The Frame Problem and Knowledge-Producing Actions

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

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.

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

*National Sciences and Engineering Research Council of Canada International Postdoctoral Fellow
†Fellow of the Canadian Institute for Advanced Research

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.

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.

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

Action Precondition Axiom

$$Poss(a(\vec{x}), s) \equiv \pi_a(\vec{x}, s) \tag{1}$$

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*.

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

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

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

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 R becoming true (respectively, false) in the successor situation. Therefore, if action a is possible and R's truth value changes from *false* to *true* as a result of doing a, then $\gamma_R^+(a, s)$ 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

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

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.

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

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

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 $\forall s P(s) \leftrightarrow Q(s)$. 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.

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).

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

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

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:

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

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.

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:

$$Poss(drop(x), s) \rightarrow K(s'', do(drop(x), s)) \equiv \exists s' (K(s', s) \wedge (s'' = do(drop(x), s'))) \quad (9)$$

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]).

$$Poss(Sense_P, s) \rightarrow K(s'', do(Sense_P, s)) \equiv [\exists s' (K(s', s) \wedge (s'' = do(Sense_P, s'))) \wedge (P(s) \equiv P(s'))] \quad (10)$$

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 **Kwhether**$(P, do(Sense_P, s))$ is true.

In the case of a $Read_t$ action that makes the denotation of the term t known, $P(s) \leftrightarrow P(s')$ is replaced by $t(s) = t(s')$. Therefore, t has the same denotation in all worlds s'' such that $K(s'', do(Read_t, s))$, and so **Kref**$(t, do(Read_t, s))$ is true.

In general, there may be many knowledge-producing actions. Each knowledge-producing action A_i will have associated with it 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 $(t_i(s) = t_i(s'))$, where t_i is a situation-dependent term. Assume that there are n knowledge-producing actions $A_1, \ldots, A_n$ 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

$$\begin{aligned} Poss(a, s) \rightarrow\ & K(s'', do(a, s)) \equiv \\ & [\exists s'\ (K(s', s) \wedge (s'' = do(a, s'))) \wedge \\ & \quad (((a \neq A_1) \wedge \ldots \wedge (a \neq A_n)) \\ & \quad \vee ((a = A_1) \wedge \varphi_1) \\ & \quad \vdots \\ & \quad \vee ((a = A_n) \wedge \varphi_n)))] \end{aligned} \tag{11}$$

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

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 successor state axiom for the fluent *Open* (i.e. be open) is:

$$\begin{aligned} Poss(a, s) \rightarrow\ & [Open(y, do(a, s)) \equiv \\ & \quad (\exists x\, a = dial(x, y) \wedge x = comb(y, s)) \\ & \quad \vee (Open\ (y, s) \wedge a \neq lock(y))] \end{aligned} \tag{12}$$

The preconditions for dial (actually dialing and pulling the handle) are:

$$\begin{aligned} & Poss(dial(x, y), s) \equiv Safe(y, s) \wedge \\ & \quad At(y, s) \wedge (\forall s' K(s', s) \rightarrow x = comb(y, s')) \end{aligned} \tag{13}$$

The idea here is that 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. The axiomatization of the initial state includes $Safe(Sf_1, S_0)$, $At(Sf_1, S_0))$, $At(Ppr, S_0)$, and $info(Ppr, S_0) = comb(Sf_1, S_0)$. Note that $\neg\exists x Poss(dial(x, Sf_1), S_0)$. It is assumed that there are successor state axioms for *Safe*, *At* and the functional fluents *info* and *comb*.

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

$$Poss(read(x), s) \equiv At(x, s) \tag{14}$$

The successor state axiom for K is as follows:

$$\begin{aligned} & Poss(a, s) \rightarrow\ K(s'', do(a, s)) \equiv \\ & \quad \exists s'(K(s', s) \wedge (s'' = do(a, s'))) \wedge \\ & \quad\ ((a \neq read(x)) \vee (\exists x\ (a = read(x)) \\ & \quad\ \wedge (info(x, s) = info(x, s')))) \end{aligned} \tag{15}$$

Note that the axiomatization entails:

$$\begin{aligned} & Kref(info(Ppr), do(read(Ppr), S0)) \wedge \\ & \quad info(Ppr, do(read(Ppr), S0)) = \\ & \quad\ comb(Sf_1, do(read(Ppr), S0)) \end{aligned} \tag{16}$$

Since the successor state axioms ensure that a *read* action does not change *At*, *Safe*, *Comb* and *Info*, it is the case that $\exists x Poss(dial(x, Sf_1), do(read(Ppr), S_0))$ and therefore the axiomatization entails that $\exists x Open(Sf_1, do(dial(x, Sf_1), do(read(Ppr), S_0)))$.

Correctness of the Solution

The following 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 fluents P (other than K) and all knowledge producing actions a, if $P(s)$ then $P(do(a, s))$.*

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.

Theorem 2 (Default Persistence of Ignorance) *If $\neg$**Knows**(P, s) then $\neg$**Knows**$(P, do(a, s))$ unless $Poss(a, s)$ and either a is an A_i and P is the corresponding φ_i in the successor state axiom for K, or P is a fluent whose successor state axiom specifies that it is changed by action a.*

Proof: For $\neg$**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 $K(s'', do(a, s))$, it is the case that $s'' = do(a, s')$ for some s' such that $K(s', s)$. Since $P(s')$ is false for some s', $P(do(a, s'))$ will (by the successor state axiom for P) also be false, unless either (1) the successor state axiom for P specifies that the effect of a is to make P true, $Poss(a, s)$ is true, and the conditions for this change are satisfied in s, or (2) a is a knowledge-producing action A_i, the corresponding φ_i in the successor state axiom for K is $P(s) \leftrightarrow P(s')$, $P(s)$ is true, and $Poss(a, s)$. If neither is the case by the successor state axiom for K, there will be an s'' such that $K(s'', do(a, s))$ where $P(s'')$ is false and therefore $\neg$**Knows**$(P, do(a, s))$ will be true. □

Finally, it is a property of this specification that agents never forget.

Theorem 3 (Memory) *For all fluents P and situations s, if* **Knows**(P, s) *then* **Knows**$(P, do(a, s))$ *unless the effect of a is to make P false.*

The proof is similar to that of the previous theorem.

Consider again the successor state axiom for *broken* given in sentence 8. If **Knows**$(\neg Broken(Obj_1), S_0)$ is true, then **Knows**$(\neg Broken(Obj_1), do(drop(Obj_2), S_0))$ must also be true. Also, note that if **Knows**$(Fragile(Obj_2), S_0)$ and **Knows**$(Poss(drop(Obj_2)), S_0)$ are true, then **Knows**$(Broken(Obj_2), do(drop(Obj_2), S_0))$ must also be true.

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:

$$\textbf{Kwhether}(\exists x\,(Object(x) \wedge \neg Holding(x) \wedge Ofshape(x, Shape_1))) \quad (17)$$

Now the formula φ_i associated with each knowledge-producing action is of the form $\alpha_i(s) \equiv \alpha_i(s')$, where α_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, 18 is understood as an abbreviation for 19.

$$\textbf{Knows}(\textbf{Knows}(P)) \quad (18)$$

$$\forall s_1\ K(s_1, S_0) \rightarrow (\forall s_2\ K(s_2, s_1) \rightarrow P(s_2)) \quad (19)$$

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 20) to be a property of the logic.

$$\textbf{Knows}(\phi) \rightarrow \textbf{Knows}(\textbf{Knows}(\phi)) \quad (20)$$

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:

$$\begin{array}{c} Init(s_1) \rightarrow (K(s, s_1) \rightarrow Init(s)) \\ \neg Init(s_1) \rightarrow (K(s, s_1) \rightarrow \neg Init(s)) \\ Init(S_0) \quad \neg Init(do(a, s)) \end{array}$$

The various restrictions are listed below.[3] 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 $\forall s_1{:}Init, s_2{:}Inits_3{:}Init$
$K(s_2, s_1) \wedge K(s_3, s_1) \rightarrow K(s_3, s_2)$

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 level will satisfy the same set of properties.*

The proof involves showing for each restriction that if the restriction holds for s, then it holds for $do(a, s)$.

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.

Reasoning

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

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

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[4]) s_{gr}, the question is whether the axiomatization of the domain $\mathcal{F}$ entails $G(s_{gr})$ where G is an arbitrary sentence. 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 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 whose situation argument is a situation variable, or the situation constant S_0, $\mathcal{R}_\Theta[A] = A$.

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

$$Poss(a, s) \rightarrow [F(x_1, \ldots, x_n, do(a, s)) \equiv \Phi_F] \quad (21)$$

then

$$\mathcal{R}_\Theta[F(t_1, \ldots, t_n, do(\alpha, \gamma))] = \Phi_F|^{x_1,\ldots,x_n,a,s}_{t_1,\ldots,t_n,\alpha,\sigma} \quad (22)$$

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. For simplicity, it is assumed that there are only knowledge-producing operators of type sense—$Sense_1 \ldots Sense_n$. Two definitions are needed for the specification to follow. When ϕ is an arbitrary sentence and s a situation term, then apply(ϕ, 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 apply$^{-1}(\phi)$ is the result of removing the last argument position from all the fluents in ϕ.

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

[4]It is also assumed that this plan is known to be executable [Reiter, 1991], i.e., each step is possible.

vi $\mathcal{R}_\Theta[\mathbf{Knows}(W, do(Sense_i, s))] = ((\varphi_i(s) \rightarrow \mathbf{Knows}(\varphi_i \rightarrow W, s)) \wedge (\neg\varphi_i(s) \rightarrow \mathbf{Knows}(\neg\varphi_i \rightarrow W, s))$

In the following theorem, $\mathcal{F}$ is the axiomatization of the domain including $\mathcal{F}_{ss}$, the successor state axioms.

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

$$\mathcal{F} \models G(s_{gr}) \leftrightarrow \mathcal{F} - \mathcal{F}_{ss} \models \mathcal{R}^*_\Theta G(s_{gr})$$

The proof is based on an induction over all ground action terms [Reiter, 1993]. Each regression step preserves logical equivalence given an axiomatization of the form developed here (i.e. successor state axioms). 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. Since each step preserves equivalence, the whole process results in an equivalent formula.

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 axiom for *Red* is given below:

$$Red(do(a, s)) \equiv (Acid(s) \wedge a = Test_1) \vee (Red(s) \wedge a \neq Test_1) \quad (23)$$

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

$$\forall a, s, s'',\ K(s'', do(a, s)) \equiv \exists s'(K(s', s) \wedge (s'' = do(a, s'))) \wedge ((a \neq Sense_R) \vee ((a = Sense_R) \wedge (Red(s) \equiv Red(s')))) \quad (24)$$

The formula to be initially regressed is

$$\mathbf{Knows}(Acid, do(Sense_R, do(Test_1, S_0))) \quad (25)$$

Step **vi** of the definition of $\mathcal{R}$ is used with 25 to yield 26.

$$(Red(do(Test_1, S_0)) \rightarrow \mathbf{Knows}(Red \rightarrow Acid, do(Test_1, S_0))) \wedge (\neg Red(do(Test_1, S_0)) \rightarrow \mathbf{Knows}(\neg Red \rightarrow Acid, do(Test_1, S_0))) \quad (26)$$

This is then regressed to sentence 27 by steps **iii**, **iv**, and **v** of the regression definition along with 23.

$$(Acid(S_0) \rightarrow \mathbf{Knows}(Acid \rightarrow Acid, S_0)) \wedge (\neg Acid(S_0) \rightarrow \mathbf{Knows}(\neg Acid \rightarrow Acid, S_0)) \quad (27)$$

Sentence 27 is clearly entailed by $Acid(S_0)$ and so 25 is entailed by the original theory. Note that 27 can be rewritten as a sentence in an ordinary modal logic because the only situation term is S_0.

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.

Acknowledgments

We thank Ray Reiter and Fangzhen Lin for useful discussions on the situation calculus and the frame problem. Additionally, we would like to thank both Ray Reiter and Sheila McIlraith for comments on an earlier version 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, 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, 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.

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

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

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

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

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.

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, 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, R. 1993. Proving properties of states in the situation calculus. *Artificial Intelligence.* to appear.

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

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.