Submitted to the journal *Artificial Intelligence*.
An earlier version appeared in Proc. AAAI-93.

The Frame Problem and Knowledge-Producing Actions

by
Richard B. Scherl and Hector J. Levesque

Department of Computer Science
University of Toronto
Toronto, Ontario
Canada M5S 3A6
email: scherl@cs.toronto.edu hector@cs.toronto.edu

March 10, 1994

ABSTRACT

This paper proposes a solution to the frame problem for knowledge-producing actions. An example of a knowledge-producing action is a sense operation performed by a robot to determine whether or not there is an object of a particular shape within its grasp. The work is an extension of Reiter's solution to the frame problem for ordinary actions and Moore's work on knowledge and action. The properties of our specification are that knowledge-producing actions do not affect fluents other than the knowledge fluent, and actions that are not knowledge-producing only affect the knowledge fluent as appropriate. In addition, *memory* emerges as a side-effect: if something is known in a certain situation, it remains known at successor situations, unless something relevant has changed. Also, it will be shown that a form of *regression* examined by Reiter for reducing reasoning about future situations to reasoning about the initial situation now also applies to knowledge-producing actions.

1 Introduction

The situation calculus provides a formalism for reasoning about actions and their effects on the world. Axioms are used to specify the prerequisites of actions as well as their effects, that is, the fluents that they change. In general, it is also necessary to provide frame axioms to specify which fluents remain unchanged by the actions. In the worst case this might require an axiom for every combination of action and fluent. Recently, Reiter [1991] (generalizing the work of Haas [1987], Schubert [1990] and Pednault [1989]) has given a set of conditions under which the explicit specification of frame axioms can be avoided. In this paper, we extend his solution to the frame problem to cover *knowledge-producing actions*, that is, actions whose effects are to change a state of knowledge.

A standard example of a knowledge-producing action is that of reading a number on a piece of paper. Consider the problem of dialing the combination of a safe [McCarthy and Hayes, 1969; Moore, 1980; Moore, 1985]. If an agent is at the same place as the safe, and knows the combination of the safe, then he can open the safe by performing the action of dialing that combination. If an agent is at the same place as both the safe and a piece of paper and he knows that the combination of the safe is written on the paper, he can open the safe by first reading the piece of paper, and then dialing that combination. The effect of the read action, then, is to change the knowledge state of the agent, typically to satisfy the prerequisite of a later action. Another example of a knowledge-producing action is performing an experiment to determine whether or not a solution is an acid [Moore, 1985]. Still other examples are a sensing operation performed by a robot to determine the shapes of objects within its grasp [Lespérance and Levesque, 1990; Lespérance, 1991] and the execution of UNIX commands such as **ls** [Etzioni *et al.*, 1992].

To incorporate knowledge-producing actions like these into the situation calculus, it is necessary to treat knowledge as a fluent that can be affected by actions. This is precisely the approach taken by Moore [1980]. What is new here is that the knowledge fluent and knowledge-producing actions are handled in a way that avoids the frame problem: we will be able to prove as a consequence of our specification that knowledge-producing actions do not affect fluents other than the knowledge fluent, and that actions that are not knowledge-producing only affect the knowledge fluent as appropriate. In addition, we will show that *memory* emerges as a side-effect: if something is known in a certain situation, it remains known at successor situations, unless something relevant has changed. We will also show that a form of *regression* examined by Reiter for reducing reasoning about future situations to reasoning about the initial situation now also applies to knowledge-producing actions. This has the desirable effect of allowing us to reduce reasoning about knowledge and action to reasoning about knowledge in the initial situation, where techniques such as those discussed in [Frisch and Scherl, 1991; Scherl, 1992] can be used. Finally, we show that if certain useful properties of knowledge (such as positive introspection) are specified to hold in the initial state, they will continue to hold automatically at all successor situations.

In the next section, we briefly review the situation calculus and Reiter's solution to the frame problem. In the following section, we introduce an epistemic fluent into the situation calculus as an accessibility relation over situations, as done by Moore[1980; 1985]. Our solution to the frame problem for knowledge producing actions, based on this epistemic fluent, is developed and illustrated over the next four sections. In the next to the last section, we consider regression for the situation calculus with knowledge-producing actions. Finally, future work is discussed in the last section.

2 The Situation Calculus and the Frame Problem

The situation calculus (following the presentation in [Reiter, 1991]) is a first-order language for representing dynamically changing worlds in which all of the changes are the result of named *actions* performed by some agent. Terms are used to represent states of the world–i.e. *situations*. If α is an action and s a situation, the result of performing α in s is represented by $do(\alpha, s)$. The constant S_0 is used to denote the initial situation[1]. Relations whose truth values vary from situation to situation, called *fluents*, are denoted by a predicate symbol taking a situation term as the last argument. For example, $Broken(x, s)$ means that object x is broken in situation s. Functions whose

[1]By convention, single lower case letters possibly with subscripts or superscripts, are used to represent variables, strings of lower case letters are used for function symbols, and strings of letters beginning with a capital letter are used for predicate symbols and for constants. When quantifiers are not indicated, the variables are implicitly universally quantified.

denotations vary from situation to situation are called *functional fluents*. They are denoted by a function symbol with an extra argument taking a situation term, as in *phone-number*$(Bill, s)$.

It is assumed that the axiomatizer has provided for each action $\alpha(\vec{x})$, an *action precondition axiom* of the form given in 1, where $\pi_\alpha(s)$ is the formula for $\alpha(\vec{x})$'s action preconditions.

Action Precondition Axiom

$$Poss(\alpha(\vec{x}), s) \equiv \pi_\alpha(\vec{x}, s) \tag{1}$$

An action precondition axiom for the action *drop* is given below.

$$Poss(drop(x), s) \equiv Holding(x, s) \tag{2}$$

Furthermore, the axiomatizer has provided for each fluent F, two *general effect axioms* of the form given in 3 and 4.

General Positive Effect Axiom for Fluent F

$$Poss(a, s) \wedge \gamma_F^+(a, s) \;\rightarrow\; F(do(a, s)) \tag{3}$$

General Negative Effect Axiom for Fluent F

$$Poss(a, s) \wedge \gamma_F^-(a, s) \;\rightarrow\; \neg F(do(a, s)) \tag{4}$$

Here $\gamma_F^+(a, s)$ is a formula describing under what conditions doing the action a in situation s leads the fluent F to become true in the successor situation $do(a, s)$ and similarly $\gamma_F^-(a, s)$ is a formula describing the conditions under which performing action a in situation s results in the fluent F becoming false in situation $do(a, s)$.

For example, 5 is a positive effect axiom for the fluent *Broken*.

$$\begin{gathered} Poss(a, s) \wedge [(a = drop(y) \wedge Fragile(y)) \\ \vee((\exists b) a = explode(b) \wedge Nexto(b, y, s))] \\ \rightarrow Broken(y, do(a, s)) \end{gathered} \tag{5}$$

Sentence 6 is a negative effect axiom for *Broken*.

$$\begin{gathered} Poss(a, s) \wedge a = repair(y) \\ \rightarrow \neg Broken(y, do(a, s)) \end{gathered} \tag{6}$$

It is also necessary to add the *frame axioms* that specify when fluents remain unchanged. The frame problem arises because the number of these frame axioms in the general case is $2 \times \mathcal{A} \times \mathcal{F}$, where $\mathcal{A}$ is the number of actions and $\mathcal{F}$ is the number of fluents.

The solution to the frame problem [Reiter, 1991; Pednault, 1989; Schubert, 1990] rests on a *completeness assumption*. This assumption is that axioms 3 and 4 characterize all the conditions under which action a can lead to a fluent F's becoming true (respectively, false) in the successor situation. Therefore, if action a is possible and F's truth value changes from *false* to *true* as a result of doing a, then $\gamma_F^+(a, s)$ must be *true* and similarly for a change from *true* to *false*. Additionally, *unique name axioms* are added for actions and situations.

Reiter[1991] shows how to derive a set of *successor state axioms* of the form given in 7 from the axioms (positive effect, negative effect and unique name) and the completeness assumption.

Successor State Axiom

$$\begin{array}{l} Poss(a, s) \rightarrow [F(do(a, s)) \equiv \\ \quad \gamma_F^+(a, s) \vee (F(s) \wedge \neg\gamma_F^-(a, s))] \end{array} \tag{7}$$

Similar successor state axioms may be written for functional fluents. A successor state axiom is needed for each fluent F, and an action precondition axiom is needed for each action a. The unique name axioms need not be explicitly represented as their effects can be compiled. Therefore only $\mathcal{F} + \mathcal{A}$ axioms are needed.

From 5 and 6 the following successor state axiom for *Broken* is obtained.

$$\begin{array}{c} Poss(a, s) \rightarrow [Broken(y, do(a, s)) \equiv \\ a = drop(y) \wedge Fragile(y) \vee (\exists b) a = explode(b) \\ \wedge Nexto(b, y, s) \vee \ Broken(y, s) \wedge a \neq repair(y)] \end{array} \tag{8}$$

Now note for example that if $\neg Broken(Obj_1, S_0)$, then it also follows (given the unique name axioms) that $\neg Broken(Obj_1, do(drop(Obj_2), S_0))$.

This discussion has assumed that there are no ramifications, i.e., indirect effects of actions. This can be ensured by prohibiting state constraints, i.e., sentences that specify an interaction between fluents. An example of such a sentence is $\forall s P(s) \equiv Q(s)$. The assumption that there are no state constraints in the axiomatization of the domain will be made throughout this paper. In [Lin and Reiter, 1993], the approach discussed in this section is extended to work with state constraints by compiling the effects of the state constraints into the successor state axioms.

3 An Epistemic Fluent

The approach we take to formalizing knowledge is to adapt the standard possible-world model of knowledge to the situation calculus, as first done by Moore[1980]. Informally, we think of there being a binary accessibility relation over situations, where a situation s' is understood as being accessible from a situation s if as far as the agent knows in situation s, he might be in situation s'. So something is known in s if it is true in every s' accessible from s, and conversely something is not known if it is false in some accessible situation.

To treat knowledge as a fluent, we introduce a binary relation $K(s', s)$, read as "s' is accessible from s" and treat it the same way we would any other fluent. In other words, from the point of view of the situation calculus, the last argument to K is the official situation argument (expressing what is known in situation s), and the first argument is just an auxiliary like the y in $Broken(y, s)$.[2]

We can now introduce the notation $\mathbf{Knows}(P, s)$ (read as P is known in situation s) as an abbreviation for a formula that uses K. For example

$$\mathbf{Knows}(Broken(y), s) \stackrel{\text{def}}{=} \forall s'\ K(s', s) \ \rightarrow Broken(y, s').$$

Note that this notation supplies the appropriate situation argument to the fluent on expansion (and other conventions are certainly possible). For the case of equality literals the convention is to supply the situation argument to each non-variable argument of the equality predicate. For example:

$$\mathbf{Knows}(number(Bill) = number(Mary), s) \stackrel{\text{def}}{=} \forall s'\ K(s', s) \ \rightarrow \\ number(Bill, s') = number(Mary, s').$$

This notation can be generalized inductively to arbitrary formulas so that,

[2]Note that using this convention means that the arguments to K are reversed from their normal modal logic use.

for example

$$\exists x \mathbf{Knows}(\exists y[Nexto(x,y) \wedge \neg Broken(y)], s) \stackrel{\text{def}}{=}$$
$$\exists x \forall s'\ K(s', s)\ \rightarrow \exists y[Nexto(x,y,s') \wedge \neg Broken(y,s')].$$

We will however restrict our attention to knowledge about atomic formulas in both this and the next section.

Turning now to knowledge-producing actions, there are two sorts of actions to consider: actions whose effect is to make known the truth value of some formula, and actions to make known the value of some term. In the first case, we might imagine a $Sense_P$ action for a fluent P, such that after doing a $Sense_P$, the truth value of P is known. We introduce the notation $\mathbf{Kwhether}(P, s)$ as an abbreviation for a formula indicating that the truth value of a fluent P is known.

$$\mathbf{Kwhether}(P, s) \stackrel{\text{def}}{=} \mathbf{Knows}(P, s) \vee \mathbf{Knows}(\neg P, s),$$

It will follow from our specification in the next section that $\mathbf{Kwhether}(P, do(Sense_P, s))$. In the second case, we might imagine an action $Read_t$ for a term t, such that after doing a $Read_t$, the denotation of t is known. For this case, we introduce the notation $\mathbf{Kref}(t, s)$ defined as follows:

$$\mathbf{Kref}(t, s) \stackrel{\text{def}}{=} \exists x \mathbf{Knows}(t = x, s)$$
$$\textit{where } x \textit{ does not appear in } t.$$

It will follow from the specification developed in the next section that $\mathbf{Kref}(t, do(Read_t, s))$. For simplicity, we assume that each type of knowledge-producing action is associated with a characteristic fluent or term in this way.

4 Solving the Frame Problem

The approach being developed here rests on the specification of a successor state axiom for the K relation. For all situations $do(a, s)$, the K relation will be completely determined by the K relation at s and the action a.

For non-knowledge-producing actions (e.g. $drop(x)$), the specification (based on Moore [1980; 1985]) is as follows:

$$Poss(drop(x), s)\ \rightarrow\ [K(s'', do(drop(x), s))\ \equiv \exists s'\,(K(s', s)\ \wedge\ (s'' = do(drop(x), s')))] \tag{9}$$

The idea here is that as far as the agent at world s knows, he could be in any of the worlds s' such that $K(s', s)$. At $do(drop(x), s)$ as far as the agent knows, he can be in any of the worlds $do(drop(x), s')$ for any s' such that $K(s', s)$. So the only change in knowledge that occurs in moving from s to $do(drop(x), s)$ is the knowledge that the action *drop* has been performed.

Now consider the simple case of a knowledge-producing action $Sense_P$ that determines whether or not the fluent P is true (following Moore [1980; 1985]).

$$\begin{aligned} Poss(Sense_P, s) \rightarrow [K(s'', do(Sense_P, s)) \equiv \\ \exists s' (K(s', s) \wedge (s'' = do(Sense_P, s'))) \\ \wedge (P(s) \equiv P(s'))] \end{aligned} \tag{10}$$

Again, as far as the agent at world s knows, he could be in any of the worlds s' such that $K(s', s)$. At $do(Sense_P, s)$ as far as the agent knows, he can be in any of the worlds $do(Sense_P, s')$ for all s' such that $K(s', s)$ and $P(s) \equiv P(s')$. The idea here is that in moving from s to $do(Sense_P, s)$, the agent not only knows that the action $Sense_P$ has been performed (as above), but also the truth value of the predicate P. Observe that the successor state axiom for P guarantees that P is true at $do(Sense_P, s)$ iff P is true at s, and similarly for s' and $do(Sense_P, s')$. Therefore, P has the same truth value in all worlds s'' such that $K(s'', do(Sense_P, s))$, and so $\mathbf{Kwhether}(P, do(Sense_P, s))$ is true.

In the case of a $Read_\tau$ action that makes the denotation of the term τ known, $P(s) \equiv P(s')$ is replaced by $\tau(s) = \tau(s')$. Therefore, τ has the same denotation in all worlds s'' such that $K(s'', do(Read_\tau, s))$, and so $\mathbf{Kref}(\tau, do(Read_\tau, s))$ is true.

In general, there may be many knowledge-producing actions. Associated with each knowledge-producing action α_i is a formula $\varphi_i(s, s')$. In the case of a *Sense* type of action, the formula is of the form $F_i(s) \equiv F_i(s')$, where F_i is a fluent. In the case of a *Read* type of action, the formula is of the form $(\tau_i(s) = \tau_i(s'))$, where τ_i is a situation-dependent term. Assume that there are n knowledge-producing actions $\alpha_1, \ldots, \alpha_n$ and therefore n associated formulas $\varphi_1, \ldots, \varphi_n$. The form of the successor state axiom for K is then as follows:

Successor State Axiom for K

$$
\begin{array}{l}
Poss(a,s) \rightarrow [K(s'', do(a,s)) \equiv \\
\quad \exists s' \, (K(s',s) \wedge (s'' = do(a,s'))) \wedge \\
\quad\quad (((a \neq \alpha_1) \wedge \ldots \wedge (a \neq \alpha_n)) \\
\quad\quad \vee ((a = \alpha_1) \wedge \varphi_1) \\
\quad\quad\quad \vdots \\
\quad\quad \vee ((a = \alpha_n) \wedge \varphi_n)))]
\end{array}
\tag{11}
$$

The relation K at a particular situation $do(a,s)$ is completely determined by the relation at s and the action a.

5 Example

Consider the example of opening a safe whose combination is written on a piece of paper (adapted from Moore[1980], but without the frame axioms). The preconditions for the action $\mathit{dial\text{-}comb}(x)$ (dialing the combination of the safe x and also pulling the handle) are:

$$
\begin{array}{l}
Poss(\mathit{dial\text{-}comb}(x), s) \equiv \\
\quad Safe(x,s) \wedge At(x,s) \wedge \mathbf{Kref}(comb(x), s)
\end{array}
\tag{12}
$$

The fluent $At(x,s)$ holds when the agent is located at the location of object x in situation s. The idea in Sentence 12 is that for the dialing action to be possible, the object being dialed needs to be a safe, the agent needs to be at the safe, and the agent needs to know the combination of the safe. In this paper, we do not discuss the details of the connection between the agent's ability to open the safe by dialing the combination and the agent's knowledge of the combination of the safe[3].

The successor state axioms for the fluents *Open* (i.e. be open), *Safe*, *At*, and the functional fluents *info* and *comb* are as follows:

$$
\begin{array}{l}
Poss(a,s) \rightarrow [Open(x, do(a,s)) \equiv \\
\quad a = \mathit{dial\text{-}comb}(x) \\
\quad\quad \vee (Open\,(x,s) \wedge a \neq lock(x))]
\end{array}
\tag{13}
$$

[3]Indeed, a more refined axiomatization would allow the safe to be opened without knowing its combination, namely by dialing a number that (magically or luckily) happens to be the combination.

$$Poss(a,s) \rightarrow [Safe(x, do(a,s)) \equiv Safe(x,s)] \tag{14}$$

$$Poss(a,s) \rightarrow [At(x, do(a,s)) \equiv a = moveto(x) \vee (At(x,s) \wedge a \neq moveto(y))] \tag{15}$$

$$Poss(a,s) \rightarrow [(info(x, do(a,s)) = y) \equiv (info(x,s) = y)] \tag{16}$$

$$Poss(a,s) \rightarrow [(comb(x, do(a,s)) = y) \equiv (comb(x,s) = y)] \tag{17}$$

The functional fluent $info(x,s)$ is used to denote what is written on paper x.

The axiomatization of the initial state includes $Safe(Sf, S_0)$, $At(Sf, S_0))$, $At(Ppr, S_0)$, and $\mathbf{Knows}(info(Ppr) = comb(Sf), S_0)$. Note that the axiomatization does not entail $Poss(dial\text{-}comb(Sf), S_0)$.

There is a knowledge-producing action $read(x)$, with the following action precondition axiom:

$$Poss(read(x), s) \equiv At(x,s) \tag{18}$$

The successor state axiom for K is as follows:

$$Poss(a,s) \rightarrow K(s'', do(a,s)) \equiv [\exists s'(K(s',s) \wedge (s'' = do(a,s'))) \wedge ((a \neq read(x)) \vee ((a = read(x)) \wedge (info(x,s) = info(x,s')))))] \tag{19}$$

In other words, $read(x)$ is the only knowledge-producing action, and its effect is to make $info(x)$ known.

Note that Sentence 19, along with Sentence 16 and the fact that the *read* action is possible in S_0, ensure that the axiomatization entails

$$\exists x\, \mathbf{Knows}(info(Ppr) = x, do(read(Ppr), S_0)). \tag{20}$$

Since the axiomatization of S_0 includes $\mathbf{Knows}(info(Ppr) = comb(Sf), S_0)$, Sentences 16, 17 and 19 ensure that

$$\mathbf{Knows}(info(Ppr) = comb(Sf), do(read(Ppr), S_0)) \tag{21}$$

also holds. Therefore, the axiomatization entails

$$\mathbf{Kref}(comb(Sf), do(read(Ppr), S_0)) \tag{22}$$

by 20, 21 and the properties of equality.

Since the successor state axioms ensure that a *read* action does not change *At*, *Safe*, *Comb* and *Info*, it is the case that

$$Poss(\textit{dial-comb}(Sf), do(read(Ppr), S_0))$$

and therefore the axiomatization entails that

$$Open(Sf, do(\textit{dial-comb}(Sf), do(read(Ppr), S_0))).$$

In other words, after reading what is on the paper, the safe can be opened by dialing its combination.

6 Correctness of the Solution

This section shows that the proposed solution to the frame problem for the situation calculus with knowledge and knowledge-producing actions does in fact ensure that knowledge only changes as appropriate. In the following, P will be used to represent an arbitrary literal (including equality).

The first theorem shows that knowledge-producing actions do not change the state of the world. The only fluent whose truth value is altered by a knowledge-producing action is K.

Theorem 1 (Knowledge-Producing Effects) *For all situations s, all fluents P (other than K) and knowledge-producing action terms α such that $Poss(\alpha, s)$, if $P(s)$ then $P(do(\alpha, s))$.*

Proof: Immediate from having successor state axioms for each fluent.

□

It is also necessary to show that actions only affect knowledge in the appropriate way. The truth of the following theorem ensures that there are no unwanted increases in knowledge. Informally, nothing is learned about a fluent P by doing action α, as long as α does not affect P and in case α is a knowledge-producing action, α does not provide information about something known to be related to P. More formally, we have:

Theorem 2 (Default Persistence of Ignorance) *For all situations s, if* $\neg\mathbf{Knows}(P, s)$ *and* $Poss(\alpha, s)$ *then* $\neg\mathbf{Knows}(P, do(\alpha, s))$ *as long as*

1. *P is a fluent whose successor state axiom specifies that it is not changed by the action α. More formally*

$$\forall s P(s) \equiv P(do(\alpha, s))$$

2. *It is not the case that α is a knowledge-producing action with the corresponding φ being of the form $F(s) \leftrightarrow F(s')$ such that either*

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

or

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

3. *It is not the case that α is a knowledge-producing action with the corresponding φ being of the form $\tau(s) = \tau(s')$ such that*

$$\exists y \mathbf{Knows}(\tau = y \rightarrow P, s)$$

4. *It is not the case that* $\neg\mathbf{Knows}(Poss(\alpha), s)$ *and*

$$\mathbf{Knows}(Poss(\alpha) \rightarrow P, s)$$

Proof: For any situation s, for $\neg\mathbf{Knows}(P, s)$ to be true, it must be the case that $\forall s' K(s', s) \rightarrow P(s')$ is false. There must be some s' such that $P(s')$ is false. By sentence 11, for all situations s'' such that $K(s'', do(\alpha, s))$, it is the case that $s'' = do(\alpha_i, s')$ for some s' such that $K(s', s)$. The truth of $\neg\mathbf{Knows}(P, do(\alpha, s))$ can be guaranteed as long as (1) $\neg P(do(\alpha, s'))$ and (2) $K(do(\alpha, s'), do(\alpha, s))$. Condition (1) will hold as long as it is not the case that the successor state axiom for P specifies that the effect of a is to change the truth value of P. In other words, if $\forall s P(s) \equiv P(do(\alpha, s))$. Condition (2) can be guaranteed as long as it is not the case that (i) α is a knowledge-producing action of the sense type, with the corresponding φ in the successor state axiom for K of the form $F(s) \leftrightarrow F(s^*)$, such that $\forall s'(K(s', s) \wedge (F(s) \equiv F(s'))) \rightarrow P(s')$, and it is not the case that (ii) α is a knowledge-producing action of the read type with the corresponding φ being of the form $\tau_i(s) = \tau_i(s^*)$, such that $\forall s'(K(s', s) \wedge \tau_i(s) = \tau_i(s')) \rightarrow P(s')$, and (iii) $\neg \forall s' K(s', s) \rightarrow Poss(\alpha, s')$ and $\forall s' (K(s', s) \wedge (Poss(\alpha, s')) \rightarrow P(s')$. Case (i) will not hold as long as it is not the case that $\mathbf{Knows}(F \rightarrow P, s)$ or $\mathbf{Knows}(\neg F \rightarrow P, s)$. Case (ii) will not hold as long as it is not the case that $\exists y \mathbf{Knows}(\tau = y \rightarrow P, s)$. Case (iii) will not hold as long as it is not the case that $\neg\mathbf{Knows}(Poss(\alpha), s)$ and $\mathbf{Knows}(Poss(\alpha) \rightarrow P, s)$. □

Finally, it is a property of this specification that agents never forget. Informally speaking, if the agent knows P at s, then P is also known at $do(\alpha, s)$ as long as the effect of α is not to make P false. More formally, we have:

Theorem 3 (Memory) *For all fluents P and situations s, if* $\mathbf{Knows}(P, s)$ *then* $\mathbf{Knows}(P, do(\alpha, s))$ *as long as P is a fluent whose successor state axiom specifies that it is not changed by the action α. More formally*

$$\forall s P(s) \equiv P(do(\alpha, s))$$

Proof: For $\mathbf{Knows}(P, s)$ to be true, it must be the case that $\forall s K(s', s) \rightarrow P(s')$ is true. By sentence 11, for all states s'' such that $K(s'', do(\alpha, s))$, it is the case that $s'' = do(\alpha, s')$ for some s' such that $K(s', s)$. Since $P(s')$ is true for each of these s', $P(do(\alpha, s'))$ will (by the successor state axiom for P) only be false, if the successor state axiom for P specifies that the effect of α is to make P false. So, $P(do(\alpha, s'))$ can be guaranteed to hold for all $do(\alpha, s')$ such that $K(do(\alpha, s'), do(\alpha, s))$ as long as it is not the case that the successor state axiom for P specifies that the effect of a is to change the truth value of P. In other words, if $\forall s P(s) \equiv P(do(\alpha, s))$. □

Consider again the successor state axiom for *broken* given in sentence 8. If $\mathbf{Knows}(\neg Broken(Obj_1), S_0)$ is true, then

$$\mathbf{Knows}(\neg Broken(Obj_1), do(drop(Obj_2), S_0))$$

must also be true. Also, note that if $\mathbf{Knows}(Fragile(Obj_2), S_0)$ and $\mathbf{Knows}(Poss(drop(Obj_2)), S_0)$ are true, then

$$\mathbf{Knows}(Broken(Obj_2), do(drop(Obj_2), S_0))$$

must also be true.

7 Knowledge of Formulas

Up to this point, all results have been stated in terms of fluents. But both the argument to **Kwhether** and **Knows** can be an arbitrary formula.

In the discussion of *sense* type actions, nothing hinged on the argument to **Kwhether** being a fluent, rather than a formula. Thus the effect of a *sense* action performed by a robot [Lespérance and Levesque, 1990; Lespérance, 1991] may be specified as follows:

$$\begin{aligned}\mathbf{Kwhether}(\exists x\,(Object(x) \wedge \neg Holding(x) \\ \wedge\, Ofshape(x, Shape_1)), do(Sense, S_0))\end{aligned} \tag{23}$$

Now the formula φ_i associated with each knowledge-producing action is of the form $\psi_i(s) \equiv \psi_i(s')$, where ψ_i is a formula.

Also, the arguments to the **Knows** operator may be arbitrary formulas. Now, we may also want nested **Knows** operators. The situation argument of the operator is then understood contextually. If it is not the outermost operator, the situation argument is understood to be the first argument of the immediately dominating K atom. For example, 24 is understood as an abbreviation for 25.

$$\mathbf{Knows}(\mathbf{Knows}(P), S_0) \tag{24}$$

$$\forall s_1\ K(s_1, S_0) \ \rightarrow (\forall s_2\ K(s_2, s_1) \ \rightarrow P(s_2)) \tag{25}$$

By a simple induction on the size of formulas, Theorems 1, 2, and 3 expressed in terms of fluents, can be generalized to formulas as well. So, the solution to the frame problem for knowledge-producing actions is correct for knowledge understood as the knowledge of arbitrary sentences.

The only remaining issue concerns requiring that the **Knows** operator conform to the properties of a particular modal logic. For example, if the logic chosen is $S4$, then we want positive introspection (sentence 26) to be a property of the logic.

$$\mathbf{Knows}(\phi, s) \ \rightarrow \ \mathbf{Knows}(\mathbf{Knows}(\phi), s) \tag{26}$$

Restrictions need to be placed on the K relation so that it correctly models the accessibility relation of a particular modal logic. The problem is to do this in a way that does not interfere with the successor state axioms for K, which must completely specify the K relation for non-initial situations.

The solution is to axiomatize the restrictions for the initial situation and then verify that the restrictions are then obeyed at all situations.

The sort *Init* is used to restrict variables to range only over S_0 and those situations accessible from S_0. It is necessary to stipulate that:

$$Init(S_0)$$
$$\forall s, s_1 Init(s_1) \rightarrow (K(s, s_1) \rightarrow Init(s))$$
$$\forall s, s_1 \neg Init(s_1) \rightarrow (K(s, s_1) \rightarrow \neg Init(s))$$
$$Init(s) \equiv \neg \exists s'(s = do(a, s'))$$

The various restrictions are listed below.[4] The reflexive restriction is always added as we want a modal logic of knowledge. Some subset of the other restrictions are then added.

Reflexive $\forall s_1{:}Init\ K(s_1, s_1)$

Euclidian $\forall s_1{:}Init, s_2{:}Init, s_3{:}Init$
$K(s_2, s_1) \wedge K(s_3, s_1) \rightarrow K(s_3, s_2)$

Symmetric $\forall s_1{:}Init, s_2{:}Init\ K(s_2, s_1) \rightarrow K(s_1, s_2)$

Transitive $\forall s_1{:}Init, s_2{:}Init, s_3{:}Init$
$K(s_2, s_1) \wedge K(s_3, s_2) \rightarrow K(s_3, s_1)$

To model the logic $S4$, for example, one would need to include the axioms for both reflexivity and transitivity.

The next step is to prove that if the K relation over the initial situations satisfies a particular restriction R, that restriction R will also hold over the other situations as well.

Theorem 4 *If the K relation on the set of initial situations is restricted to conform to some subset of the properties of reflexive, symmetric, transitive and euclidian, then the K relation at every situation, resulting from the execution of a sequence of possible actions (as defined by Poss), will satisfy the same set of properties.*

[4] $\forall s{:}Init\,\varphi$ is an abbreviation for $\forall s Init(s) \rightarrow \varphi$

Proof: For each of the restrictions[5], it is only necessary to prove that if the restriction holds at s then it must also hold at $do(a, s)$ for any action a. Since every ground situation term is constructed out of S_0 and a finite number of do function symbols and action terms, and since the restriction holds at S_0, the restriction must hold at every situation[6].

- **Reflexive** It is necessary to show for all s and a that $K(s, s) \rightarrow K(do(a, s), do(a, s))$. If $K(s, s) \rightarrow K(do(a, s), do(a, s))$ is false, then for some s, it must be the case that $K(s, s)$ is true and $K(do(a, s), do(a, s))$ is false. But by the form of the successor state axiom for K (sentence 11) and the fact that for all s, $\tau_i(s) = \tau_i(s)$ and $\psi_i(s) \equiv \psi_i(s)$, and the assumption $K(s, s)$, then $K(do(a, s), do(a, s))$ must be true. Therefore $K(s, s) \rightarrow K(do(a, s), do(a, s))$ cannot be false.

- **Symmetric** It is necessary to show for all s and a that

  $$\forall s' \, (K(s', s) \rightarrow K(s, s')) \rightarrow \forall s'' \, (K(s'', do(a, s)) \rightarrow K(do(a, s), s'')).$$

  Assume $\forall s, s'(K(s', s) \rightarrow K(s, s'))$ is true. For $\forall s, s'' K(s'', do(a, s)) \rightarrow K(do(a, s), s''))$ to be false it must be the case that for some s and s'' $K(s'', do(a, s))$ is true, but $K(do(a, s), s'')$ is false. Note that by the successor state axiom for K (sentence 11), the only way for $K(s'', do(a, s))$ to be true is for s'' to be equal to $do(a, s')$ for some s', for $K(s', s)$ to be true, and depending on a for $\tau_i(s) = \tau_i(s')$ or $\psi_i(s) \equiv \psi_i(s')$ to be true. Then it must be the case that $K(s, s')$ is true by assumption. Then by the successor state axiom for K (sentence 11) and the symmetry of $\equiv$ and $=$, it must also be the case that $K(do(a, s), s'')$ is true. Therefore $\forall s, s'' K(s'', do(a, s)) \rightarrow K(do(a, s), s''))$ cannot be false.

- **Transitive** It is necessary to show for all s and a that

[5] As an example of a restriction that is not guaranteed to hold at every level, even if it does hold over the set of initial situations, consider the condition of being weakly directed:

$$\forall s_1, s_2, s_3 \; K(s_2, s_1) \wedge K(s_3, s_1) \rightarrow \exists s_4 K(s_4, s_2) \wedge K(s_4, s_3)$$

Note that assuming the restriction holds at s, it is not guaranteed to hold at $do(a, s)$, since it is possible that ψ_i has the same truth value in s_1, s_2, and s_3, but not in s_4.

[6] Alternatively, we can appeal to the induction principle for the situation calculus described in [Lin and Reiter, 1993; Reiter, 1993a].

$$\forall s_1, s_2\, (K(s_1, s) \land K(s_2, s_1) \rightarrow K(s_2, s)) \rightarrow \\ \forall s_3, s_4\, (K(s_3, do(a, s)) \land K(s_4, s_3) \rightarrow K(s_4, do(a, s)).$$

Assume $\forall s_1, s_2\ K(s_1, s) \land K(s_2, s_1) \rightarrow K(s_2, s)$ is true for all s. For $\forall s_3 s_4 K(s_3, do(a, s)) \land K(s_4, s_3) \rightarrow K(s_4, do(a, s))$ to be false, it must be the case that for some s, s_3 and s_4, $K(s_3, do(a, s)) \land K(s_4, s_3)$ is true, but $K(s_4, do(a, s))$ is false. Note that by the successor state axiom for K (sentence 11), the only way for $K(s_3, do(a, s)) \land K(s_4, s_3)$ to be true is for is for s_3 to be equal to $do(a, s_3^*)$ for some s_3^* and for $K(s_3^*, s)$ to be true, and for s_4 to be equal to $do(a, s_4^*)$ for some s_4^* and for $K(s_4^*, s_3^*)$ to be true, and depending on a for $\tau_i(s) = \tau_i(s_3^*) \land \tau_i(s_3^*) = \tau_i(s_4^*)$ or $(\psi_i(s) \equiv \psi_i(s_3^*)) \land (\psi_i(s) \equiv \psi_i(s_3^*))$ to be true. Then it must also, be the case that $K(s_4^*, s)$ is true by assumption. Then by the successor state axiom for K (sentence 11) and the transitivity of $=$ and $\equiv$, it must also be the case that $K(do(a, s_4^*), do(a, s))$ is true. Therefore $\forall s_3 s_4 K(s_3, do(a, s)) \land K(s_4, s_3) \rightarrow K(s_4, do(a, s))$ cannot be false.

- **Euclidian** It is necessary to show for all s and a

$$\forall s_1, s_2\, (K(s_1, s) \land K(s_2, s) \rightarrow K(s_2, s_1)) \rightarrow \\ \forall s_3, s_4\, (K(s_3, do(a, s)) \land K(s_4, do(a, s)) \rightarrow K(s_4, s_3)).$$

Assume $\forall s_1, s_2\ K(s_1, s) \land K(s_2, s) \rightarrow K(s_2, s)$ is true for all s. For $\forall s_3 s_4 K(s_3, do(a, s)) \land K(s_4, do(a, s)) \rightarrow K(s_4, s_3)$ to be false, it must be the case that for some s, s_3 and s_4 $K(s_3, do(a, s)) \land K(s_4, do(a, s))$ is true, but $K(s_4, s_3)$ is false. Note that by the successor state axiom for K (sentence 11), the only way for $K(s_3, do(a, s)) \land K(s_4, do(a, s))$ to be true is for is for s_3 to be equal to $do(a, s_3^*)$ for some s_3^* and for $K(s_3^*, s)$ to be true, and for s_4 to be equal to $do(a, s_4^*)$ for some s_4^* and for $K(s_4^*, s)$, and depending on a for $\tau_i(s) = \tau_i(s_3^*) \land \tau_i(s) = \tau_i(s_4^*)$ or $(\psi_i(s) \equiv \psi_i(s_3^*)) \land (\psi_i(s) \equiv \psi_i(s_4^*))$ to be true. Then it must also, be the case that $K(s_4^*, s_3^*)$ is true by assumption. Then by sentence 11 and the transitivity of $=$ and $\equiv$, it must also be the case that $K(do(a, s_4^*), do(a, s_3^*))$ is true. Therefore $\forall s_3 s_4 K(s_3, do(a, s)) \land K(s_4, do(a, s)) \rightarrow K(s_4, s_3)$ cannot be false.

$\square$

The significance of this theorem is that if the K relation at the initial situation is defined as satisfying certain conditions, then the K relation at all situations reachable from the initial situation also satisfies those properties. So, if we decide to use, for example, the logic $S4$ to model knowledge, we can go ahead and stipulate that the K relation at the initial situation is reflexive and transitive. Then we are guaranteed that the relation at all reachable situations will also satisfy those properties and our model of knowledge will remain $S4$, without danger of conflicting with the successor state axiom.

8 Reasoning

Given the representation of actions and their effects, we would like to have a method for addressing the *projection problem* [Reiter, 1992]. This is the question of determining whether or not some sentence G is true in the situation resulting from the execution of an action sequence $\alpha_1, \ldots, \alpha_n$ of ground action terms. This question is represented as the query

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

where $\mathcal{F}$ is the axiomatization of actions and their effects, and the initial situation. The expression $do([a_1, \ldots, a_n], s)$ is a compact notation for the situation term

$$do(a_n, do(a_{n-1}, \ldots, do(a_1, s) \ldots)),$$

which denotes the state resulting from performing the action a_1, followed by $a_2, \ldots$, followed by a_n, beginning in situation s.

One method of answering the query is to translate the axiomatization of the initial situation and the query G (both of which contain the modal operator **Knows**) into first-order logic using the well known method implicit in much of the discussion here and used by Moore [1980; 1985]. Then any first-order theorem proving method can be used to query a particular axiomatization. It would be much more efficient to use a method designed for performing automated deduction in modal logics, but sentence 11 has no such representation in a modal logic of knowledge[7].

[7] In fact, the results reported in [Frisch and Scherl, 1991; Scherl, 1992] indicate that even modal deduction methods that can be understood as reasoning with the K literals representing the accessibility relation, cannot be easily modified to handle a sentence like 11.

Here, regression operators are utilized to address the projection problem. Regression is also used to determine whether or not the sequence of actions $\alpha_1, \ldots, \alpha_n$ is an executable sequence of actions resulting in a *legal* situation.

8.1 Regression

Reiter [Reiter, 1991] develops a form of *regression* to reduce reasoning about future situations to reasoning about the initial situation. In this section, a regression operator is developed for knowledge-producing actions and applied to the problem of determining whether or not a particular plan satisfies a particular property. So given a plan, expressed as a ground situation term (i.e. a term built on S_0 with the function *do* and ground action terms) s_{gr}, the question is whether the axiomatization of the domain $\mathcal{F}$ entails $G(s_{gr})$ where G is an arbitrary sentence including modal operators. Under these circumstances, the successor state axioms (including 11) are only used to regress the formula $G(s_{gr})$. The result of the regression is a formula in ordinary modal logic—i.e. a formula without action terms and where the only situation term is S_0. Then an ordinary modal theorem proving method (e.g. that developed in [Frisch and Scherl, 1991; Scherl, 1992]) may be used to determine whether or not the regressed formula holds. In what follows, it is assumed that the formulas do not use the fluent K except as abbreviated by **Knows**.

The regression operator $\mathcal{R}$ is defined relative to a set of successor state axioms Θ. The first four parts of the definition of the regression operator $\mathcal{R}_\Theta$ concern ordinary (i.e. not knowledge-producing) actions [Reiter, 1991; Pednault, 1989].

i. When A is a non-fluent atom, including equality atoms, and atoms with the predicate symbol *Poss*, or when A is a fluent atom or **Knows** operator, whose situation argument is the situation constant S_0, $\mathcal{R}_\Theta[A] = A$.

ii. There are two cases here. Case **a** covers ordinary (non-functional) fluents. Case **b** covers equality literals with at least one functional fluent as an argument.

a When F is a fluent (other than K) whose successor state axiom in Θ is

$$Poss(a, s) \rightarrow [F(x_1, \ldots, x_n, do(a, s)) \equiv \Phi_F]$$

then

$$\mathcal{R}_\Theta[F(t_1, \ldots, t_n, do(\alpha, \sigma))] = \Phi_F|^{x_1,\ldots,x_n,a,s}_{t_1,\ldots,t_n,\alpha,\sigma}$$

b When the item to be regressed is an equality literal with an argument being the functional fluent f whose successor state axiom in Θ is

$$Poss(a, s) \rightarrow [f(x_1, \ldots, x_n, do(a, s)) = y \equiv \Phi_f]$$

then

$$\mathcal{R}_\Theta[f(t_1, \ldots, t_n, do(\alpha, \sigma)) = t_{n+1}] = \Phi_f|^{x_1,\ldots,x_n,a,s,y}_{t_1,\ldots,t_n,\alpha,\sigma,t_{n+1}}$$

and

$$\mathcal{R}_\Theta[t_{n+1} = f(t_1, \ldots, t_n, do(\alpha, \sigma))] = \Phi_f|^{x_1,\ldots,x_n,a,s,y}_{t_1,\ldots,t_n,\alpha,\sigma,t_{n+1}}$$

iii. Whenever W is a formula, $\mathcal{R}_\Theta[\neg W] = \neg\mathcal{R}_\Theta[W]$, $\mathcal{R}_\Theta[(\forall v)W] = (\forall v)\mathcal{R}_\Theta[W]$, $\mathcal{R}_\Theta[(\exists v)W_1] = (\exists v)\mathcal{R}_\Theta[W_1]$.

iv. Whenever W_1 and W_2 are formulas, $\mathcal{R}_\Theta[W_1 \wedge W_2] = \mathcal{R}_\Theta[W_1] \wedge \mathcal{R}_\Theta[W_1]$, $\mathcal{R}_\Theta[W_1 \vee W_2] = \mathcal{R}_\Theta[W_1] \vee \mathcal{R}_\Theta[W_1]$, $\mathcal{R}_\Theta[W_1 \rightarrow W_2] = \mathcal{R}_\Theta[W_1] \rightarrow \mathcal{R}_\Theta[W_1]$.

Additional steps are needed to extend the regression operator to knowledge-producing actions. Two definitions are needed for the specification to follow. When φ is an arbitrary sentence and s a situation term, then $\varphi[s]$ is the sentence that results from adding an extra argument to every fluent of φ and inserting s into that argument position. The reverse operation φ^{-1} is the result of removing the last argument position from all the fluents in φ.

Step **v** covers the case of regressing a non-knowledge producing action through the **Knows** operator. Step **vi** covers the cases of regressing a knowledge producing action of the *sense* type through the **Knows** operator and step **vii** does the same for *read* type actions. In the definitions below, s' is a new situation variable.

v. Whenever a is not a knowledge-producing action,
$\mathcal{R}_\Theta[\mathbf{Knows}(W, do(a, s))] = \mathbf{Knows}(\mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s)$.

vi. $\mathcal{R}_\Theta[\mathbf{Knows}(W, do(sense_i, s))] =$
$((\psi_i(s) \rightarrow \mathbf{Knows}(\psi_i \rightarrow \mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s)) \wedge$
$(\neg\psi_i(s) \rightarrow \mathbf{Knows}(\neg\psi_i \rightarrow \mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s))$

vii. $\mathcal{R}_\Theta[\mathbf{Knows}(W, do(read_i, s))] =$
$\quad \exists y \; \mathbf{Knows}(t_i = y \;\rightarrow\; \mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s)$

In the following theorem, $\mathcal{F}$ is the axiomatization of the domain including $\mathcal{F}_{ss}$, the successor state axioms. The notation $\mathcal{R}^*_\Theta(\varphi)$ is used to indicate that the regression operator is applied repeatedly until further applications leave the formula unchanged.

Theorem 5 *For any ground situation term s_{gr}*

$$\mathcal{F} \models G(s_{gr}) \text{ iff } \mathcal{F} - \mathcal{F}_{ss} \models \mathcal{R}^*_\Theta[G(s_{gr})]$$

Proof: It suffices to show that the process of regression preserves logical equivalence given the axiomatization $\mathcal{F}$.

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

This is done by showing that each step preserves logical equivalence. The process must terminate as every step removes the outer *do* from the situation terms and the number of *do* function symbols making up any such term is finite. As each step preserves equivalence, the whole process results in an equivalent formula. Since after regression terminates the sentence G does not contain any action terms (i.e. the only situation term is S_0) the successor state axioms are no longer needed to determine whether or not $\mathcal{F} \models \mathcal{R}^*_\Theta[G(s_{gr})]$.

To prove that each step preserves logical equivalence, it suffices to show the following:

$$\mathcal{F} \models \forall a, s\, G(do(a, s)) \;\equiv\; \mathcal{R}_\Theta[G(do(a, s))].$$

The proof is by induction on the size of the sentence G.

The proofs of equivalence for the first four steps of the regression operator $\mathcal{R}_\Theta$ are relatively simple.

i Immediate
ii. By the form of the successor state axioms.
iii. Follows from the definition of negation and the quantifiers, and the inductive hypothesis.
iv. From the definition of the connectives and the inductive hypothesis.

Before going on to **v**, **vi**, and **vii**, note that the successor-state axiom for K entails a set of sentences of the forms indicated below. For each knowledge-producing action a, there is a sentence of the following form:

$$\begin{array}{l}\forall s, s',\ K(do(a, s')), do(a, s)) \equiv \\ \quad K(s', s) \wedge \varphi_i)\end{array} \tag{27}$$

For each non-knowledge-producing action a, there is a sentence of the following form:

$$\forall s, s',\ K(do(a, s')), do(a, s)) \equiv K(s', s) \tag{28}$$

For every action a, there is a sentence of the following form:

$$\forall s''(K(s'', do(a, s)) \rightarrow \exists s'(s'' = do(a, s'))) \tag{29}$$

The proofs of equivalence for steps **v**, **vi**, and **vii** are as follows:

v.

$$\mathbf{Knows}(W, do(a, s))$$

by definition of **Knows**

$$\forall s'' K(s'', do(a, s)) \rightarrow W[s'']$$

by successor state axiom for K (in particular sentence 29) and specialization

$$\forall s' K(do(a, s'), do(a, s)) \rightarrow W[do(a, s')]$$

by successor state axiom for K (in particular sentence 28), and the inductive hypothesis

$$\forall s' K(s', s) \rightarrow \mathcal{R}_\Theta[W[do(a, s')]]$$

by definition of **Knows**

$$\mathbf{Knows}(\mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s)$$

vi.

$$\mathbf{Knows}(W, do(sense_i, s))$$

by definition of **Knows**

$$\forall s'' K(s'', do(sense_i, s)) \rightarrow W[s'']$$

by the successor state axiom for K (in particular sentence 29)

$$\forall s' K(do(sense_i, s'), do(sense_i, s)) \rightarrow W[do(sense_i, s')]$$

by the successor state axiom for K, and the inductive hypothesis

$$\forall s'(K(s', s) \wedge (\psi_i(s) \equiv \psi_i(s'))) \rightarrow \mathcal{R}_\Theta[W[do(a, s')]]$$

by the definition of the connectives

$$\begin{aligned}\forall s'(\neg K(s', s) \vee \\ &((\psi_i(s') \vee \psi_i(s)) \wedge \\ &(\neg\psi_i(s) \vee \psi_i(s)) \wedge \\ &(\neg\psi_i(s') \vee \psi_i(s')) \wedge \\ &(\neg\psi_i(s) \vee \neg\psi_i(s'))) \wedge \\ &\quad \vee \mathcal{R}_\Theta[W[do(a, s')]])\end{aligned}$$

by Boolean logic

$$\begin{gathered}\forall s'((\neg K(s', s) \vee \psi_i(s') \vee \psi_i(s) \vee \mathcal{R}_\Theta[W[do(a, s')]]) \wedge \\ (\neg K(s', s) \vee \neg\psi_i(s) \vee \neg\psi_i(s') \vee \mathcal{R}_\Theta[W[do(a, s')]]))\end{gathered}$$

by definition of $\forall$ and $\vee$

$$\begin{gathered}(\psi_i(s) \vee \forall s'(\neg K(s', s) \vee \psi_i(s') \vee \mathcal{R}_\Theta[W[do(a, s')]])) \wedge \\ (\neg\psi_i(s) \vee \forall s'(\neg K(s', s) \vee \neg\psi_i(s') \vee \mathcal{R}_\Theta[W[do(a, s')]]))\end{gathered}$$

by definition of **Knows**, and $\rightarrow$

$$\begin{gathered}(\neg\psi_i(s) \rightarrow \mathbf{Knows}(\neg\psi_i \rightarrow \mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s)) \wedge \\ (\psi_i(s) \rightarrow \mathbf{Knows}(\psi_i \rightarrow \mathcal{R}_\Theta[W[do(a, s')]]^{-1}, s))\end{gathered}$$

vii.

$$\mathbf{Knows}(W, do(read_i, s))$$

by definition of **Knows**

$$\forall s'' K(s'', do(read_i, s)) \rightarrow W[s'']$$

by the successor state axiom for K (in particular sentence 29)

$$\forall s' K(do(read_i, s'), do(read_i, s)) \rightarrow W[do(read_i, s')]$$

by the successor state axiom for K, and the inductive hypothesis

$$\forall s'(K(s',s) \wedge (\tau_i(s) = \tau_i(s'))) \rightarrow \mathcal{R}_\Theta[W[do(a,s')]]$$

by the definition of the connectives

$$\forall s' \neg K(s',s) \vee \tau_i(s') \neq \tau_i(s) \vee \mathcal{R}_\Theta[W[do(a,s')]]$$

by definition of $\exists$, reflexivity of K

$$\exists y \forall s' \neg K(s',s) \vee \tau_i(s') \neq y \vee \mathcal{R}_\Theta[W[do(a,s')]]$$

by definition of **Knows**, and $\rightarrow$

$$\exists y \mathbf{Knows}(\tau_i = y \rightarrow \mathcal{R}_\Theta[W[do(a,s')]]^{-1}, s)$$

□

The result means that to test if some sentence G is true after executing a plan, it is only necessary to first regress $G(s_{gr})$, where s_{gr} is the plan expressed as a situation term, using the successor state axioms. This is accomplished by repeatedly passing the regression operator through the formula until the only situation term is S_0. Then the successor state axioms (including 11) are no longer needed. At that point an ordinary modal logic theorem proving method can be utilized to perform the test to determine whether or not $\mathcal{F} - \mathcal{F}_{ss} \models \mathcal{R}^*_\Theta[G(s_{gr})]$.

Consider the following example adapted from [Moore, 1985] (but without the frame axioms). The task is to show that after an agent performs a litmus paper test on an acidic solution, the agent will know that the solution is acidic. The litmus paper turns red if and only if the solution is acidic. The axiomatization includes $Acid(S_0)$. The actions are $Test_1$ and $Sense_R$. As the action preconditions are all $True$, the predicate $Poss$ is ignored in the presentation here. The successor state axioms for Red and $Acid$ are given below:

$$\begin{array}{l} Red(do(a,s)) \equiv \\ \quad (Acid(s) \wedge a = Test_1) \vee (Red(s) \wedge a \neq Test_1) \end{array} \tag{30}$$

$$Acid(do(a,s)) \equiv Acid(s) \tag{31}$$

The instance of the successor state axiom (11) for the K relation is:

$$\begin{array}{l} \forall a, s, s'',\ K(s'', do(a,s)) \equiv \\ \quad \exists s'(K(s',s) \wedge (s'' = do(a,s'))) \wedge \\ \qquad ((a \neq Sense_R) \vee ((a = Sense_R) \\ \qquad \wedge (Red(s) \equiv Red(s')))) \end{array} \tag{32}$$

The formula to be initially regressed is

$$\mathbf{Knows}(Acid, do(Sense_R, do(Test_1, S_0))) \tag{33}$$

Step **vi** of the definition of $\mathcal{R}$ is used with 33 to yield 34.

$$\begin{aligned}&(Red(do(Test_1, S_0)) \rightarrow \\ &\quad \mathbf{Knows}(Red \rightarrow Acid, do(Test_1, S_0))) \wedge \\ &(\neg Red(do(Test_1, S_0)) \rightarrow \\ &\quad \mathbf{Knows}(\neg Red \rightarrow Acid, do(Test_1, S_0)))\end{aligned} \tag{34}$$

Note that $Red(do(Test_1, S_0))$ regresses to $Acid(S_0)$ and $\neg Red(do(Test_1, S_0))$ regresses to $\neg Acid(S_0)$, by step **ii** with Sentence 30, and some simplification. Additionally, $\mathbf{Knows}(Red \rightarrow Acid, do(Test_1, S_0)))$ regresses to $\mathbf{Knows}(Acid \rightarrow Acid, S_0))$, and $\mathbf{Knows}(\neg Red \rightarrow Acid, do(Test_1, S_0)))$ regresses to $\mathbf{Knows}(\neg Acid \rightarrow Acid, S_0))$, by steps **ii**, **iii**, **iv**, and **v** with Sentences 30 and 31, and some simplification. Therefore, the result of regressing 34 is 35.

$$\begin{aligned}&(Acid(S_0) \rightarrow \mathbf{Knows}(Acid \rightarrow Acid, S_0)) \wedge \\ &\quad (\neg Acid(S_0) \rightarrow \mathbf{Knows}(\neg Acid \rightarrow Acid, S_0))\end{aligned} \tag{35}$$

Sentence 35 is clearly entailed by $Acid(S_0)$ and so 33 is entailed by the original theory. Note that 35 can be rewritten as a sentence in an ordinary modal logic because the only situation term is S_0.

Now, consider the safe opening example given earlier. We wish to prove that

$$Open(Sf, do(dial\text{-}comb(Sf), do(read(Ppr), S_0))). \tag{36}$$

Sentence 36 is regressed to 37 by step **ii** with Sentence 13.

$$\begin{aligned}&dial\text{-}comb(Sf) = dial\text{-}comb(Sf) \\ &\quad \vee (Open\ (Sf, do(read(Ppr), S_0)) \wedge dial\text{-}comb(Sf) \neq lock(Sf))]\end{aligned} \tag{37}$$

At this point it can be seen that the regressed formula is entailed by the axioms since one of the disjuncts $dial\text{-}comb(Sf) = dial\text{-}comb(Sf)$ is clearly entailed by the theory. The crucial part of this particular problem lies in determining that the *dial-comb* action is possible in $do(read(Ppr), S_0)$. This is the topic of the next section.

8.2 Legality Testing

In order to determine whether or not a particular sequence of actions results in a state in which a particular sentence is true, it is also necessary to show that each step in the sequence of actions is executable or possible. Following Reiter[1993b], we define a *legal action sequence.* Consider a sequence of ground action terms $\alpha_1, \ldots, \alpha_n$. This sequence is *legal* if and only if, beginning in the initial situation S_0, each action α_i in the sequence is possible in the state resulting from performing the actions $\alpha_i, \ldots, \alpha_{i-1}$. The situation term

$$do(\alpha_m, do(\alpha_{m-1}, \ldots, do(\alpha_1, s)\ldots))$$

is a *legal situation* if and only if $[\alpha_1, \ldots, \alpha_m]$ is a legal action sequence.

We need a method of testing the legality of action sequences. Suppose that there are n actions and the action precondition axioms are as follows:

$$\begin{gathered}(\forall \vec{x_1})Poss(Act_1(\vec{x_1}), s) \equiv \Pi_{Act_1}(\vec{x}_1, s),\\ \vdots \\ (\forall \vec{x_n})Poss(Act_n(\vec{x_n}), s) \equiv \Pi_{Act_n}(\vec{x}_n, s)\end{gathered}$$

Given a particular ground action term $\alpha_i(\vec{t})$, we need to pick out the proper instance of the corresponding Π_{Act_j}. The notation $\Pi_{(\alpha_i)}$ is used to indicate the $\Pi_{Act_j}[\vec{x} \mapsto \vec{t}]$ such that $\alpha_i = Act_j(\vec{x})[\vec{x} \mapsto \vec{t}]$. Then the sequence $[\alpha_1, \ldots, \alpha_m]$, where each α_i is a ground action term, is a legal action sequence if and only if

$$\begin{aligned}\mathcal{F} \models \Pi_{(\alpha_1)}[s \mapsto S_0] \ \wedge \ \Pi_{(\alpha_2)}[s \mapsto do(\alpha_1, S_0)] \ \wedge \ \ldots \\ \wedge \ \Pi_{(\alpha_m)}[s \mapsto do([\alpha_1, \ldots, \alpha_{m-1}], S_0)]\end{aligned} \tag{38}$$

by the definition of a legal action sequence.

By the correctness of regression, we can conclude that the sequence $[\alpha_1, \ldots, \alpha_n]$ is a legal action sequence if and only if

$$\begin{aligned}\mathcal{F} - \mathcal{F}_{ss} \models \Pi_{(\alpha_1)}[s \mapsto S_0] \ \wedge \ \mathcal{R}^*_\Theta[\Pi_{(\alpha_2)}[s \mapsto do(\alpha_1, S_0)]] \ \wedge \ldots \\ \wedge \ \mathcal{R}^*_\Theta[\Pi_{(\alpha_m)}[s \mapsto do([\alpha_1, \ldots, \alpha_{m-1}], S_0)]]\end{aligned}$$

Returning to the safe opening example, consider testing the legality of the following situation:

$$do(\textit{dial-comb}(Sf), do(read(Ppr), S_0)) \tag{39}$$

This is answered by determining whether or not the axiomatization entails Sentence 40, which is formed on the basis of 38, with the preconditions of *dial-comb* and *read* as axiomatized with Sentences 12 and 18.

$$\begin{array}{l} At(Ppr, S_0) \wedge \\ Safe(Sf, do(read(Ppr), S_0)) \ \wedge \ At(Sf, do(read(Ppr), S_0)) \wedge \\ \qquad\qquad \mathbf{Kref}(comb(Sf), do(read(Ppr), S_0)) \end{array} \tag{40}$$

Note that $Safe(Sf, do(read(Ppr), S_0))$ regresses to $Safe(Sf, S_0)$ by step **ii** of the regression operator and Sentence 14. Also, $At(Sf, do(read(Ppr), S_0))$ regresses to

$$(read(Ppr) = moveto(Sf) \ \vee \ (At(Sf, S_0) \ \wedge \ read(Ppr) \neq moveto(Sf)))$$

by step **ii** and Sentence 15. Furthermore, $\mathbf{Kref}(comb(Sf), do(read(Ppr), S_0))$ is an abbreviation for $\exists x \ \mathbf{Knows}(comb(Sf) = x, do(read(Ppr), S_0))$, which regresses to

$$\exists x \ \exists y \mathbf{Knows}(info(Ppr, S_0) = y \ \rightarrow \ comb(sf) = x, S_0))$$

by step **vii** and Sentence 19. Therefore Sentence 40 regresses to the following sentence:

$$\begin{array}{l} At(Ppr, S_0) \wedge \ Safe(Sf, S_0) \wedge \\ \quad (read(Ppr) = moveto(Sf) \vee (At(Sf, S_0) \ \wedge \ read(Ppr) \neq moveto(Sf))) \wedge \\ \qquad \exists x \ \exists y \mathbf{Knows}(info(Ppr, S_0) = y \ \rightarrow \ comb(sf) = x, S_0)) \end{array} \tag{41}$$

It can be readily determined that the axiomatization of the initial situation entails this sentence. The axiomatization includes $At(Ppr, S_0)$, $Safe(Sf, S_0)$, and $At(Sf, S_0)$. Since the axiomatization also includes $\mathbf{Knows}(info(Ppr) = Comb(Sf), S_0)$, by the properties of equality

$$\exists x \ \exists y \mathbf{Knows}(info(Ppr) = y \ \rightarrow \ comb(sf) = x, S_0)$$

is entailed as well.

9 Conclusion

This paper provides a solution to the frame problem for knowledge-producing actions. As long as the conditions needed for Reiter's solution for ordinary actions can be met, the work presented here provides a solution for knowledge-producing actions as well.

In terms of future work, we are extending the work discussed here so that the knowledge prerequisites and effects of actions can be *indexical* rather than objective knowledge. Following [Lespérance and Levesque, 1990; Lespérance, 1991], this will be done by making situations a composite of agents, times and worlds.

Also, the consideration of logics of *belief* is a topic for future research. The results presented in this paper are limited to logics of knowledge—logics with a possible world semantics in which the accessibility relation is reflexive. Note that in the case of a knowledge-producing action a that causes P to be known at $do(a,s)$, there must be a situation s' such that $K(s',s)$, and $P(s')$. But in the case of a belief-producing action, there is no guarantee that such a situation s' exist. This is why the results do not directly extend to modal logics without a reflexive accessibility relation.

10 Acknowledgments

Our work on the situation calculus and the frame problem has been carried out in collaboration with Yves Lespérance, Fangzhen Lin, and Ray Reiter. We thank them for many useful discussions, and for comments on earlier versions of this paper. Additionally, we thank Joe Halpern, Sheila McIlraith, Bill Millar, and Steven Shapiro for helpful comments on earlier versions of this paper. This research was funded in part by the National Sciences and Engineering Research Council of Canada, and the Institute for Robotics and Intelligent Systems.

References

[Etzioni *et al.*, 1992] Etzioni, Oren; Hanks, Steve; Weld, Daniel; Draper, Denise; Lesh, Neal; and Williamson, Mike 1992. An approach to planning with incomplete information. In Nebel, Bernhard; Rich, Charles; and Swartout, William, editors 1992, *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference*, Cambridge, Massachusetts. 115–125.

[Frisch and Scherl, 1991] Frisch, Alan and Scherl, Richard 1991. A general framework for modal deduction. In Allen, J.A.; Fikes, R.; and Sandewall, E., editors 1991, *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, San Mateo,CA : Morgan Kaufmann. 196–207.

[Haas, 1987] Haas, A. R. 1987. The case for domain-specific frame axioms. In Brown, F. M., editor 1987, *The Frame Problem in Artificial Intelligence. Proceedings of the 1987 Workshop.* Morgan Kaufmann Publishers, Inc., San Mateo, California. 343–348.

[Lespérance and Levesque, 1990] Lespérance, Yves and Levesque, Hector J. 1990. Indexical knowledge in robot plans. In *Proceedings Eighth National Conference On Artificial Intelligence.* 1030–1037.

[Lespérance, 1991] Lespérance, Yves 1991. *A Formal Theory of Indexical Knowledge and Action.* Ph.D. Dissertation, University of Toronto.

[Lin and Reiter, 1993] Lin, Fangzhen and Reiter, Raymond 1993. State constraints revisited. Presented at the Second Symposium on Logical Formalizations of Commonsense Reasoning.

[McCarthy and Hayes, 1969] McCarthy, J. and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B. and Michie, D., editors 1969, *Machine Intelligence 4.* Edinburgh University Press, Edinburgh, UK. 463–502.

[Moore, 1980] Moore, R.C. 1980. Reasoning about knowledge and action. Technical Note 191, SRI International.

[Moore, 1985] Moore, R.C. 1985. A formal theory of knowledge and action. In Hobbs, J.R. and Moore, R.C., editors 1985, *Formal Theories of the Commonsense World.* Ablex, Norwood, NJ. 319–358.

[Pednault, 1989] Pednault, E.P.D. 1989. ADL: exploring the middle ground between STRIPS and the situation calculus. In Brachman, R.J.; Levesque, H.; and Reiter, R., editors 1989, *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning.* Morgan Kaufmann Publishers, Inc., San Mateo, California. 324–332.

[Reiter, 1991] Reiter, Raymond 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz, Vladimir, editor 1991, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*. Academic Press, San Diego, CA. 359–380.

[Reiter, 1992] Reiter, Raymond 1992. The projection problem in the situation calculus: a soundness and completeness result, with an application to database updates. In *Proceedings of the First International Conference on AI Planing Systems*, College Park, Maryland. 198–203.

[Reiter, 1993a] Reiter, Raymond 1993a. Proving properties of states in the situation calculus. *Artificial Intelligence*. to appear.

[Reiter, 1993b] Reiter, Raymond 1993b. Representing and reasoning about actions. Course Notes for Computer Science 2532: Logic and Artificial Intelligence.

[Scherl, 1992] Scherl, Richard 1992. *A Constraint Logic Approach To Automated Modal Deduction*. Ph.D. Dissertation, University of Illinois.

[Schubert, 1990] Schubert, L.K. 1990. Monotonic solution of the frame problem in the situation calculus: an efficient method for worlds with fully specified actions. In Kyberg, H. E.; Loui, R.P.; and Carlson, G.N., editors 1990, *Knowledge Representation and Defeasible Reasoning*. Kluwer Academic Press, Boston, Mass. 23–67.