Reasoning about the Interaction of Knowlege, Time and Concurrent Actions in the Situation Calculus

Richard B. Scherl
Computer Science Department
Monmouth University
West Long Branch, New Jersey
07764-1898 U.S.A.
rscherl@monmouth.edu

Abstract

A formal framework for specifying and developing agents/robots must handle not only knowledge and sensing actions, but also time and concurrency. Researchers have extended the situation calculus to handle knowledge and sensing actions. Other researchers have addressed the issue of adding time and concurrent actions. Here both of these features are combined into a uni£ed logical theory of knowledge, sensing, time, and concurrency. The result preserves the solution to the frame problem of previous work, maintains the distinction between indexical and objective knowledge of time, and is capable of representing the various ways in which concurrency interacts with time and knowledge. Furthermore, a method based on regression is developed for solving the projection problem for theories speci£ed in this version of the situation calculus.

1 Introduction

The aim of this paper is to develop a uni£ed approach for axiomatizing the interaction of knowledge, sensing, time, and concurrency. Actions have preconditions which may include knowledge preconditions. Sensing actions alter knowledge. The knowledge produced depends upon the relative time at which sensing actions occur and also whether or not other sorts of actions occur concurrently. All of this interacts with the agent's evolving knowledge of time.

Consider a robot gathering knowledge about its environment. It moves about while concurrently panning the camera and at various points senses the environment for the presence of objects with various characteristics. The knowledge obtained through sensing (positions of objects of various sizes, shapes and colors) depends upon the position of the robot at a particular point in time, the angle of the camera, etc. For many purposes, the results of sensing actions that occur at the same time are important. Not only do we have the need for two distinct concurrent sensing actions in binocular stereopsis, but also in the simultaneous use of other features such as texture gradients and shading to achieve knowledge of depth relationships. Here it is relative, not absolute time, that is important.

Furthermore, speci£cation of an agent's ability to achieve a goal in general involves requiring that the agent know what to do to arrive at a goal state [Moore, 1980; Lesperance *et al,* 2000]. As the ability to achieve particular goals will often involve the ability to perform concurrent actions, the integration of knowledge and concurrency is an important step in fully formalizing these aspects of ability.

We develop our framework within the situation calculus - a £rst-order formalism for representing and reasoning about the effects of actions. The language is based upon the dialect of the situation calculus used in the *Cognitive Robotics Group* at the University of Toronto. Certainly, another formalism could have been used. Within A.I. numerous varying formalisms are available for nrepresenting and reasoning about action (to name a few, [Shanahan, 1995; Baral *et al,* 1997]). Knowledge has been incorporated into a number of these (for example, [Lobo *et al,* 2001; Thielscher, 2000]). Outside of A.I. proper, there is also [Fagin *et al,* 1995] on the interaction of knowledge and time.

But by working in the situation calculus, we are able to extend previous work on reasoning (by regression and theorem proving) to cover the language developed here as well. Furthermore, our work is suitable to be incorporated into the robot programming languages GOLOG and CONGOLOG. It can then be used to specify agents that must reason about the interactions of time, knowledge, and concurrent actions (including sensing actions).

The situation calculus [McCarthy and Hayes, 1969] is a language for modeling dynamically changing worlds. Changes to the world are the results of *actions,* while possible world histories consisting of sequences of actions are represented by *situations.* The situation calculus can be used for agent planning via goal regression. Reiter [Reiter, 1991] proposed a simple solution to the frame problem, an approach to axiomatization within the situation calculus. Although this approach to the frame problem requires certain simplifying assumptions, it has proven to be useful and is the foundation for both goal regression and the programming language GOLOG. Goal regression was extended by Scherl and Levesque[Scherl and Levesque, 2003] to apply to an agent who can sense properties of the external world (e.g., read numbers on a piece of paper or determine the shape of an object).

This paper[1] combines and extends the work of Scherl and Levesque [Scherl and Levesque, 2003] incorporating the model of *concurrency* and *time* presented by Reiter [Reiter, 2001]. At the same time, Reiter's simple solution to the frame problem is preserved. The real difficulty is to perform the synthesis in such a way that the important distinction between indexical and objective time [Lesperance and Levesque, 1995] is preserved.

If the agent currently knows the absolute time, then he knows the absolute time after executing an action. But if he doesn't know the absolute time, then he only knows that he began executing the action some number of time units ago, unless of course he reads a clock. While maintaining these properties, the results presented here allow the representation of the various ways in which actions (including sensing actions and possibly other concurrent actions) interact with time and knowledge. The method of regression is also extended to work with this augmented language.

Section 2 gives a quick introduction to the situation calculus and Section 3 does the same for the foundational axioms. The representation of knowledge and sensing actions are covered in Section 4. Concurrency is integrated into the framework in Section 5. Section 6 covers some additional constructs of the language and illustrates their representational power. A number of properties of the formulation are discussed in Section 7. Regression is covered in Section 8. Finally, Section 9 is the conclusion.

2 Situation Calculus and the Frame Problem

Space does not permit a full exposition of the background material on the situation calculus. The framework developed in [Reiter, 2001] is followed and full details may be found there or in [Scherl, 2003]. We assume that the frame problem has been handled by utilizing successor state axioms.

3 Foundational Axioms

Following [Lin and Reiter, 1994; Reiter, 2001] the foundational axioms for the situation calculus are utilized. These axioms provide us with a definition of $s \preceq s'$, which says that there is a sequence of zero or more executable actions that move from situation s to situation s'. Again, space does not permit full development of this material here. Full details may be found in [Reiter, 2001; Scherl, 2003].

4 An Epistemic 'Fluent

Scherl and Levesque [Scherl and Levesque, 2003] adapt the standard possible-world model of knowledge to the situation calculus, as first done by Moore [Moore, 1980]. Informally, one can 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.

'An earlier version of some of this work has appeared in [Zimmerbaum and Scherl, 2000].

To treat knowledge as a fluent, they introduce a binary relation $\mathrm{K}(s', s)$, read "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 $\mathrm{BROKEN}(y, s)$.[2]

The notation Knows(P(now),s) (read as P is known in situation s) can then be introduced as as an abbreviation for a formula that uses K. For example

$$\mathbf{Knows}(\mathrm{BROKEN}(y, \mathrm{now}), s) \stackrel{\mathrm{def}}{=} \forall s'\ \mathrm{K}(s', s) \rightarrow \mathrm{BROKEN}(y, s').$$

The special indexical now is instantiated with a situation variable upon expansion.

The approach also handles actions that make known the denotation of a term. For this case, one needs the notation Kref(T(now), s) defined as follows:

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

In general, there may be many knowledge-producing actions, as well as many ordinary actions. To characterize all of these, we have following the presentation in [Scherl and Levesque, 2003], a function SR (for sensing result), and for each action a, a sensing-result axiom of the form:

$$\mathrm{SR}(\alpha(\vec{x}), s) = r \equiv \phi_\alpha(\vec{x}, r, s) \tag{1}$$

This result is "YES" if "Q" is true and "NO" otherwise. The symbols are given in quotes to indicate that they are not fluents. The sensing result function for SENSEQ is axiomatized as follows:

$$\mathrm{SR}(\mathrm{SENSE}_\mathrm{Q}, s) = r \equiv (r = \text{"YES"} \wedge \mathrm{Q}(s)) \vee (r = \text{"NO"} \wedge \neg \mathrm{Q}(s)) \tag{2}$$

For ordinary actions, the SR result is always the same, with the specific result not being significant. For example, we could have:

$$\mathrm{SR}(\mathrm{PICKUP}(x), s) = r \equiv r = \text{"OK"} \tag{3}$$

In the case of a READ action that makes the denotation of the term τ known, we would have:

$$\mathrm{SR}(\mathrm{READ}_\tau, s) = r \equiv r = \tau(s) \tag{4}$$

Therefore, τ has the same denotation in all worlds s'' such that $\mathrm{K}(s'', \mathrm{DO}(\mathrm{READ}_\tau, s))$, and so Kref($\tau$,DO(READ$_\tau$, S)) is true.

The form of the successor state axiom for K without concurrency is as follows:

$$K(s'', (\mathrm{DO}(a, s))) \equiv (\exists s')\, s'' = \mathrm{DO}(a, s') \wedge K(s', s) \wedge \mathrm{POSS}(a, s') \wedge \mathrm{SR}(a, s) = \mathrm{SR}(a, s') \tag{5}$$

The relation K at a particular situation $\mathrm{DO}(a, s)$ is completely determined by the relation at s and the action a.

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

5 Concurrency

As originally de£ned in the situation calculus [McCarthy and Hayes, 1969], actions had to occur sequentially, with one action completed before another could begin. Furthermore, there was no facility to deal with the continuous passage of time. This contrasted with other formalisms such as the event calculus which could naturally handle concurrent actions and continuous time.

5.1 Concurrency with Knowledge

The work of Pinto [Pinto, 1994] and Reiter [Reiter, 2001] proposed an approach to dealing with concurrency, natural actions and continuous time while still maintaining the solution to the frame problem. Reiter [Reiter, 2001] de£ned a new sort *concurrent,* sets of simple actions. Variables $a, a', ..$ represent the sort *actions* and $c, c', ...$ represent the sort *concurrent.* In Reiter's notation, the time of an action's occurrence is the value of that action's temporal argument. Thus an action has the form $A(\vec{x}, t)$ and for each action an axiom of the form $\mathrm{TIME}(\mathbf{A}(\vec{x}, t)) = t$ is required to indicate the time of the action.

Concurrent actions are sets of ordinary actions that are taken to represent instantaneous acts. An action with duration is represented by two instantaneous actions — a start action and an end action. Additionally, the foundational axioms are modi£ed to rule out the possibility of prior actions having later times.

So if we want to represent a PRESSING action with duration (as in pressing a button that keeps a light on), the approach is to de£ne two actions; STARTPRESS and ENDPRESS. We also must introduce a auent PRESSING. The needed successor state axiom is as follows:

$$\begin{aligned}&\textsc{Pressing}(\textsc{do}(a, s)) \equiv \\ &\quad a = \textsc{startpress} \vee \\ &\quad\quad (\textsc{Pressing}(s) \wedge a \neq \textsc{endpress})\end{aligned} \tag{6}$$

This approach to representing actions with duration is something that we make use of here.

But, the use of a temporal argument for actions is problematic in the presence of knowledge. Given our successor state axiom for K, it would require the agent to know the time after any action, even if it was unknown in the previous situation. To avoid this, we can not represent time as an argument to the instantaneous actions.

Instead, we represent the instantaneous actions and associated times as a tuple of the form $< a, t >$ with functions ACTION and TIME de£ned, returning the £rst and second elements of the tuple:

$$\textsc{action}(< a, t >) = a \tag{7}$$

$$\textsc{time}(< a, t >) = t \tag{8}$$

These pairs, represented by variables $p, p', ...$ are elements of the sort *action-time pairs.* Concurrent Actions are now a set of such tuples. The sort *action* contains actions without a temporal argument.

We also have the following

$$\textsc{start}(\textsc{do}(p, s)) = \textsc{time}(p) \tag{9}$$

which is needed to relate the time of the action/time pair to the start of a situation. There may also be an axiom giving the start time of the initial situation So. We also de£ne a variant notation:

$$\textsc{time}(\textsc{do}(p, s)) = \textsc{time}(p) \tag{10}$$

We adopt, without signi£cant change, Reiter's requirement that concurrent actions be coherent, that is there is at least one action-time pair p in the collection, and the time of all pairs in the collection is the same:

$$\begin{aligned}&\textsc{coherent}(c) \equiv \\ &\quad (\exists p)\, p \in c \wedge (\exists t)(\forall p')[p' \in c \rightarrow \textsc{time}(p') = t].\end{aligned} \tag{11}$$

A set of action-time pairs are coherent if each of them have the same time component.

The de£nition of *time* can readily be extended to sets of concurrent actions and this allows us to de£ne the function *start* of a situation resulting from the execution of a concurrent action.

$$\begin{aligned}&\textsc{coherent}(c) \rightarrow \\ &\quad [\textsc{time}(c) = t \equiv \exists p\,(p \in c \wedge \textsc{time}(p) = t)] \\ &\quad \wedge \textsc{start}(do(c, s)) = \textsc{time}(c)].\end{aligned} \tag{12}$$

The predicate POSS(c\ $) means that it is possible to execute concurrent action *c* in situation .s.

$$\begin{aligned}&\textsc{Poss}(\textsc{action}(p), s) \wedge \textsc{time}(p) > \textsc{time}(s) \\ &\quad \rightarrow \textsc{Poss}(p, s),\end{aligned} \tag{13}$$

$$\textsc{Poss}(p, s) \rightarrow \textsc{Poss}(\{p\}, s), \tag{14}$$

$$\textsc{Poss}(c, s) \rightarrow \textsc{coherent}(c) \wedge (\forall p)\,[p \in c \rightarrow \textsc{Poss}(p, s)]. \tag{15}$$

If it is possible to execute a concurrent actions, then it is coherent and each of the simple actions is possible.

We implicitly assume an additional sort ranging over time points which can be integers, rationals or reals; depending on how one wants to model time. The standard Arabic numerals are used to represent time points. Additionally, the symbols for addition and subtraction are interpreted as the usual operations on numbers (integers, reals etc.)

5.2 Precondition Interactions

We need to be able to conclude when a particular concurrent action c is possible in a situation *c* in order for the machinery being developed in the rest of this paper to work. Unfortunately, Sentence 15 does not suffice. The conditional can not be changed to a biconditional because of the precondition interaction problem [Pinto, 1994; 1998].

This issue needs to be handled by the axiomatizer of the domain. For example, the axiomatizer might provide the following axiom:

$$\begin{aligned}&\textsc{Poss}(\{p_1, p_2\}, s) \equiv \\ &\quad \textsc{Poss}(\textsc{action}(a_1), s) \wedge \textsc{Poss}(\textsc{action}(a_2), s) \wedge \\ &\quad \neg \textsc{Precint}(a_1, a_2) \wedge \textsc{coherent}(\{p_1, p_2\})\end{aligned} \tag{16}$$

As discussed in [Pinto, 1994; 1998], the axiomatization of PRECINT is domain dependent and can be done at increasing levels of detail.

For the purposes of this paper, the point here is that whatever solution is used for the precondition interaction problem in the ordinary situation calculus carries over to the case of knowledge and sensing.

5.3 Successor State Axiom for K with Concurrency

The Successor State Axiom for K using concurrency can be stated in several alternative ways depending on what conditions one wishes to apply regarding the agent's knowledge of time. We continue to require that the relation K at a particular situation DO(<?, *s)* is completely determined by the relation at * and the set of concurrent actions c.

The following successor state axiom models an agent who knows how much time is passing[3]. This is an agent who has an accurate clock.

$$\begin{aligned} K(s'', (\text{DO}(c, s)) \equiv \\ & (\exists s', c')\ s'' = \text{DO}(c', s') \wedge\ K(s', s) \wedge\ \text{POSS}(c', s') \\ & \wedge\ (\forall p')\ p' \in c'\ (\exists p)\ p \in c \\ & \wedge\ \text{ACTION}(p') = \text{ACTION}(p) \\ & \wedge\ (\forall p)\ p \in c\ (\exists p')\ p' \in c' \\ & \wedge\ \text{ACTION}(p) = \text{ACTION}(p') \\ & \wedge\ \text{START}(s'') = \text{START}(s') + (\text{TIME}(c) - \text{START}(s)) \\ & \wedge\ (\forall p)\ p \in c \rightarrow \\ & \quad \text{SR}(\text{ACTION}(p), s) = \text{SR}(\text{ACTION}(p), s') \end{aligned} \tag{17}$$

After executing a set of concurrent actions *c* in situation s, as far as the agent knows, it could be in a situation s'' **iff** s'' is the result of performing c' in some previously accessible situation s', provided that c is possible in s' and that *s'* is identical to *s* in terms of what is being sensed. Furthermore, it is required that the concurrent action c' being performed in all situations accessible from .s be identical to c in terms of the individual actions that make up the set.

Note that it is not required that the TIME of the actions in all the accessible situations be identical. If this were the case, it would force the agent to know the objective time after executing any action. Rather, it is only required that the difference between the *start* of the current situation and the *start* of the previous situation be the same in all accessible situations (including the actual situation). This requirement does ensure that the agent knows how much time is passing, but the objective time is only known if it was known before the action has occurred.

5.4 Concurrency and Sensing

One can readily imagine cases of sensing actions where the desired result of sensing (knowing whether or not some proposition holds) depends upon some other action occurring concurrently. For example, the light needs to be turned on while the camera is clicked. If the light is off, then sensing produces no knowledge.

Consider representing, the requirement that the light switch be pressed while SENSEp occurs for the knowledge of whether or not P holds to result from the execution of SENSKp. We need to de£ne a predicate SCOND.

$$\text{SCOND(SHNSEp,cs)} = \text{PRESSING(s)} \tag{18}$$

[3]Other possibilities are considered in [Scherl, 2003].

Now the successor state axiom for K needs to be modi£ed to include SCOND as well.

$$\begin{aligned} K(s'', (\text{DO}(c, s)) \equiv \\ & (\exists s', c')\ s'' = \text{DO}(c', s') \wedge\ K(s', s) \wedge\ \text{POSS}(c', s') \\ & \wedge\ (\forall p')\ p' \in c'\ (\exists p)\ p \in c \\ & \wedge\ \text{ACTION}(p') = \text{ACTION}(p) \\ & \wedge\ (\forall p)\ p \in c\ (\exists p')\ p' \in c' \\ & \wedge\ \text{ACTION}(p) = \text{ACTION}(p') \\ & \wedge\ \text{START}(s'') = \text{START}(s') + (\text{TIME}(c) - \text{START}(s)) \\ & \wedge\ (\forall p)\ p \in c \wedge \\ & \quad (\text{SCOND}(\text{ACTION}(p), s) \wedge \text{SCOND}(\text{ACTION}(p), s') \rightarrow \\ & \quad\ \text{SR}(\text{ACTION}(p), s) = \text{SR}(\text{ACTION}(p), s') \end{aligned} \tag{19}$$

For every action an appropriate SCON axiom needs to be written. For most actions, the action is simply:

$$\text{SCOND}(a, s) \equiv \text{T} \tag{20}$$

6 The Language and Examples

6.1 Further Constructs

Wc need some way to refer to the current time without specifying the value of the current time. To achieve this we use the special indexical term now. Upon expansion the term is replaced with the appropriate situation term. So, START(now) can be used to refer to the current time. Here we illustrate by example. The agent's knowing the objective time is expressed as $\exists t$ Know (start (now) $= t, s$). This expands into

$$\exists t \forall s' (K(s', s) \rightarrow \text{START}(s') = t).$$

We augment our language with a number of additional expressions. These are based on ideas developed by Lesperancc and Levesque [Lesperance and Levesque, 1995] and require the use of the notion of precedence of situations as de£ncd earlier. Note that we distinguish between the $<$ relation on integers used to represent time points and the $\prec$ relations on situations as de£ned in the foundational axioms for the situation calculus.

The macro Happened is introduced to allow one to talk about an action occurring at a particular time point.

$$\begin{aligned} \textbf{Happened}(t, Act, s) \stackrel{\text{def}}{=} (\exists c, s', s'')\ s'' = \text{DO}(c, s') \wedge \exists p \in c \\ \wedge\ \text{ACTION}(p) = Act \wedge\ \text{TIME}(p) = t \wedge \\ s'' \preceq s \wedge\ s' \prec s'' \end{aligned} \tag{21}$$

It speci£es that an action occurred prior to s and it was time t at some point during the action's duration.

The macro Wasat is introduced to allow one to assert that a fluent was true at a particular point in time.

$$\begin{aligned} \textbf{Wasat}(t, \text{P(then)}, s) \stackrel{\text{def}}{=} (\exists s') P(\text{then})\{s'/\text{then}\} \wedge \\ t = \text{START}(s') \vee\ t > \text{START}(s') \\ \wedge\ \neg(\exists s'')\ s' \prec s'' \prec s \wedge\ \text{START}(s'') \leq t \end{aligned} \tag{22}$$

It speci£es that P held at *s'* and t was the time of s' or s' preceded *t* and no other situation after *s'* preceded t. Here we introduce another special indexical then which is needed to ensure that the correct situation is substituted into the situation argument of the predicate which is the middle argument to Wasat.

7 Properties of the Formulation

First of all, we show that the distinction between indexical and objective time is preserved.

Proposition 1 (Persistence of Ignorance of Objective Time) *For all situations s, if* $\neg\exists t\ \mathbf{Know}(\text{START}(\text{now}) = t, s)$ *then in* $do(c, s)$ *where c is any concurrent action it is also the case that* $\neg\exists t\ \mathbf{Know}(\text{START}(\text{now}) = t, s)$ *unless there is some* $a \in c$ *with an* SR *axiom equivalent to the following:*

$$\text{SR}(a, s) = r \equiv r = \text{TIME}(s)$$

Proposition 2 (Persistence of Knowledge of Objective Time) *For all situations s, if* $\exists t\ \mathbf{Know}(\text{start}(\text{now}) = t, s)$ *then in* $do(c, s)$ *where c is any concurrent action it is also the case that* $\exists t\ \mathbf{Know}(\text{start}(\text{now}) = t, s)$

Even if agents do not know the objective time, they do know how much time has passed since the last occurrence of a particular action or the last time at which a particular nuent was true.

Proposition 3 (Knowledge of Indexical Time 1) *For every,* Act *such that* Act *occurs in s,*

$$\exists t\ \mathbf{Knows}(\exists t'\mathbf{Happened}(t', Act, \text{now}) \wedge (\text{TIME}(\text{now}) - t' = t, s)$$

Proposition 4 (Knowledge of Indexical Time 2) *For every* P *such that there is some* $s' \prec s$ *such that* $P(s')$ *holds,*

$$\exists t\ \mathbf{Knows}(\exists t'\mathbf{Wasat}(t', \text{P}(\text{then}), \text{now}) \wedge (\text{TIME}(\text{now}) - t' = t, s)$$

Additionally, the crucial results of [Scherl and Levesque, 2003] carry over to the case considered here. These include

Proposition 5 (Default Persistence of Ignorance) *For an action a and a situation s, if* -iKnows(P, s) *holds and the axiomatization entails*

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

and

$$\forall y \neg\mathbf{Knows}((\text{POSS}(\alpha) \wedge \text{SR}(\alpha) = y) \rightarrow \text{P}, s)$$

then

-iKnows(P, DO(a, *s*))

holds as well.

Proposition 6 (Knowledge Incorporation) *For a knowledge-producing action* α, *a* □*uent or the negation of a* □*uent a* □*uent r the negation of a* \uent P, *and a situation a, if the axiomatization entails*

$$\exists y\ \text{Knows}(\text{F} \equiv \text{SR}(\alpha) = y, s)$$

and also

$$\text{F}(s), \quad \text{POSS}(\alpha, s),$$

and

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

hold, then

$$\text{Knows}(\text{P}, \text{DO}(\alpha, s))$$

holds as well.

Proposition 7 (Memory) *For all* □*uents* P *and situations s,* if Knows(P, s) *holds then* $\text{Knows}(\text{P}, \text{DO}(\alpha, s))$ *holds as long as the axiomatization entails*

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

These results ensure that actions only affect knowledge in the appropriate way.

8 Reasoning

A regression operator $\mathcal{R}$ is de£ned relative to a set of successor state axioms Θ. Space limitations here only allow a sketch of the regression operators. Full details may be found in [Scherl, 2003].

The operators satisfy the following regression theorem:

Theorem 1 *For any ground situation term* s_{gr} *and formula G:*

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

Here $\mathcal{F}$ is the initial axiomatization of the domain and $\mathcal{F}_{ss}$ is the set of successor state axioms.

8.1 Regression Operators

The regression operator *7Z* is de£ned relative to a set of successor state axioms Θ. The £rst four parts of the de£nition of the regression operator $\mathcal{R}_\Theta$ concern ordinary (i.e. not knowledge-producing) actions [Reiter, 2001]. They are exactly the same as those in [Reiter, 2001]. Additionally, it is necessary to correctly regress the equality predicate as discussed in [Scherl and Levesque, 2003; Reiter, 2001].

Additional steps are needed to extend the regression operator to knowledge-producing actions. Two de£nitions are needed for the speci£cation to follow. When φ is an arbitrary sentence and *s* a situation term, then $\varphi[s]$ is the sentence that results from instantiating every occurrence of now in φ with s'. The reverse operation φ^{-1} is the result of instantiating every occurrence of *s'* in φ with now.

Step v covers the case of regressing the Knows operator through a non-knowledge-producing action. Step vi covers the case of regressing the Knows operator through a knowledge producing action. In the de£nitions below, ‚s' is a new situation variable.

v. Whenever *c* does not contain a knowledge-producing action,

$$\mathcal{R}_\Theta[\mathbf{Knows}(\text{W}, \text{DO}(c, s))] = \text{Knows}(\text{Poss(c)} \rightarrow \mathcal{R}_\Theta[\text{W}[\text{DO}(\text{trans}(c), s')]]^{-1}, s).$$

vi. Whenever *c* does contain a knowledge-producing action,

$$\mathcal{R}_\Theta[\mathbf{Knows}(W, \text{DO}(c, s))] = \exists y\ \text{SR}(\text{SENSE}_t, s) = y \wedge \mathbf{Knows}((\text{POSS}(c) \wedge \text{SR}(\text{SENSE}_r) = y) \rightarrow \mathcal{R}_\Theta[W[\text{DO}(\text{trans}(c))]]^{-1}, s)$$

The special function trans replaces *c* with a c' that has the identical action in each of the action-time pairs. But it replaces the time portion with a relative time. This is an expression of the form (3 = TIMK(now)). So, if *c* occurs in a situation term of the form DO(C, *s)* and TIME(r) is 7 and START(s) is 4, then the 7 in the time part of every action-time pair in *c* would be replaced by (3 = TIME(now)). These are properly handled by the operators for regressing equality predicates.

The regression operators for Happened and Wasat follow:

viii.

$$\mathcal{R}[\mathbf{Happened}(t, Act, \text{DO}(c, s))] = (\mathbf{Happened}(t, Act, s) \wedge t \leq \text{START}(s)) \vee \exists p \in c\,(\text{ACTION}(p) = Act \wedge t = \text{START}(\text{DO}(c, s)))$$

ix.

$$\mathcal{R}[\mathbf{Wasat}(t, W, \mathrm{DO}(c, s))] = (\mathbf{Wasat}(t, W, s) \wedge t < \mathrm{START}(s)) \vee (W(s) \wedge t \geq \mathrm{START}(s) \wedge t < \mathrm{START}(\mathrm{DO}(c, s)))$$

The end result of regression is a sentence (possibly much larger) in a language without actions. The language is a modal language with both temporal and epistemic operators.

9 Conclusions

The results reported in this paper can be combined with Concurrent, Temporal Golog or RGolog[Reiter, 2001] or CONGOLOG[Giacomo *et al,* 1997] to specify and control an agent (such as the robot discussed in Section 1) that concurrently moves about its environment and performs various actions including sensing actions. Future work will address the issue of developing a good theorem proving method for the language resulting from regression. This method needs to combine modal theorem proving with a method for solving integer constraints.

Acknowledgments

Numerous helpful discussions have been held with Yves Lesperance on many of the topics covered in this paper. Useful suggestions made by Patrick Doherty, Daniele Nardi, and a number of anonymous reviewers of earlier versions of this paper have been incorporated. Additional thanks are due to Steve Zimmerbaum with whom the initial stages of this work were carried out. This research was partially supported by NSF grants SES-9819116 and CISE-9818309.

References

[Baralet al., 1997] Chitta Baral, Michael Gelfond, and Alessandro Provetti. Representing actions: Laws, observations and hypotheses. *Journal of Logic Programming,* 31(I-3):201-243, 1997.

[Fagin *et al,* 1995] R. Fagin, J.Y. Halpern, Y.O. Moses, and M.Y. Vardi. *Reasoning about Knowledge.* MIT Press, Cambridge, Mass, 1995.

[Giacomoefa/., 1997] G. De Giacomo, Y. Lesperance, and H. J. Levesque. Reasoning about concurrent execution, prioritized interrupts, and exogeneous actions in the situation calculus. In *Proceedings of the Fifteenth International Joint Conference on Artiicial Intelligence*\ Nagoya, Japan, 1997.

[Lesperance and Levesque, 1995] Yves Lesperance and Hector Levesque. Indexical knowledge and robot action—a logical account. *Artilcial Intelligence,* 73(1-2):69-115, February 1995.

[Lesperance *et al,* 2000] Yves Lesperance, Hector J. Levesque, Fangzhen Lin, and Richard B. Scherl. Ability and knowing how in the situation calculus. *Studia Logica,* 66(1), 2000.

[Lin and Reiter, 1994] Fangzhen Lin and Raymond Reiter. State constraints revisited. *Journal of Logic and Computation,* 4(5):655-678, 1994.

[Lobo et al, 2001] Jorge Lobo, Gisela Mendez, and Stuart Taylor. Knowledge and the action description language *A. Journal of Logic Programming,* 1 (2): 129-184, 2001.

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

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

[Pinto, 1994] J.A. Pinto. *Temporal Reasoning in the Situation Calculus.* PhD thesis, Department of Computer Science, University of Toronto, Toronto, Ontario, 1994. Available as technical report KRR-TR-94-1.

[Pinto, 1998] Javier A. Pinto. Concurrent actions and interacting effects. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixth International Conference (KR-98),* pages 292-303. Morgan Kaufmann Publishing, 1998.

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

[Reiter, 2001] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems.* The MIT Press, Cambridge, Massachusetts, 2001.

[Scherl and Levesque, 2003] Richard Scherl and Hector Levesque. Knowledge, action, and the frame problem. *Artiicial Intelligence,* 144:1-39, 2003.

[Scherl, 2003] Richard Scherl. Axiomatizing the interaction of knowledge, time, and concurrency with the situation calculus, journal version in progress, 2003.

[Shanahan, 1995] Murray Shanahan. A circumscriptive calculus of events. *Arti cial Intelligence,* 77(2):249-284, September 1995.

[Thielscher, 2000] Michael Thielscher. Representing the knowledge of a robot. In A. Cohn, F. Giunchiglia, and B. Selman, editors, *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR),* pages 109-120. Morgan Kaufmann, 2000.

[Zimmerbaum and Scherl, 2000] Stephen Zimmerbaum and Richard Scherl. Knowledge, time, and concurrency in the situation calculus. In C. Castelfranchi and Y. Lesperance, editors, *Intelligent Agents VII: Proceedings of the 2000 Workshop on Agent Theories, Architectures, and Languages (ATAL-2000),* LNAI, Berlin, 2000. Springer-Verlag.