation*: the constant S_0, which denotes the initial situation, and the binary function symbol do, where $do(\alpha, \sigma)$ denotes the situation that is reached after executing action α in situation σ. There are several domain-independent foundational axioms which, among other things, characterize the predicate $\sqsubset$ which denotes the predecessor relation between situations: $s' \sqsubset s$ means that s can be reached from s' by a sequence of actions. The abbreviation $s' \sqsubseteq s$ stands for $s' \sqsubset s \vee s' = s$.

An action is a term $A(t_1, \ldots, t_n)$, where A is a n-ary action function. The action functions used here are $enter$ and $leave$ for entering and leaving a room, $open$, $close$ and $lock$ for opening, closing and locking doors. In the example, the rooms and the hall are denoted by the object constants $R1, R2, R3, \ldots, Rm$ and H. Properties of situations are represented using so-called fluents. Here we confine ourselves to relational fluents, which are predicates whose last argument is of sort *situation*, e. g., $Locked(r, s)$, $Open(r, s)$ and $RoLoc(l, s)$, whose meaning is "The (door of) room r is locked in situation s", "The (door of) room r is open in situation s" and "The robot is at location l in situation s", respectively. For each fluent there is a successor state axiom of the form

$$F(x_1, \ldots, x_n, do(a, s)) \equiv \Phi_F[x_1, \ldots, x_n, a, s] \quad \text{(SSA)}$$

which states under which condition, Φ_F, the property $F(x_1, \ldots, x_n, \_)$ holds in the successor situation $do(a, s)$ of situation s after executing action a, e. g.,

$$\begin{aligned}
&Locked(r, do(a, s)) \\
&\quad \equiv\ a = lock(r)\ \vee\ Locked(r, s) \\
&Open(r, do(a, s)) \\
&\quad \equiv\ \ a = open(r) \\
&\qquad \vee [Open(r, s) \wedge \neg[a = close(r) \vee a = lock(r)]\,] \\
&RoLoc(l, do(a, s)) \\
&\quad \equiv\ \ a = enter(l) \\
&\qquad \vee [a = leave \wedge l = H] \\
&\qquad \vee [RoLoc(l, s) \wedge \neg[\exists r\, a = enter(r) \vee a = leave]\,]
\end{aligned}$$

Axioms describing the initial situation and situation independent facts form the initial database, e. g.,

$$\begin{aligned}
&RoLoc(H, S_0)\ \ \wedge\ \ Open(R1, S_0) \\
&\neg[Locked(r, S_0) \wedge Open(r, S_0)] \\
&[Open(r, S_0) \vee Locked(r, S_0)] \supset Room(r) \\
&Room(r) \equiv\ r = R1 \vee r = R2 \vee \cdots \vee r = Rm \\
&\text{unique names axioms for } R1, \ldots, Rm \text{ and } H
\end{aligned}$$

Action precondition axioms of the form

$$Poss(A(x_1, \ldots, x_n), s) \equiv \Pi_A[x_1, \ldots, x_n, s]$$

are used to state under which condition, Π_A, it is possible to execute action $A(x_1, \ldots, x_n)$ in situation s, e. g.,

$$\begin{aligned}
&Poss(lock(r), s) \\
&\quad \equiv\ \ Room(r) \wedge [RoLoc(r, s) \vee RoLoc(H, s)] \\
&Poss(close(r), s) \\
&\quad \equiv\ \ Room(r) \wedge [RoLoc(r, s) \vee RoLoc(H, s)] \\
&Poss(open(r), s) \\
&\quad \equiv\ \ Room(r) \wedge [RoLoc(r, s) \vee RoLoc(H, s)] \\
&\qquad \wedge \neg Locked(r, s) \\
&Poss(enter(r), s) \\
&\quad \equiv\ \ RoLoc(H, s) \wedge Open(r, s) \\
&Poss(leave, s) \\
&\quad \equiv \exists r\, [RoLoc(r, s) \wedge Open(r, s)]
\end{aligned}$$

A situation is said to be *executable* if, starting in the initial situation, it is possible to execute all the actions that lead to the situation. Formally:

$$Exec(s) \quad \doteq \quad \forall a', s'\, [do(a', s') \sqsubseteq s \supset Poss(a', s')]$$

As discussed in detail in (Reiter 2001), a basic action theory $\mathcal{D}$ describes the initial state of the world and how the world evolves under the effects of actions. It consists of foundational axioms for situations, unique names axioms for actions, an action precondition axiom for each action function, a successor state axiom for each fluent, and axioms describing the initial situation and situation independent facts.[2]

In what follows we also need the notion of *situation-suppressed formulas*, i. e., formulas where all occurrences of situation terms are "deleted" (details omitted). If ϕ is a situation-suppressed formula then $\phi[\sigma]$ denotes the situation calculus formula obtained after restoring suppressed situation arguments by "inserting" the situation σ where necessary, e. g., if $\phi = \exists x\, (Room(x) \wedge RoLoc(x) \wedge Open(x))$ then $\phi[\sigma] = \exists x\, (Room(x) \wedge RoLoc(x, \sigma) \wedge Open(x, \sigma))$.

In contrast to the form of successor state axioms used above the form

$$Poss(a, s) \supset \big(\ F(\vec{x}, do(a, s)) \equiv \Phi_F[\vec{x}, a, s]\ \big) \quad \text{(PSSA)}$$

is also found in the literature (e. g., in (Reiter 1991; Levesque *et al.* 1997; McIlraith 1998)) and is in fact the "original" form from (Reiter 1991). We refer to it as the $Poss$-guarded form, and it only allows to infer effects of executable actions. Note that the unguarded form is logically equivalent to the $True$-guarded form

$$True \supset \big(\ F(\vec{x}, do(a, s)) \equiv \Phi_F[\vec{x}, a, s]\ \big) \qquad \text{(TSSA)}$$

Hence, from now on, we will write $\mathcal{D}_{True}$ for basic action theory using the unguarded form (or the $True$-guarded form) and $\mathcal{D}_{Poss}$ for basic action theory using the $Poss$-guarded form.

[2] Note that with the given action theory for the robot example, e. g., $\mathcal{D} \models \forall r, s\, [(Exec(s) \wedge Locked(r, s)) \supset \neg Open(r, s)]$.

$\mathcal{D}_{True}$ and $\mathcal{D}_{Poss}$ are equivalent w. r. t. planning and projection for executable situations, strictly speaking *projection into the future* which means reasoning forward (in time) (cf. Section 5). But the example in the next section seems to suggest that they are not equivalent w. r. t. *projection into the past*, that is, reasoning backward in time. Worse yet, both forms suddenly seem inappropriate. However, Sections 4 and 5 will show how both problems can be avoided by accurately formalizing the information contained in observations.

3 The Robot Example Revisited

With the situation-calculus account of the robot example given in the previous section, let us now reconsider the situation where the robot planned to perform the action $enter(R1)$ but fails. According to (Iwan 2002), $enter(\varrho)$ is a diagnosis of this plan execution failure for any $\varrho \in \{R2, \ldots, Rm\}$ provided the executability of $enter(\varrho)$ can be proved. Let ϕ be a situation-suppressed closed formula representing the observation that was made after performing the action $enter(\varrho)$. Then $\phi\big[do(enter(\varrho), S_0)\big]$ is the additional information provided by this observation. Since we want to determine whether it was possible to enter room ϱ in the initial situation we consider the sets of rooms $\mathsf{R}_1^{\oplus}(\mathcal{D}, \phi)$ and $\mathsf{R}_1^{\ominus}(\mathcal{D}, \phi)$ *intended* to be the sets of rooms different from $R1$ for which we are able to infer that it was possible or impossible to enter them respectively:[3]

$$\begin{aligned}\mathsf{R}_1^{\oplus}(\mathcal{D}, \phi) = \{\varrho \in \{R2, \ldots, Rm\} \mid \\ \mathcal{D} \wedge \phi\big[do(enter(\varrho), S_0)\big] \models Open(\varrho, S_0)\}\\ \mathsf{R}_1^{\ominus}(\mathcal{D}, \phi) = \{\varrho \in \{R2, \ldots, Rm\} \mid \\ \mathcal{D} \wedge \phi\big[do(enter(\varrho), S_0)\big] \models \neg Open(\varrho, S_0)\}\end{aligned}$$

Note that, no matter which form of successor state axioms is used, $Poss(enter(\varrho), S_0) \equiv Open(\varrho, S_0)$ can be inferred from the action precondition axiom for $enter$ together with $RoLoc(H, S_0)$.

The three cases regarding the information about the door states can be represented as follows:

$$\begin{aligned}\phi_{\mathrm{N}} &= True,\\ \phi_{\mathrm{R}} &= \forall r\,[Open(r) \equiv r = R1 \vee r = R2 \vee r = R3]\\ \phi_{\mathrm{L}} &= \exists r\,[Room(r) \wedge RoLoc(r) \wedge Open(r)]\end{aligned}$$

where ϕ_{N} expresses that no observation was made, ϕ_{R} corresponds to the case where the agent was told that only the first 3 rooms are open, and ϕ_{L} corresponds to the case where the agent itself is able to figure out that the door to the room it just entered is open.

Let $\mathsf{R}^{\oplus}(\phi)$ and $\mathsf{R}^{\ominus}(\phi)$ denote the intended results for an observation ϕ. The argument given in the introduction suggests the following values for $\mathsf{R}^{\oplus}(\phi)$ and $\mathsf{R}^{\ominus}(\phi)$, respectively:

$$\begin{aligned}&\mathsf{R}^{\oplus}(\phi_{\mathrm{R}}) = \{R2, R3\} && \mathsf{R}^{\oplus}(\phi_{\mathrm{L}}) = \emptyset && \mathsf{R}^{\oplus}(\phi_{\mathrm{N}}) = \emptyset\\ &\mathsf{R}^{\ominus}(\phi_{\mathrm{R}}) = \{R4, \ldots, Rm\} && \mathsf{R}^{\ominus}(\phi_{\mathrm{L}}) = \emptyset && \mathsf{R}^{\ominus}(\phi_{\mathrm{N}}) = \emptyset\end{aligned}$$

[3] In a slight abuse of notation, we often write $\mathcal{D} \wedge \Phi$ instead of $\mathcal{D} \cup \{\Phi\}$.

However, depending on whether we use $Poss$- or $True$-guarded successor state axioms, we sometimes get surprisingly different results as shown in the following table:

ϕ	ϕ_{R}	ϕ_{L}	ϕ_{N}
$\mathsf{R}_1^{\oplus}(\mathcal{D}_{True}, \phi)$	$\{R2, R3\}$	$\{R2, \ldots, Rm\}$	$\emptyset$
$\mathsf{R}_1^{\ominus}(\mathcal{D}_{True}, \phi)$	$\{R4, \ldots, Rm\}$	$\emptyset$	$\emptyset$
$\mathsf{R}_1^{\oplus}(\mathcal{D}_{Poss}, \phi)$	$\emptyset$	$\emptyset$	$\emptyset$
$\mathsf{R}_1^{\ominus}(\mathcal{D}_{Poss}, \phi)$	$\{R4, \ldots, Rm\}$	$\emptyset$	$\emptyset$

Thus for the observation ϕ_{R}, $\mathcal{D}_{Poss}$ is too weak but $\mathcal{D}_{True}$ gives the intended results, whereas for ϕ_{L}, $\mathcal{D}_{Poss}$ gives the intended results and $\mathcal{D}_{True}$ gives a completely counter-intuitive result.

This result can be meliorated by a more careful formulation of the sets $\mathsf{R}_1^{\oplus}(\mathcal{D}, \phi)$ and $\mathsf{R}_1^{\ominus}(\mathcal{D}, \phi)$:

$$\begin{aligned}\mathsf{R}_2^{\oplus}(\mathcal{D}, \phi) = \{\xi \in \{R2, \ldots, Rm\} \mid \\ \mathcal{D} \wedge \phi\big[do(enter(\varrho), S_0)\big] \models Open(\xi, S_0)\}\\ \mathsf{R}_2^{\ominus}(\mathcal{D}, \phi) = \{\xi \in \{R2, \ldots, Rm\} \mid \\ \mathcal{D} \wedge \phi\big[do(enter(\varrho), S_0)\big] \models \neg Open(\xi, S_0)\}\end{aligned}$$

Here we distinguish between the room $\varrho \neq R1$ which is assumed to be entered and the room ξ for which we are able to infer that it was possible or impossible to enter it. Determining the sets of rooms yields the same results except that (*1.*) for observation ϕ_{R}, $\mathcal{D}_{Poss}$ becomes even weaker since $\mathsf{R}_2^{\ominus}(\mathcal{D}_{Poss}, \phi_{\mathrm{R}}) = \emptyset$ if $\varrho \in \{R2, R3\}$ and $\mathsf{R}_2^{\ominus}(\mathcal{D}_{Poss}, \phi_{\mathrm{R}}) = \{\varrho\}$ if $\varrho \in \{R4, \ldots, Rm\}$, and (*2.*) for the other observation, $\mathsf{R}_2^{\oplus}(\mathcal{D}_{True}, \phi_{\mathrm{L}}) = \{\varrho\}$:

ϕ	ϕ_{R}	ϕ_{L}	ϕ_{N}
$\mathsf{R}_2^{\oplus}(\mathcal{D}_{True}, \phi)$	$\{R2, R3\}$	$\{\varrho\}$	$\emptyset$
$\mathsf{R}_2^{\ominus}(\mathcal{D}_{True}, \phi)$	$\{R4, \ldots, Rm\}$	$\emptyset$	$\emptyset$
$\mathsf{R}_2^{\oplus}(\mathcal{D}_{Poss}, \phi)$	$\emptyset$	$\emptyset$	$\emptyset$
$\varrho \in \{R2, R3\}$:			
$\mathsf{R}_2^{\ominus}(\mathcal{D}_{Poss}, \phi)$	$\emptyset$	$\emptyset$	$\emptyset$
$\varrho \in \{R4, \ldots, Rm\}$:			
$\mathsf{R}_2^{\ominus}(\mathcal{D}_{Poss}, \phi)$	$\{\varrho\}$	$\emptyset$	$\emptyset$

Given the intuition that $\mathsf{R}^{\oplus}(\phi_{\mathrm{L}}) = \emptyset$ the result $\mathsf{R}_2^{\oplus}(\mathcal{D}_{True}, \phi_{\mathrm{L}}) = \{\varrho\}$ is "less counter-intuitive" but nevertheless strange because it is trivial that the robot can leave a room it just entered. Thus, the robot should not gain any information from the observation ϕ_{L} after an $enter$-action. Let us consider the argument for $\mathcal{D}_{True}$ and ϕ_{L} in a bit more detail:

1. An effect of the execution of $enter(\varrho)$ is that $RoLoc(r) \equiv r = \varrho$, i. e., the location of the robot is ϱ [by the (unguarded) successor state axiom for $RoLoc$].
2. If the location of the robot is ϱ then the observation ϕ_{L} is equivalent to $Open(\varrho)$, i. e., room ϱ is open.

3. If room ϱ is open after action $enter(\varrho)$ then room ϱ must have been open before action $enter(\varrho)$ [by the (unguarded) successor state axiom for $Open$].
4. If room ϱ is open before $enter(\varrho)$ then $enter(\varrho)$ is executable (since the location was the hallway before) [by the action precondition axiom for $enter$].

So we have a kind of circular argument here: "An effect of the execution of $enter(\varrho)$ is that ... $enter(\varrho)$ is executable." The crucial point in this circular argument is the first one which is considering effects of actions that are not known to be executable.

This discovery leads us to a defect of the formalization of observed information so far: we consider observations made in situations which are not known to be executable. But this seems absurd: if we make an observation in some situation then we have reached this situation, and if we have reached some situation then the actions leading to this situation must have been executable. So it is revealed now that the intuition $\mathsf{R}^{\oplus}(\phi_{\mathrm{L}}) = \emptyset$ was wrong if we assume that a room ϱ was entered. The right intuition should have been $\mathsf{R}^{\oplus}(\phi_{\mathrm{L}}) = \{\varrho\}$. Generally, $\varrho \in \mathsf{R}^{\oplus}(\dots)$ must hold. For ϕ_{N} this brings $\mathsf{R}^{\oplus}(\phi_{\mathrm{N}}) = \{\varrho\} \neq \emptyset = \mathsf{R}_2^{\oplus}(\mathcal{D}, \phi_{\mathrm{N}})$. And $\mathcal{D} \wedge \phi_{\mathrm{R}}\big[do(enter(\varrho), S_0)\big]$ should turn out to be inconsistent if $\varrho \notin \mathsf{R}^{\oplus}(\phi_{\mathrm{R}}) = \{R2, R3\}$ ($\varrho \neq R1$ was presupposed). This is not the case. Rather there are models where $\neg Open(\varrho, S_0)$ is true. Moreover, if $\varrho \in \{R4, \dots, Rm\}$ then $\mathcal{D} \wedge \phi_{\mathrm{R}}\big[do(enter(\varrho), S_0)\big] \models \neg Open(\varrho, S_0)$. But this means that $enter(\varrho)$ was not executable, contradicting the assumption that ϱ was entered. To resolve this problem, we need to be more careful in representing what observations really tell us.

4 What Observations Really Tell Us

We define an *observation* as a situation-suppressed closed formula. In order to obtain the information provided by an observation one has to restore an appropriate situation, namely the situation where the observation was made. When an agent, while carrying out a course of actions, makes several observations $\phi_1, \dots, \phi_n$ in sequence the *basic information* contained in this sequence of observations is

- For each observation ϕ_i there is a situation s_i where it was made.
- There is a situation s_* which is the current situation and therefore is executable.
- The sequence $s_1, \dots, s_n, s_*$ provides us with a natural ordering of the situations.

This is captured by the formula $\overline{\exists}\,\Omega[\phi_1, \dots, \phi_n]$ where

$$\begin{aligned}\Omega[\phi_1, \dots, \phi_n] \quad \doteq \quad & s_1 \sqsubseteq \dots \sqsubseteq s_n \sqsubseteq s_* \\ & \wedge\, Exec(s_*) \\ & \wedge\, \phi_1\big[s_1\big] \wedge \dots \wedge \phi_n\big[s_n\big]\end{aligned}$$

Here $s_1 \sqsubseteq s_2 \wedge \dots \wedge s_{n-1} \sqsubseteq s_n \wedge s_n \sqsubseteq s_*$ is abbreviated by $s_1 \sqsubseteq \dots \sqsubseteq s_n \sqsubseteq s_*$. Note that, as a consequence of the foundational situation calculus axioms, $s \sqsubseteq s_* \wedge Exec(s_*)$ implies $Exec(s)$.

The message then is that the sole information as given by the observation formulas is not enough to draw the right conclusions, but that we also need to take into account the history and, in particular, the fact that it is executable.

In the robot example, the basic information contained in ϕ_{L} is no new information since $\mathcal{D}_{True}$ and $\mathcal{D}_{Poss}$ both imply $\overline{\exists}\,\Omega[\phi_{\mathrm{L}}]$: they both imply $\sigma_1 \sqsubseteq \sigma_* \wedge Exec(\sigma_*) \wedge \phi_{\mathrm{L}}\big[\sigma_1\big]$, e. g., with $\sigma_1 = \sigma_* = do(enter(R1), S_0)$. In general, the basic information may contain new information. For instance, for both $\mathcal{D} = \mathcal{D}_{True}$ and $\mathcal{D} = \mathcal{D}_{Poss}$

$$\mathcal{D} \wedge \overline{\exists}\,\Omega[\phi_{\mathrm{R}}] \models \neg Locked(R2, S_0) \wedge \neg Locked(R3, S_0)$$

Without this information we have[4] $\mathcal{D} \models \neg Locked(R1, S_0)$ but

$$\mathcal{D} \not\models \neg Locked(R2, S_0) \quad \text{and} \quad \mathcal{D} \not\models \neg Locked(R3, S_0)$$

However, mostly we have some additional assumptions about the action history. In the robot example, the assumption was $s_1 = do(enter(\varrho), S_0)$ (with $\varrho \neq R1$). Let this formula be named Θ_ϱ.

A *history assumption* for $\Omega[\phi_1, \dots, \phi_n]$ can be expressed by a formula $\Theta\lceil s_1, \dots, s_n, s_*\rceil$ (i. e., by a formula Θ with free variables among $s_1, \dots, s_n, s_*$) where $s_1, \dots, s_n, s_*$ are the variables chosen in $\Omega[\phi_1, \dots, \phi_n]$. The basic information together with the history assumption then is $\overline{\exists}\,[\Omega[\phi_1, \dots, \phi_n] \wedge \Theta]$.

We accordingly redefine the sets $\mathsf{R}_2^{\oplus}(\mathcal{D}, \phi)$ and $\mathsf{R}_2^{\ominus}(\mathcal{D}, \phi)$ from the previous section:

$$\begin{aligned}\mathsf{R}_\varrho^{\oplus}(\mathcal{D}, \phi) = \{\xi \in \{R2, \dots, Rm\} \mid \\ \mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi] \wedge \Theta_\varrho] \models Open(\xi, S_0)\} \\ \mathsf{R}_\varrho^{\ominus}(\mathcal{D}, \phi) = \{\xi \in \{R2, \dots, Rm\} \mid \\ \mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi] \wedge \Theta_\varrho] \models \neg Open(\xi, S_0)\}\end{aligned}$$

Note that $\overline{\exists}\,[\Omega[\phi] \wedge \Theta_\varrho]$ is equivalent to

$$Poss(enter(\varrho), S_0) \wedge \phi\big[do(enter(\varrho), S_0)\big]$$

(since the foundational situation calculus axioms imply $Exec(do(a, S_0)) \equiv Poss(a, S_0)$). $\mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi_{\mathrm{R}}] \wedge \Theta_\varrho]$ is inconsistent if $\varrho \notin \{R1, R2, R3\}$. Therefore the ϕ_{R}-results are only reported for $\varrho \in \{R2, R3\}$ ($\varrho \neq R1$ was presupposed):

ϕ	$\phi_{\mathrm{R}} \quad (\varrho \in \{R2, R3\})$	ϕ_{L}	ϕ_{N}
$\mathsf{R}_\varrho^{\oplus}(\mathcal{D}_{True}, \phi)$	$\{R2, R3\}$	$\{\varrho\}$	$\{\varrho\}$
$\mathsf{R}_\varrho^{\ominus}(\mathcal{D}_{True}, \phi)$	$\{R4, \dots, Rm\}$	$\emptyset$	$\emptyset$
$\mathsf{R}_\varrho^{\oplus}(\mathcal{D}_{Poss}, \phi)$	$\{R2, R3\}$	$\{\varrho\}$	$\{\varrho\}$
$\mathsf{R}_\varrho^{\ominus}(\mathcal{D}_{Poss}, \phi)$	$\{R4, \dots, Rm\}$	$\emptyset$	$\emptyset$

So for each of the three observations both $\mathcal{D}_{True}$ and $\mathcal{D}_{Poss}$ give the intended results that were discussed at the end of the

[4] Because of the axioms describing the initial situation we have $\mathcal{D} \models Open(R1, S_0) \wedge \forall r\, \neg[Locked(r, S_0) \wedge Open(r, S_0)]$.

previous section (including the inconsistence for observation ϕ_{R} if $\varrho \notin \{R2, R3\}$).

Of course, in contrast to the previous section, here the primary statement that $enter(\varrho)$ is executable in the initial situation is given additionally. This was not done in the previous section. But the absence of this premise was exactly the shortcoming of the formalization of observed information in the previous section. Note that the premise $Poss(enter(\varrho), S_0)$ does not originate from the basic information contained in the observation but from the history assumption (together with the basic information).

> The explicit distinction between the basic information contained in the observations on the one hand and the *assumption* about the history on the other hand is a valuable quality of the approach to observation information presented here.

Instead of explicitly introducing the entered room ϱ we could have used a modified history assumption

$$\exists r\,[Room(r) \wedge r \neq R1 \wedge s_* = s_1 = do(enter(r), S_0)]$$

which is closer to what we wanted to express actually. Let this formula be named $\Theta_\exists$ and used within $\mathsf{R}^{\oplus}_{\varrho}(\mathcal{D}, \phi)$ and $\mathsf{R}^{\ominus}_{\varrho}(\mathcal{D}, \phi)$ instead of Θ_ϱ yielding:

$$\mathsf{R}^{\oplus}_{\exists}(\mathcal{D}, \phi) = \{\xi \in \{R2, \ldots, Rm\} \mid \mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi] \wedge \Theta_\exists] \models Open(\xi, S_0)\}$$

$$\mathsf{R}^{\ominus}_{\exists}(\mathcal{D}, \phi) = \{\xi \in \{R2, \ldots, Rm\} \mid \mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi] \wedge \Theta_\exists] \models \neg Open(\xi, S_0)\}$$

Note that $\overline{\exists}\,[\Omega[\phi] \wedge \Theta_\exists]$ is equivalent to

$$\exists r\,[\quad Room(r) \wedge r \neq R1 \\ \wedge\, Poss(enter(r), S_0) \wedge \phi\big[do(enter(r), S_0)\big]\,]$$

Furthermore $\mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi_{\mathrm{R}}] \wedge \Theta_\exists]$ is consistent. The results now are:

ϕ	ϕ_{R}	ϕ_{L}	ϕ_{N}
$\mathsf{R}^{\oplus}_{\exists}(\mathcal{D}_{True}, \phi)$	$\{R2, R3\}$	$\emptyset$	$\emptyset$
$\mathsf{R}^{\ominus}_{\exists}(\mathcal{D}_{True}, \phi)$	$\{R4, \ldots, Rm\}$	$\emptyset$	$\emptyset$
$\mathsf{R}^{\oplus}_{\exists}(\mathcal{D}_{Poss}, \phi)$	$\{R2, R3\}$	$\emptyset$	$\emptyset$
$\mathsf{R}^{\ominus}_{\exists}(\mathcal{D}_{Poss}, \phi)$	$\{R4, \ldots, Rm\}$	$\emptyset$	$\emptyset$

So for each of the three observations both $\mathcal{D}_{True}$ and $\mathcal{D}_{Poss}$ give what we expect to be the intended results as initially stated at the beginning of the previous section.

Our robot example is a projection problem into the past where the form of successor state axioms does not matter. In the next section we will see that with our approach to observation information this is true in a rather general sense (correcting the impression that was given in Section 3).

5 Planning and Projection with Observations

In the last section, we used inferences like $\mathcal{D} \wedge \overline{\exists}\,[\Omega[\phi_1, \ldots, \phi_n] \wedge \Theta] \models \Psi$. This can be rephrased as

$$\mathcal{D} \models \overline{\forall}\,[(\Omega[\phi_1, \ldots, \phi_n] \wedge \wr\Theta) \supset \Psi] \qquad (\star)$$

where $\wr\Theta$ is obtained from Θ by removing all $\exists$-quantifiers from the front of Θ (so $\Theta = \overline{\exists}\,\wr\Theta$). Note though that $(\star)$ is more general since Ψ may refer to the free variables of $\Omega[\phi_1, \ldots, \phi_n] \wedge \wr\Theta$. This feature is needed, e. g., for general reachability, (re-)planning and projection problems.

For instance, $\wr\Theta_\exists$ is

$$Room(r) \wedge r \neq R1 \wedge s_* = s_1 = do(enter(r), S_0)$$

and we can ask for properties of r, e. g., whether r is currently open which is obviously true (since the robot just entered r): $\mathcal{D} \models \overline{\forall}\,[(\Omega[\phi] \wedge \wr\Theta_\exists) \supset Open(r, s_*)]$. Note that this could not be inferred if $\Theta_\exists$ would not contain $s_* = s_1$. Likewise, if we would like to know whether it is possible to recover from error in our robot example we have to check whether the original goal of the robot can be reached from the current situation. This is a reachability problem. Its general form is

$$\mathcal{D} \models \overline{\forall}\,[(\Omega[\phi_1, \ldots, \phi_n] \wedge \wr\Theta) \supset \exists s\,[s_* \sqsubseteq s \wedge Exec(s) \wedge \Gamma(s)]\,] \qquad \textbf{(reachability)}$$

where s_* is the variable referring to the current situation in $\Omega[\phi_1, \ldots, \phi_n]$.

(Re-)Planning means: Find an action sequence $\alpha_1, \ldots, \alpha_m$ and check that it is executable in the current situation and achieves the goal?[5] The general form therefore is with $\sigma_\triangleright = do(\alpha_m, \ldots do(\alpha_1, s_*)\ldots)$

$$\mathcal{D} \models \overline{\forall}\,[(\Omega[\phi_1, \ldots, \phi_n] \wedge \wr\Theta) \supset [Exec(\sigma_\triangleright) \wedge \Gamma(\sigma_\triangleright)]\,] \qquad \textbf{(planning)}$$

Projection into the future means: Does a given action sequence $\alpha_1, \ldots, \alpha_m$ achieve the goal if it is performed in the current situation? So it refers to the given future situation $\sigma_\triangleright$ and asks whether a certain property (the goal) will hold in $\sigma_\triangleright$. Projection into the past instead refers to a given past situation $\sigma_\triangleleft$ (S_0 in our robot example) and asks whether a certain property (the goal) held in $\sigma_\triangleleft$. The general form for projection is

$$\mathcal{D} \models \overline{\forall}\,[(\Omega[\phi_1, \ldots, \phi_n] \wedge \wr\Theta) \supset \Gamma(\sigma)] \qquad \textbf{(projection)}$$

Note that plan checking (i. e., checking whether an action sequence is a plan) and reachability testing can be viewed as special cases of projection. If σ is not a bygone situation, i. e., $\mathcal{D} \not\models \overline{\forall}\,[(\Omega[\phi_1, \ldots, \phi_n] \wedge \wr\Theta) \supset \sigma \sqsubset s_*]$, then projection is *hypothetical reasoning*. Since they refer to future situations, plan checking and reachability testing are hypothetical reasoning, too.

If there are no observations (i. e.: $n = 0$) and the current situation is the initial situation (i. e.: Θ simply is $s_* = S_0$) then any occurrence of s_* in Γ can be replaced by S_0 and reachability, planning and projection (into the future) reduce to their standard forms

$$\mathcal{D} \models \exists s\,[Exec(s) \wedge \Gamma(s)] \qquad \text{(reachability)}$$

$$\mathcal{D} \models Exec(\sigma') \wedge \Gamma(\sigma') \qquad \text{(planning)}$$

$$\mathcal{D} \models \Gamma(\sigma') \qquad \text{(projection)}$$

[5] That is in a way: Does $s = \sigma_\triangleright$ yield an instance of the reachability problem?

with $\sigma' = do(\alpha_m, \dots do(\alpha_1, S_0) \dots)$ for variable-free actions $\alpha_1, \dots, \alpha_m$. Note that the standard forms are hypothetical reasoning (since the foundational situation calculus axioms imply $\neg(\sigma' \sqsubset S_0)$).

Of course, the given general forms do not automatically guarantee that the form of the successor state axioms does not matter. For instance, if

$$\Theta' = \phi_{\mathrm{L}}\big[do(enter(\varrho), S_0)\big]$$

then $\overline{\forall}\,[(\Omega[\phi_{\mathrm{R}}] \wedge \wr\Theta') \supset Open(\varrho, S_0)]$ is equivalent to $\Theta' \supset Open(\varrho, S_0)$ w. r. t. both $\mathcal{D}_{True}$ and $\mathcal{D}_{Poss}$, but

$$\mathcal{D}_{True} \wedge \Theta' \models Open(\varrho, S_0)$$
$$\mathcal{D}_{Poss} \wedge \Theta' \not\models Open(\varrho, S_0)$$

(cf. Section 3). The crucial point here is that nothing is known about the executability of $do(enter(\varrho), S_0)$. If $do(enter(\varrho), S_0)$ were claimed to be executable, either directly, e. g., by $Exec(do(enter(\varrho), S_0))$ or $Poss(enter(\varrho), S_0)$, or indirectly, e. g., by $do(enter(\varrho), S_0) = s_1$ or $do(enter(\varrho), S_0) \sqsubseteq s_*$, both $\mathcal{D}_{True}$ and $\mathcal{D}_{Poss}$ would give the same answer. For example, if instead of Θ' we use

$$\Theta'' = \phi_{\mathrm{L}}\big[do(enter(\varrho), S_0)\big] \wedge do(enter(\varrho), S_0) \sqsubseteq s_*$$

then for both $\mathcal{D} = \mathcal{D}_{True}$ and $\mathcal{D} = \mathcal{D}_{Poss}$

$$\mathcal{D} \models \overline{\forall}\,[(\Omega[\phi_{\mathrm{R}}] \wedge \wr\Theta'') \supset Open(\varrho, S_0)]$$

An appropriate definition of *claimed executable* would give us the following theorem:

> If all situations are claimed executable within $\Omega[\dots]$, Θ and Ψ then
>
> $$\mathcal{D}_{True} \models \overline{\forall}\,[(\Omega[\phi_1, \dots, \phi_n] \wedge \wr\Theta) \supset \Psi]$$
> $$\text{iff} \quad \mathcal{D}_{Poss} \models \overline{\forall}\,[(\Omega[\phi_1, \dots, \phi_n] \wedge \wr\Theta) \supset \Psi]\,.$$

It is a topic under investigation to formulate a precise (yet as comprehensive as possible) syntactic criterion for "all situations are claimed executable within ...." However, reasonable formula are likely to comply with this condition as can be seen from the following considerations.

For reachability and planning, s and $\sigma_{\triangleright}$ are explicitly claimed executable by $Exec(s)$ and $Exec(\sigma_{\triangleright})$ respectively.[6] Since reasoning about non-executable actions does not make much sense, for projection into the future Θ should contain $Exec(\sigma_{\triangleright})$, i. e., we assume that the action sequence $\alpha_1, \dots, \alpha_m$ is executable in the current situation. Likewise, for projection into the past it is safe to have $\sigma_{\triangleleft} \sqsubseteq s_*$ contained in Θ as the previous situation $\sigma_{\triangleleft}$ lies before the current situation s_*. Also, if the history assumption Θ mentions previous situations it can contain $\sigma \sqsubseteq s_*$ for each of these situations σ. Yet, Θ may also refer to the future, e. g., if we assume that the current situation is so that after performing some actions $\alpha'_1, \dots, \alpha'_k$ a situation will be reached which has some property ψ, i. e., Θ contains $\psi\big[do(\alpha'_k, \dots do(\alpha'_1, s_*) \dots)\big]$. But if we think that we are able to reach $do(\alpha_m, \dots do(\alpha_1, s_*) \dots)$ then $Exec(do(\alpha_m, \dots do(\alpha_1, s_*) \dots))$ should be contained in Θ, too. Similar considerations can be made for situations in $\Gamma(s)$. However, often s will be the only situations mentioned in $\Gamma(s)$ in which case no problem arises from $\Gamma(s)$.

[6] The fact that $Exec(\sigma_{\triangleright})$ occurs on the right side of the implication does not cause trouble since, e. g., $Exec(do(\alpha', do(\alpha, s_*)))$ is an abbreviation equivalent to $Poss(\alpha, s_*) \wedge Poss(\alpha', do(\alpha, s_*))$. So the first $Poss$-atom claims $do(\alpha, s_*)$ executable (because s_* is executable) and the second $Poss$-atom then claims $do(\alpha', do(\alpha, s_*))$ executable.

6 Related Work

As we already remarked at the beginning of this paper, this work grew out of Iwan's investigations into diagnosing plan execution failures (Iwan 2002). There, as well as in McIlraith's earlier work on diagnosis (McIlraith 1998), observations similar to those used in this paper, play a central role. Although we make no commitment as to how observations come about, they are often a result of sensing actions performed by the agent. Sensing in the framework of the situation calculus is considered, for example, in (De Giacomo & Levesque 1999a; 1999b). McIlraith and Scherl recently addressed the question what sensing tells us (McIlraith & Scherl 2000). However, they are mainly concerned with knowledge, ramifications, and the notion of tests, issues which are orthogonal to those addressed in this paper.

There are also interesting connections between observations and narratives. Just as narratives talk about what *actually* happened in the past, so do observations.[7] Narratives were formalized in the situation calculus in (Pinto 1998; Miller & Shanahan 1994).[8] We remark that the models of time considered in these approaches can easily be added to our framework as well. An interesting approach to modeling narratives in an action language different from the situation calculus is discussed in (Baral, Gelfond, & Provetti 1997). There the language $\mathcal{L}_1$ is proposed, which extends the language $\mathcal{A}$ (Gelfond & Lifschitz 1992; 1993) by adding the notions of actually occurring actions and observations to capture narratives. In the remainder of this section, we take a closer look at this work.

To start with, $\mathcal{L}_1$ is a propositional language, just like $\mathcal{A}$. The constants S_i that are used to denote actual situations (in contrast to hypothetical situations) roughly correspond to our situation variables s_i in $\Omega \wedge \wr\Theta$. The *causal laws* (A causes F if ψ) correspond to effect axioms $\forall s\,[\psi[s] \supset F(do(A, s))]$ in the situation calculus and can be encoded by successor state axioms (cf. (Reiter 1991); the necessary completeness assumption is also part of the semantics of $\mathcal{L}_1$). Observations are expressed by *fluent facts* (ϕ at S_i) which correspond to $\phi[s_i]$; *precedence facts* (S_j precedes S_i) correspond to $s_j \sqsubset s_i$; *occurrence facts* (A occurs_at S_i) translate to $do(A, s_i) \sqsubset s_*$; and *hypotheses* (ϕ after $[A_1, \dots, A_n]$ at S_i) correspond to $\phi\big[do(A_n, \dots do(A_1, s_i) \dots)\big]$. Domain descriptions are collections of laws and facts and can be translated into

[7] Of course, in general observations could be mistaken, an issue we have ignored here altogether.

[8] See also (Baral, McIlraith, & Son 2000), where the connection between diagnosis and narratives is investigated.

successor state axioms (laws) and an observations-plus-history-assumption formula $\Omega \wedge \wr\Theta$ (facts). $\mathcal{L}_1$ has no notion of executability of actions, which means that $\forall s\,[Poss(A, s) \equiv True]$ for all actions A. Therefore, of course, problems regarding $Poss$- or $True$-guarded laws do not arise in $\mathcal{L}_1$ at all. (There are extensions of $\mathcal{L}_1$ containing executability conditions, e. g., in (Baral, McIlraith, & Son 2000) where, applying our terminology, the causal laws are $True$-guarded.) The semantics of $\mathcal{L}_1$ imposes a minimality condition on the occurrence of actions leading to the current situation. It is an interesting question whether and how the same minimality property can be achieved within the situation calculus. Then $\mathcal{L}_1$ may be emulated in the situation calculus using our approach to observations with history assumptions.

7 Summary

Our concern has been to answer the question "What do observations really tell us?" This question arose from examples where an unsophisticated formalization within the situation calculus led to unintended and unintuitive results when drawing conclusions and where the use of either unguarded or $Poss$-guarded successor state axioms unwantedly yields different results. So there was the need for a closer look on how to formalize information provided by observations. We found that the information can (and should) be divided into the *basic information* which only reflects the sequence of observations (up to the current situation) and an *assumption* about the history (including the current situation and possibly assumption about potential future evolutions). With this formalization at hand, we revised the general form of planning and projection (now into the future and into the past) in the presence of observations and argued that unguarded and $Poss$-guarded successor state axioms will behave equivalently.

References

Baral, C.; Gelfond, M.; and Provetti, A. 1997. Representing actions: Laws, observations and hypotheses. *Journal of Logic Programming* 31(1–3).

Baral, C.; McIlraith, S.; and Son, T. 2000. Formulating diagnostic problem solving using an action language with narratives and sensing. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning*.

Burgard, W.; Cremers, A.; Fox, D.; Hähnel, D.; Lakemeyer, G.; Schulz, D.; Steiner, W.; and Thrun, S. 1999. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence* 114(1–2).

De Giacomo, G., and Levesque, H. 1999a. An incremental interpreter for high-level programs with sensing. In Levesque, H. J., and Pirri, F., eds., *Logical Foundation for Cognitive Agents: Contributions in Honor of Ray Reiter*. Springer.

De Giacomo, G., and Levesque, H. 1999b. Projection using regression and sensors. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*.

Gelfond, M., and Lifschitz, V. 1992. Representing actions in extended logic programming. In *Proceedings of the Joint International Conference and Symposium on Logic Programming*.

Gelfond, M., and Lifschitz, V. 1993. Representing action and change by logic programs. *Journal of Logic Programming* 17(2/3&4).

Iwan, G. 2002. History-based diagnosis templates in the framework of the situation calculus. *AI Communications*. To appear.

Levesque, H.; Reiter, R.; Lespérance, Y.; Lin, F.; and Scherl, R. 1997. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming* 31(1–3).

Levesque, H.; Pirri, F.; and Reiter, R. 1998. Foundations for the situation calculus. *Linköping Electronic Articles in Computer and Information Science* 3(018).

McCarthy, J., and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B., and Michie, D., eds., *Machine Intelligence 4*. Edinburgh University Press.

McCarthy, J. 1963. Situations, actions and causal laws. Stanford Artificial Intelligence Project: Memo 2. Reprinted in: M.L. Minsky, editor. *Semantic Information Processing*. MIT Press, 1968.

McIlraith, S., and Scherl, R. 2000. What sensing tells us: Towards a formal theory of testing for dynamical systems. In *Proceedings of the 17th National Conference on Artificial Intelligence*.

McIlraith, S. 1998. Explanatory diagnosis: Conjecturing actions to explain observations. In *Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning*.

Miller, R., and Shanahan, M. 1994. Narratives in the situation calculus. *Journal of Logic and Computation* 4(5).

Pinto, J. 1998. Occurrences and narratives as constraints in the branching structure of the situation calculus. *Journal of Logic and Computation* 8(6).

Reiter, R., and Pirri, F. 1999. Some contributions to the metatheory of the situation calculus. *Journal of the ACM* 46(3).

Reiter, R. 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz, V., ed., *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*. Academic Press.

Reiter, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press.