The Frame Problem in the Situation Calculus: A Simple Solution (Sometimes) and a Completeness Result for Goal Regression

Raymond Reiter

Department of Computer Science
University of Toronto
Toronto, Canada
and
The Canadian Institute for Advanced Research

1 Introduction

Ever since it was first pointed out by McCarthy and Hayes in 1969 [5], the frame problem has remained an obstacle to the formalization of dynamically changing worlds. Despite many attempts, no completely satisfactory solution has been obtained. Without presuming to have solved the frame problem in its full generality, we propose a solution for an interesting special case, and explore some of its consequences. Specifically, we have two objectives in this chapter:

1. To provide an analysis of two recent proposals for dealing with the frame problem in the situation calculus (Pednault [6], Schubert [9]) and to show how they can be combined, under a suitable closure assumption that is appropriate in settings when the effects of all actions on all fluents can be specified.

2. To show how the axioms arising from the analysis of 1. provide a systematic treatment of goal regression (Waldinger [10]) for plan synthesis, together with natural conditions under which regression is provably sound and complete.

Proofs for all results in this chapter may be found in (Reiter [8]).

2 Axiomatizing Change

In this section, we describe a class of axioms that characterize the effects of actions on fluents. We follow this by a survey of two different approaches to solving the frame problem, namely, the proposals of Pednault [6] and Schubert [9]. Finally, we provide an axiomatization combining the central ideas of Pednault and Schubert.

2.1 Effect Axioms

Following Pednault [6], we assume that the effects of actions on fluents are specified by *effect axioms* of the following forms:

Positive Effect Axiom for Fluent R wrt Action a

For each action $a(\mathbf{x})$ and fluent R, there is one axiom of the form:

$$\pi_a(\mathbf{x}, s) \wedge \varepsilon_R^+(\mathbf{x}, \mathbf{y}, s) \supset R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

Here, the variables $\mathbf{x}, \mathbf{y}$, and s are implicitly universally quantified.[1] The variables of $\mathbf{x}$ and $\mathbf{y}$ are distinct from one another. The metaformula $\pi_a(\mathbf{x}, s)$ denotes the *action preconditions* of the action $a(\mathbf{x})$. These are the prerequisites that must be satisfied in order that action a can be carried out, and they depend only on a, not on R. The metaformula $\varepsilon_R^+(\mathbf{x}, \mathbf{y}, s)$ denotes the *fluent preconditions* under which action $a(\mathbf{x})$, if performed, leads R to become true for $\mathbf{y}$ in the successor state $do(a(\mathbf{x}), s))$.

Examples

1. The effect on the predicate *broken* of dropping something:

$$holding(r, x, s) \wedge y = x \wedge fragile(y) \supset broken(y, do(drop(r, x), s)).$$

Here, $\pi_{drop}(r, x, s)$ is $holding(r, x, s)$, the precondition for robot r to drop object x in state s. $\varepsilon_{broken}^+(r, x, y, s)$ is $y = x \wedge fragile(y)$, the prerequisite for y becoming broken in that state $do(drop(r, x), s)$ resulting from robot r dropping x in state s.

2. How picking something up affects the predicate *holding*:

$$[(\forall z)\neg holding(r, z, s)] \wedge \neg heavy(x) \wedge nexto(r, x, s) \wedge r' = r \wedge y = x$$
$$\supset holding(r', y, do(pickup(r, x), s)).$$

[1] In the subsequent development, lower case Roman letters in formulas will denote variables, and are understood to be implicitly universally quantified whenever no explicit quantifier is indicated.

Here, $\pi_{pickup}(r, x, s)$ is

$$[(\forall z)\neg holding(r, z, s)] \wedge \neg heavy(x) \wedge nexto(r, x, s),$$

the precondition for robot r to pick up x in state s. $\varepsilon^{+}_{holding}(r, x, r', y, s)$ is $r' = r \wedge y = x$.

Negative Effect Axiom for Fluent R wrt Action a

For each action $a(\mathbf{x})$ and fluent R, there is one axiom of the form:

$$\pi_a(\mathbf{x}, s) \wedge \varepsilon^{-}_{R}(\mathbf{x}, \mathbf{y}, s) \supset \neg R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

As before, the metaformula $\pi_a(\mathbf{x}, s)$ denotes the action preconditions of the action a. The metaformula $\varepsilon^{-}_{R}(\mathbf{x}, \mathbf{y}, s)$ denotes the *fluent preconditions* under which action $a(\mathbf{x})$, if performed, leads R to become false for $\mathbf{y}$ in the successor state $do(a(\mathbf{x}), s))$.

Example

$$hasglue(r, s) \wedge broken(x, s) \wedge y = x \supset \neg broken(y, do(repair(r, x), s)).$$

Here, $\pi_{repair}(r, x, s)$ is $hasglue(r, s) \wedge broken(x, s)$, the precondition for robot r to repair object x in state s. $\varepsilon^{-}_{repair}(r, x, y, s)$ is $y = x$.

2.2 Frame Axioms

As has been long recognized [5], axioms other than effect axioms are required for formalizing dynamic worlds. These are called *frame axioms*, and they specify the action *invariants* of the domain, i.e., those fluents unaffected by the performance of an action.

Examples

Dropping things does not affect an object's color:

$$holding(r, x, s) \wedge color(y, c, s) \supset color(y, c, do(drop(r, x), s)).$$

Not breaking things:

$$\begin{aligned} &holding(r, x, s) \wedge \neg broken(y, s) \wedge [y \neq x \vee \neg fragile(y)] \\ &\quad \supset \neg broken(y, do(drop(r, x), s)). \end{aligned}$$

In general, a frame axiom has one of the following syntactic forms:

Positive Frame Axiom for Fluent R wrt Action a

$$\pi_a(\mathbf{x}, s) \wedge \phi^{+}_{R}(\mathbf{x}, \mathbf{y}, s) \wedge R(\mathbf{y}, s) \supset R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

When a's action preconditions are satisfied and the prerequisites $\phi_R^+(\mathbf{x}, \mathbf{y}, s)$ hold, R remains true for $\mathbf{y}$ after a is performed.

Negative Frame Axiom for Fluent R wrt Action a

$$\pi_a(\mathbf{x}, s) \wedge \phi_R^-(\mathbf{x}, \mathbf{y}, s) \wedge \neg R(\mathbf{y}, s) \supset \neg R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

When a's action preconditions are satisfied and the prerequisites $\phi_R^-(\mathbf{x}, \mathbf{y}, s)$ hold, R remains false for $\mathbf{y}$ after a is performed.

2.3 Frame Axioms: Pednault's Proposal

In view of the large expected number of frame axioms, together with the practical problem of having to think of them all, it is natural to ask whether there is some systematic way of obtaining them from the effect axioms. Pednault [6] provides just such a method, but with a critical proviso. Consider again the positive and negative effect axioms for a fixed action a and a fixed fluent R:

$$\pi_a(\mathbf{x}, s) \wedge \varepsilon_R^+(\mathbf{x}, \mathbf{y}, s) \supset R(\mathbf{y}, do(a(\mathbf{x}), s)),$$

$$\pi_a(\mathbf{x}, s) \wedge \varepsilon_R^-(\mathbf{x}, \mathbf{y}, s) \supset \neg R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

Suppose we make the following:

Completeness Assumption for Fluent Preconditions

The fluent precondition $\varepsilon_R^+(\mathbf{x}, \mathbf{y}, s)$ specifies all the conditions under which action a, if performed, will lead to the truth of R for $\mathbf{y}$ in a's successor state. Similarly, $\varepsilon_R^-(\mathbf{x}, \mathbf{y}, s)$ specifies all the conditions under which action a, if performed, will lead to the falsity of R for $\mathbf{y}$ in a's successor state.

Now, by the completeness assumption, we can reason as follows: Suppose a's action preconditions $\pi_a(\mathbf{x}, s)$ hold. Suppose further that both $R(\mathbf{y}, s)$ and $\neg R(\mathbf{y}, do(a(\mathbf{x}), s))$ hold. Then R, which was true in state s, was made false by action a. By the completeness assumption, the only way R could become false is if $\varepsilon_R^-(\mathbf{x}, \mathbf{y}, s)$ were true. This intuition can be expressed axiomatically by:

$$\pi_a(\mathbf{x}, s) \wedge R(\mathbf{y}, s) \wedge \neg R(\mathbf{y}, do(a(\mathbf{x}), s)) \supset \varepsilon_R^-(\mathbf{x}, \mathbf{y}, s).$$

This is logically equivalent to:

$$\pi_a(\mathbf{x}, s) \wedge R(\mathbf{y}, s) \wedge \neg \varepsilon_R^-(\mathbf{x}, \mathbf{y}, s) \supset R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

A symmetric argument yields the axiom:

$$\pi_a(\mathbf{x}, s) \wedge \neg R(\mathbf{y}, s) \wedge \neg \varepsilon_R^+(\mathbf{x}, \mathbf{y}, s) \supset \neg R(\mathbf{y}, do(a(\mathbf{x}), s)).$$

These have precisely the syntactic forms of positive and negative frame axioms and, by virtue of the argument leading to these, they play exactly the role of frame axioms. We conclude that, provided the completeness assumption is true, there is a systematic way of obtaining the frame axioms from the effect axioms.

Notice that the completeness assumption makes no reference to the action preconditions $\pi_a(\mathbf{x}, s)$ for the action a. These may well fail to characterize all the preconditions of the action a, but this incompleteness would in no way compromise the intuitive correctness of the above systematic transformation of the effect axioms to obtain the frame axioms. The plausibility of this transformation relies solely on the assumed completeness of the fluent preconditions.

Notice also the intimate connection between the completeness assumption and the so-called *qualification problem* (McCarthy [4]). By assuming complete information about fluent preconditions, we are effectively assuming no qualifications about these preconditions. Equivalently, if there are any unstated qualifications, we are taking them to be false. The completeness assumption makes no such claims about possible qualifications for the action preconditions.

Example

Consider the positive effect axiom for *broken* wrt *drop*:

$$holding(r, x, s) \wedge y = x \wedge fragile(y) \supset broken(y, do(drop(r, x), s)).$$

The completeness assumption for this setting is that the only precondition for y being broken as a result of dropping x is that y be fragile and the same as x. If this assumption is accepted, then the negative frame axiom for the fluent *broken* wrt the action *drop* is:

$$\begin{aligned} &holding(r, x, s) \wedge \neg broken(y, s) \wedge \neg[y = x \wedge fragile(y)] \\ &\quad \supset \neg broken(y, do(drop(r, x), s)). \end{aligned}$$

To obtain the frame axioms in this way for all fluent-action pairs requires considering a large number of "vacuous" effect axioms. For example, to obtain a frame axiom for the fluent *color* wrt action *drop*, consider the positive effect axiom for *color* wrt *drop*:

$$holding(r, x, s) \wedge false \supset color(y, c, do(drop(r, x), s)).$$

From this we obtain the negative frame axiom for *color* wrt *drop*:

$$holding(r, x, s) \wedge \neg color(y, c, s) \supset \neg color(y, c, do(drop(r, x), s)).$$

This illustrates two problems with Pednault's proposal:

- To systematically determine the frame axioms for all fluent-action pairs from their effect axioms, we must enumerate (or at least consciously consider) all these effect axioms, including the "vacuous" ones. In particular, we must enumerate all fluent-action pairs for which the action has no effect on the fluent's truth value, which really amounts to enumerating most of the frame axioms directly.
- The number of frame axioms so obtained is $2 \times \mathcal{A} \times \mathcal{F}$, where $\mathcal{A}$ is the number of actions, and $\mathcal{F}$ the number of fluents. Some of these may be vacuously true (i.e., when the fluent precondition of the corresponding effect axiom is $true$), but in general, we are faced with the usual difficulty associated with the frame problem – too many frame axioms.

Finally, the completeness assumption warrants a bit more attention: when can it fail? The answer seems to be, roughly speaking: whenever the effects of an action on a fluent are not completely known. Consider the effect of pulling the trigger of a (possibly loaded) gun on the fluent *loaded*. It is indeterminate whether the gun will be loaded after the action *pulltrigger*. Thus, both the positive and negative effect axioms for *loaded* wrt *pulltrigger* are "vacuous":

$$false \supset loaded(do(pulltrigger, s)),$$

$$false \supset \neg loaded(do(pulltrigger, s)).$$

The corresponding frame axioms, obtained from these effect axioms under the completeness assumption, are:

$$loaded(s) \supset loaded(do(pulltrigger, s)),$$

$$\neg loaded(s) \supset \neg loaded(do(pulltrigger, s)).$$

The first of these is intuitively false. One way out, at least for this example, is to refine the effect axioms to make the effects of *pulltrigger* on *loaded* determinate:

$$containsbullets(n, s) \wedge n \geq 2 \supset loaded(do(pulltrigger, s)),$$

$$containsbullets(n, s) \wedge n \leq 1 \supset \neg loaded(do(pulltrigger, s)).$$

It is unlikely that such fixes are possible, in a natural way, for all indeterminate action-fluent pairs.

2.4 Frame Axioms: Schubert's Proposal

Schubert [9], elaborating on a proposal of Haas [3], argues in favor of what he calls explanation closure axioms for representing the usual frame axioms. We illustrate Schubert's approach with an example.

Consider the fluent *holding*, and suppose that both $holding(r, x, s)$ and $\neg holding(r, x, do(a, s))$ are true. How can we explain the fact that *holding* ceases to be true? If we assume that the only way this can happen is if the robot r put down or dropped x, we can express this with the explanation closure axiom:

$$\begin{aligned} &holding(r, x, s) \wedge \neg holding(r, x, do(a, s)) \\ &\qquad \supset a = putdown(r, x) \vee a = drop(r, x). \end{aligned}$$

As usual, all variables, including the action variable a, are implicitly universally quantified. To see how this functions as a frame axiom, rewrite it in the logically equivalent form:

$$\begin{aligned} &holding(r, x, s) \wedge a \neq putdown(r, x) \wedge a \neq drop(r, x) \\ &\qquad \supset holding(r, x, do(a, s)). \end{aligned} \tag{1}$$

This says that all actions other than *putdown* and *drop* leave *holding* invariant,[2] which is the standard form of a frame axiom (actually, a set of frame axioms, one for each action distinct from *putdown* and *drop*).

In general, an *explanation closure axiom* has one of the two forms:

$$R(\mathbf{x}, s) \wedge \neg R(\mathbf{x}, do(a, s)) \supset \alpha_R(\mathbf{x}, a, s),$$

$$\neg R(\mathbf{x}, s) \wedge R(\mathbf{x}, do(a, s)) \supset \beta_R(\mathbf{x}, a, s).$$

In these, the action variable a is universally quantified. These say that if ever the fluent R changes truth value, then α_R or β_R provides an exhaustive explanation for that change.

As before, to see how explanation closure axioms function like frame axioms, rewrite them in the logically equivalent form:

$$R(\mathbf{x}, s) \wedge \neg \alpha_R(\mathbf{x}, a, s) \supset R(\mathbf{x}, do(a, s)),$$

and

$$\neg R(\mathbf{x}, s) \wedge \neg \beta_R(\mathbf{x}, a, s) \supset \neg R(\mathbf{x}, do(a, s)).^3$$

[2] To accomplish this, we shall require unique names axioms like $pickup(r, x) \neq drop(r', x')$. We shall explicitly introduce these later.

[3] Schubert [9] omits the action preconditions for a in his examples of explanation closure axioms, as we do here. Nevertheless, they should be present. We shall restore them in the next section.

These have the same syntactic form as frame axioms with the important difference that action a is universally quantified. Whereas there would be $2 \times \mathcal{A} \times \mathcal{F}$ frame axioms, there are just $2 \times \mathcal{F}$ explanation closure axioms. This parsimonious representation is achieved by quantifying over actions in the explanation closure axioms.

Schubert proposes that explanation closure axioms must be provided independently of the effect axioms. Like the effect axioms, these are domain-dependent. In particular, Schubert argues that they cannot be obtained from the effect axioms by any kind of systematic transformation. Thus, Schubert and Pednault entertain conflicting intuitions about the origins of frame axioms.

As Schubert observes, his appeal to explanation closure as a substitute for frame axioms involves an assumption.

The Explanation Closure Assumption

The only way the fluent R's truth value could have changed from true to false under action a is if α_R *were true. This means, in particular, that* α_R *completely characterizes all those actions a that can lead to this change; similarly for* β_R.

We can see clearly the need for this assumption from the example explanation closure axiom (1). If, in the intended application, there were an action (say, $eat(r, x)$) that could lead to r no longer holding x, axiom (1) would be false.

3 A Simple Solution to the Frame Problem (Sometimes)

3.1 The Basic Idea: An Example

The basic idea is best illustrated with an example. Suppose there are two positive effect axioms for the fluent *broken*:

$$holding(r, x, s) \wedge y = x \wedge fragile(y) \supset broken(y, do(drop(r, x), s)),$$

$$bomb(b) \wedge nexto(b, y, s) \supset broken(y, do(explode(b), s)).$$

These can be rewritten in the logically equivalent form:

$$\begin{array}{l} \{[holding(r, x, s) \wedge a = drop(r, x) \wedge y = x \wedge fragile(y)] \\ \quad \vee [bomb(b) \wedge nexto(b, y, s) \wedge a = explode(b)]\} \\ \qquad \supset broken(y, do(a, s)). \end{array}$$

This can be represented more compactly by introducing the new predicate $Poss(a,s)$, meaning that action a is possible in state s:

$$\begin{aligned} Poss(a,s) \wedge [(\exists r,x)a = drop(r,x) \wedge y = x \wedge fragile(y) \\ \vee(\exists b)a = explode(b) \wedge nexto(b,y,s)] \\ \supset broken(y, do(a,s)), \end{aligned} \tag{2}$$

$$holding(r,x,s) \supset Poss(drop(r,x),s),$$

$$bomb(b) \supset Poss(explode(b),s).$$

Similarly, consider the negative effect axiom for *broken*:

$$hasglue(r,s) \wedge broken(x,s) \wedge y = x \supset \neg broken(y, do(repair(r,x),s)).$$

In exactly the same way, this can be rewritten as:

$$\begin{aligned} Poss(a,s) \wedge (\exists r,x)a = repair(r,x) \wedge y = x \supset \\ \neg broken(y, do(a,s)), \end{aligned} \tag{3}$$

$$hasglue(r,s) \wedge broken(x,s) \supset Poss(repair(r,x),s).$$

Notice that, with the exception of the introduction of the predicate $Poss$, this transformation is similar to the first stage of Clark's [1] *predicate completion* technique for logic programs.

Now we can appeal to the following completeness assumption:

Axiom (2) characterizes all the conditions under which action a leads to y being broken.

Then if $Poss(a,s), \neg broken(y,s), broken(y, do(a,s))$ are all true, the truth value of *broken* must have changed because

$$\begin{aligned} (\exists r,x)a = drop(r,x) \wedge y = x \wedge fragile(y) \\ \vee(\exists b)a = explode(b) \wedge nexto(b,y,s) \end{aligned}$$

was true. This intuition can be formalized, after some logical simplification, by the following explanation closure axiom:

$$\begin{aligned} Poss(a,s) \wedge \neg broken(y,s) \wedge broken(y, do(a,s)) \supset \\ (\exists r)a = drop(r,y) \wedge fragile(y) \vee (\exists b)a = explode(b) \wedge nexto(b,y,s). \end{aligned}$$

Similarly, (3) yields the following explanation closure axiom:

$$Poss(a,s) \wedge broken(y,s) \wedge \neg broken(y, do(a,s)) \supset (\exists r)a = repair(r,y).$$

3.2 The General Case

The previous example obviously generalizes. We suppose given, for each fluent R, the following two *general effect axioms*:

General Positive Effect Axiom for Fluent R

$$Poss(a, s) \wedge \gamma_R^+(a, s) \supset R(do(a, s)).^4 \tag{4}$$

General Negative Effect Axiom for Fluent R

$$Poss(a, s) \wedge \gamma_R^-(a, s) \supset \neg R(do(a, s)). \tag{5}$$

These two axioms are systematically obtained from all of the positive (respectively, negative) effect axioms for R by the same process illustrated in the previous examples.

The predicate $Poss$ must also be defined.

Action Precondition Axioms

For each action A,

$$\pi_A(s) \supset Poss(A, s),$$

where $\pi_A(s)$ is the formula for A's action preconditions.

We shall make the following:

Completeness Assumption

Axioms (4) and (5), respectively, characterize all the conditions under which action a can lead to R becoming true (respectively, false) in the successor state.

Hence, 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*; similarly, if R's truth value changes from *true* to *false*. This informally stated assumption can be represented axiomatically by the following:

Explanation Closure Axioms

$$Poss(a, s) \wedge R(s) \wedge \neg R(do(a, s)) \supset \gamma_R^-(a, s), \tag{6}$$

$$Poss(a, s) \wedge \neg R(s) \wedge R(do(a, s)) \supset \gamma_R^+(a, s). \tag{7}$$

We shall sometimes appeal to:

[4] Henceforth, in formulas of this kind, we shall suppress all but the action and state arguments.

Unique Names Axioms for Actions

For distinct action names a and a',

$$a(\mathbf{x}) \neq a'(\mathbf{y}).$$

Identical actions have identical arguments:

$$a(x_1, ..., x_n) = a(y_1, ..., y_n) \supset x_1 = y_1 \wedge ... \wedge x_n = y_n.$$

Finally, we shall need:

Unique Names Axioms for States

$$S_0 \neq do(a, s),$$ [5]

$$do(a, s) = do(a', s') \supset a = a' \wedge s = s'.$$

The following slightly generalizes a result of Pednault[6]:

Result 1 *Let T be a firstr-order theory that entails $\neg\exists(Poss(a,s) \wedge \gamma_R^+(a,s) \wedge \gamma_R^-(a,s))$, where $\exists$ denotes the existential closure of the formula in its scope. Then T entails that the general effect axioms (4) and (5), together with the explanation closure axioms (6) and (7), are logically equivalent to:*

$$Poss(a, s) \supset [R(do(a, s)) \equiv \gamma_R^+(a, s) \vee R(s) \wedge \neg\gamma_R^-(a, s)]. \tag{8}$$

The requirement that $\neg\exists(Poss(a,s) \wedge \gamma_R^+(a,s) \wedge \gamma_R^-(a,s))$ be entailed by the background theory T simply guarantees the integrity of the effect axioms (4) and (5); under these circumstances, it will be impossible for both $R(do(a,s))$ and $\neg R(do(a,s))$ to be simultaneously derived. Notice that by the unique names axioms for actions, this condition is satisfied by the example of Section 3.1.

We call formula (8) the *successor state axiom for fluent R*. For the example of Section 3.1, the successor state axiom for $broken$ is:

$$\begin{gathered} Poss(a, s) \supset [broken(y, do(a, s)) \equiv \\ (\exists r)a = drop(r, y) \wedge fragile(y) \vee (\exists b)a = explode(b) \wedge nexto(b, y, s) \\ \vee broken(y, s) \wedge \neg(\exists r)a = repair(r, y)]. \end{gathered}$$

[5] In the subsequent development, S_0 will denote the initial state, and will be the only state constant allowed in the theory.

3.3 Summary

Our proposed solution relies on the completeness assumption of the previous section for each fluent R. This yields the following axioms:

1. Successor state axioms: for each fluent R,

$$Poss(a,s) \supset [R(do(a,s)) \equiv \gamma_R^+(a,s) \vee R(s) \wedge \neg\gamma_R^-(a,s)].$$

2. For each action A, a single action precondition axiom of the form:

$$\pi_A(s) \supset Poss(A,s).$$

3. Unique names axioms for actions and for states.

Ignoring the unique names axioms (whose effects can be compiled), this axiomatization requires $\mathcal{F}+\mathcal{A}$ axioms in total, compared with the $2\times\mathcal{A}\times\mathcal{F}$ explicit frame axioms that would otherwise be required. There is also a significant improvement in space complexity of our axiomatization over one appealing to explicit frame axioms. More precisely, it is easy to see that the space complexity is decreased by a factor of $\mathcal{A}$, roughly the same decrease in complexity as is obtained by the simple axiom-counting argument above.

The conciseness and perspicuity of this axiomatization relies on two things: quantification over actions via the transformations of Section 3.1, and the Generalized Completeness Assumption.

4 Plan Synthesis

The standard formal account of plan synthesis views this as a theorem-proving problem. A plan to achieve a certain goal $G(s)$ is obtained as a side effect of a proof of $(\exists s)G(s)$ from premises axiomatizing the domain of application and conditions holding in the initial state S_0 (Green [2]). Any binding for the variable s as a result of such a proof is a successful plan. Here, $G(s)$ is any first-order formula with the single free variable s.

4.1 Executable Plans and Ghost States in the Situation Calculus

The above theorem-proving account of plan synthesis has a flaw; it is possible to obtain *non-executable* plans this way. To see why, consider an axiomatization $\mathcal{F}$ of a blocks world domain for which

$$\mathcal{F} \models onfloor(B, do(drop(B), S_0)).$$

Then trivially, $do(drop(B), S_0)$ is a plan to get B onto the floor; but this plan need not be executable, i.e., the action precondition of *drop*, namely, $holding(B, S_0)$, need not be entailed by $\mathcal{F}$. Under these circumstances, $do(drop(B), S_0)$ is a *ghost state* of $\mathcal{F}$; it is not reachable from S_0 by any executable sequence of actions. Notice that this is a feature of the situation calculus, not of our proposed solution to the frame problem. To circumvent this problem of non-executable plans, we propose to modify the theorem to be proved in the process of plan synthesis. Define a new predicate $ex(s)$ as follows:

$$ex(s) \equiv s = S_0 \vee (\exists a, s')s = do(a, s') \wedge Poss(a, s') \wedge ex(s'). \tag{9}$$

Intuitively, $ex(s)$ states that s is an executable plan, i.e., it is composed of a sequence of plan steps, each of whose action preconditions is true in the previous state. We reformulate the problem of synthesizing a plan to achieve a goal G as the task of establishing that

$$\mathcal{F} \models (\exists s)G(s) \wedge ex(s).$$

Any plan so obtained is guaranteed to be executable.

4.2 Plan Synthesis by Goal Regression

> "In solving a problem of this sort, the grand thing is to be able to reason backward."
>
> Sherlock Holmes, *A Study in Scarlet*

This section provides foundations for a systematic, backward-reasoning-style proof theory for plan synthesis, usually called goal regression (Waldinger [10]). The approach substantially generalizes some ideas of Pednault [6], who considers goal regression within the setting defined by his solution to the frame problem.

Our objective is a method for establishing that

$$\mathcal{F} \models (\exists s)G(s) \wedge ex(s)$$

whenever $\mathcal{F}$ is a suitable axiomatization of a dynamic world. To do so we must be more precise than we have been about the first-order language of our axiomatization. Let $\mathcal{L}$ be a sorted first-order language with two disjoint sorts for actions and states, and suppose these sorts are disjoint from any other sorts of the language. Assume $\mathcal{L}$ has the following vocabulary:

- Variables: Infinitely many of each sort.

- Function symbols of sort *state*: There are just two of these – the constant S_0 and the binary function symbol *do*, which takes arguments of sort *action* and *state*, respectively.
- Function symbols of sort *action*: Finitely many.
- Other function symbols: Infinitely many of sort other than *action* and *state* for each arity, none of which take an argument of sort *state*.
- Predicate symbols:
  1. A distinguished binary predicate symbol $Poss$ taking arguments of sort *action* and *state*, respectively.
  2. A distinguished unary predicate symbol ex taking argument of sort *state*.
  3. A distinguished equality symbol $=$.
  4. Finitely many predicate symbols, distinct from the predicate symbols $Poss$ and ex, each of which takes, among its arguments, exactly one of sort *state*; these are the *fluents*. Notice that the predicates $Poss$ and ex, which do take an argument of sort *state*, are not fluents.
  5. Infinitely many predicate symbols of each arity, none of which take arguments of sort *state*.
- Logical constants and punctuation: As usual.

Notice that $\mathcal{L}$ does not allow state-dependent functions like *employer-of(x,s)*, or *Canadian-prime-minister(s)*.

The predicate symbols of $\mathcal{L}$ other than $Poss$ and ex are called *domain predicate symbols*. The intention is that these will denote domain-specific relations like *holding*, *nexto*, etc. The two non-domain predicate symbols $Poss$ and ex are included to facilitate our axiomatization of change, as in Section 3.2.

Definition: The Simple Formulas

Let s be a variable of $\mathcal{L}$ of sort *state*. The formulas of $\mathcal{L}$ that are *simple wrt s* are defined to be the smallest set such that:

1. $F(\vec{t}, s)$ and $F(\vec{t}, S_0)$ are simple wrt s when F is a fluent and $\vec{t}$ are terms.[6] An equality atom mentioning no state variable at all, or mentioning only the state variable s, is simple wrt s. Any other atom with predicate symbol other than $Poss$ or ex is simple wrt s.

[6] For notational convenience, we assume that the last argument of a fluent is always the (only) argument of sort *state*.

2. If S_1 and S_2 are simple wrt s, so are $\neg S_1$, $S_1 \wedge S_2$, $S_1 \vee S_2$, $S_1 \supset S_2$, $S_1 \equiv S_2$.

3. If S is simple wrt s, so are $(\exists x)S$ and $(\forall x)S$ whenever x is a variable not of sort *state*.

In short, the simple formulas wrt s are those that mention only domain predicate symbols, whose fluents do not mention the function symbol *do*, which do not quantify over variables of sort *state*, and which have at most one free variable s of sort *state*.

Recall that *Poss* was a predicate symbol introduced to provide a uniform representation for successor state axioms, and that the preconditions for specific actions were defined by action precondition axioms (Section 3.2).

Definition: Action Precondition Axiom

An action precondition axiom is a formula of the form:

$$(\forall x_1, \cdots, x_n, s)[\Pi_A \supset Poss(A(x_1, \cdots, x_n), s)],$$

where A is an n-ary action function of $\mathcal{L}$, and Π_A is a formula of $\mathcal{L}$ that is simple wrt s and whose free variables are among $x_1, \cdots, x_n, s$.

Notation

Suppose $\mathcal{F} \subseteq \mathcal{L}$ contains an action precondition axiom for each action function of $\mathcal{L}$, say, the following n axioms:

$$(\forall \vec{x}, s)[\Pi_{A_1} \supset Poss(A_1(\vec{x}), s)],$$

$$\vdots$$

$$(\forall \vec{z}, s)[\Pi_{A_n} \supset Poss(A_n(\vec{z}), s)].$$

Then $\mathcal{D}_{\mathcal{F}}(a, s)$ denotes the formula:

$$(\exists \vec{x})a = A_1(\vec{x}) \wedge \Pi_{A_1} \vee \cdots \vee (\exists \vec{z})a = A_n(\vec{z}) \wedge \Pi_{A_n}.$$

Recall the central role of successor state axioms in our approach to the frame problem (Section 3.2).

Definition: Successor State Axiom

A successor state axiom for an $(n+1)$-ary fluent F of $\mathcal{L}$ is a sentence of $\mathcal{L}$ of the form:

$$(\forall a, s)(\forall x_1, \ldots, x_n)Poss(a, s) \supset F(x_1, \ldots, x_n, do(a, s)) \equiv \Phi_F, \tag{10}$$

where, for notational convenience, we assume that F's last argument is of sort *state*, and where Φ_F is a simple formula wrt s, all of whose free variables are among $a, s, x_1, \ldots, x_n$.

Notice that we do not assume that successor state axioms have the exact syntactic form (8) of Section 3.2. The discussion of Section 3 was meant to motivate one way that successor state axioms of the form (10) might arise, but nothing in the development that follows depends on the approach of Section 3.2.

Definition: A Regression Operator

Let $\Theta \subseteq \mathcal{L}$ contain one successor state axiom for each distinct fluent of the language $\mathcal{L}$. The *regression operator* $\mathcal{R}_\Theta$ when applied to a formula of $\mathcal{L}$ is defined recursively as follows:

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

2. When F is a fluent whose successor state axiom in Θ is

$$(\forall a, s)(\forall x_1, \ldots, x_n) Poss(a, s) \supset F(x_1, \ldots, x_n, do(a, s)) \equiv \Phi_F,$$

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

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

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

$\mathcal{R}_\Theta[G]$ is simply that formula obtained from G by substituting suitable instances of Φ_F in F's successor axiom for each occurrence in G of a fluent atom of the form $F(t_1, \ldots, t_n, do(\alpha, \sigma))$.

The idea behind the regression operator $\mathcal{R}_\Theta$ is to reduce the depth of nesting of the function symbol do in the fluents of G by substituting suitable instances of Φ_F from (10) for each occurrence of a fluent atom of G of the form $F(t_1, \ldots, t_n, do(\alpha, \sigma))$. Since Φ_F is simple wrt s, the effect of this substitution is to replace each such F by a formula whose fluents mention only the state term σ, and this reduces the depth of nesting by one.

We say that a formula of $\mathcal{L}$ is *s-universal* iff in its prenex normal form every variable of sort *state* is universally quantified.

Definition: The Formulas $\Gamma_i(s)$

Let $G(s) \in \mathcal{L}$ have the state variable s as its only free variable. Suppose $\mathcal{F} \subseteq \mathcal{L}$ contains a successor state axiom for each fluent of $\mathcal{L}$ and an action precondition axiom for each action function of $\mathcal{L}$. Define

$$\Gamma_0(s) = G(s),$$

$$\Gamma_i(s) = (\exists a_i)\mathcal{R}_{\mathcal{F}}[\Gamma_{i-1}(do(a_i, s))] \wedge \mathcal{D}_{\mathcal{F}}(a_i, s) \quad i = 1, 2, \cdots$$

The following is the principal result of this section.

Theorem 1 (Regression Theorem) *Suppose $\mathcal{F} \subseteq \mathcal{L}$ contains the executability axiom (9), unique names axioms for states, a successor state axiom for each fluent of $\mathcal{L}$, and an action precondition axiom for each action function of $\mathcal{L}$. Suppose further that the remaining axioms of $\mathcal{F}$ are s-universal and mention only domain predicate symbols. Finally, suppose that $G(s) \in \mathcal{L}$ is simple wrt s, and that the state variable s is the only free variable of $G(s)$. Then*

1. $$\mathcal{F} \models (\exists s)G(s) \wedge ex(s)$$

   iff for some n,

   $$\mathcal{F}^- \models \Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0),$$

   where $\mathcal{F}^-$ is $\mathcal{F}$ without the executability axiom.

2. *For every n, $\Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0)$ mentions only domain predicate symbols.*

3. *For every n, S_0 is the only state term mentioned by the fluent atoms of $\Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0)$.*

4.3 Soundness and Completeness of Goal Regression

The motivation underlying the Regression Theorem is that by successively applying the regression operator to the goal statement $G(s)$, we can obtain an equivalent expression, $\Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0)$, that mentions only the initial state instead of the state variable s, and that this regressed expression will be entailed by those axioms of $\mathcal{F}$ other than the successor state and action precondition axioms. The intuition is that the successor state and action precondition axioms will have done their job through their contribution to the regression; they should have no further role to play in proving the formula $\Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0)$. Unfortunately, this intuition is false.

Example

Suppose $\mathcal{F}$ contains unique names axioms for states together with the following two successor state axioms and single action precondition axiom:

$$Poss(a,s) \supset P(do(a,s)) \equiv D(s),$$

$$Poss(a,s) \supset Q(do(a,s)) \equiv E(s),$$

$$true \supset Poss(A,s).$$

Suppose $\mathcal{F}$ also contains an initial state axiom $E(S_0)$, and a general axiom $Q(s) \supset P(s)$. Consider the goal $(\exists s)P(s) \wedge ex(s)$. Then $\Gamma_0(S_0) \vee \Gamma_1(S_0)$ is (after some slight simplification) $P(S_0) \vee D(S_0)$, and $\mathcal{F} \models P(S_0) \vee D(S_0)$; but

$$E(S_0),\ \ Q(s) \supset P(s) \not\models P(S_0) \vee D(S_0).$$

So it appears that, even after regressing a goal formula, the successor state and action precondition axioms cannot be discarded. This observation leads to the natural question: under what conditions can these axioms be discarded without sacrificing the completeness of goal regression? In other words, under what conditions will

$$\mathcal{F} \models (\exists s)G(s) \wedge ex(s)$$

iff for some n, $\Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0)$ is entailed by $\mathcal{F}$ *without* the successor state and action precondition axioms? The rest of this section is devoted to answering this question.

Definition: The s-Admissible Sentences

A sentence is *s-admissible* iff it mentions no state variable at all, or it is of the form $(\forall s)W(s)$ where s is a state variable, and $W(s)$ is simple wrt s.

The s-admissible sentences are meant to define a sizable class of general facts that can serve as domain-specific background knowledge.

Examples

Unique names axioms for actions are s-admissible. So are:

$$holding(R,B,S_0),$$

$$(\forall s,r,x)holding(r,x,s) \supset \neg ontable(x,s),$$

$$(\forall s)(\exists x)ontable(x, S_0) \wedge color(x, Red, s),$$

$$(\forall s, a)s \neq S_0 \wedge s \neq do(a, S_0) \wedge P(s) \supset Q(s),$$

The following are not s-admissible:

$$(\forall s, x)ontable(x, s) \supset (\forall s')color(x, Red, s'),$$

$$(\exists s)ontable(B, s),$$

$$(\forall s)(\forall x, r)holding(r, x, do(pickup(r, x), s)) \supset color(x, Red, s),$$

$$(\exists x)ontable(x, S_0) \wedge (\forall s)s \neq S_0 \supset \neg ontable(x, s).$$

Definition: Closure under Regression

Suppose $\mathcal{F} \subseteq \mathcal{L}$ contains a successor state axiom for each fluent of $\mathcal{L}$, and an action precondition axiom for each action function of $\mathcal{L}$. Suppose $\mathcal{S}$ is a set of s-admissible sentences of $\mathcal{L}$. Then $\mathcal{S}$ is *closed under regression wrt* $\mathcal{F}$ iff whenever $(\forall s)W(s) \in \mathcal{S}$,

$$\mathcal{S} \models (\forall s, a)\mathcal{D}_{\mathcal{F}}(a, s) \supset \mathcal{R}_{\mathcal{F}}[W(do(a, s))].$$

Theorem 2 (Soundness and Completeness of Regression)
Suppose

$$\mathcal{F} = \{ex\text{-}ax\} \cup \mathcal{F}_{ss} \cup \mathcal{F}_{ap} \cup \mathcal{F}_{uns} \cup \mathcal{F}_{\forall s} \subseteq \mathcal{L},$$

where ex-ax is the executability axiom, $\mathcal{F}_{ss}$ is a set of successor state axioms, one for each fluent of $\mathcal{L}$, $\mathcal{F}_{ap}$ is a set of action precondition axioms, one for each action function of $\mathcal{L}$, $\mathcal{F}_{uns}$ is the set of unique names axioms for states, and $\mathcal{F}_{\forall s}$ is a set of s-admissible sentences that is closed under regression wrt $\mathcal{F}$. Suppose that $G(s) \in \mathcal{L}$ has as its only free variable the state variable s, and that $G(s)$ is simple wrt s. Then

$$\mathcal{F} \models (\exists s)G(s) \wedge ex(s)$$

iff for some n

$$\mathcal{F}_{uns} \cup \mathcal{F}_{\forall s} \models \Gamma_0(S_0) \vee \cdots \vee \Gamma_n(S_0).$$

To see what the requirement that $\mathcal{F}_{\forall s}$ be closed under regression means, consider a sentence $(\forall s)W(s) \in \mathcal{F}_{\forall s}$. Regardless of whether or not $\mathcal{F}_{\forall s}$ is closed under regression wrt $\mathcal{F}$, it is possible to prove

$$\mathcal{F} \models (\forall s, a)\mathcal{D}_{\mathcal{F}}(a, s) \supset \mathcal{R}_{\mathcal{F}}[W(do(a, s))].$$

This new sentence is s-admissible. If $\mathcal{F}_{\forall s}$ is indeed closed under regression, it entails this sentence. In other words, $\mathcal{F}_{\forall s}$ completely captures all of the s-admissible facts about the domain embodied in the larger theory $\mathcal{F}$. This natural requirement is sufficient to guarantee the completeness of goal regression for plan synthesis.

Theorem 2 also provides an important relative consistency result.

Corollary 1 (Consistency) *Suppose $\mathcal{F}$ satisfies the conditions of Theorem 2. Then $\mathcal{F}$ is satisfiable iff $\mathcal{F}_{uns} \cup \mathcal{F}_{\forall s}$ is.*

5 Discussion

We have proposed a solution to the frame problem in the situation calculus for the special case of worlds for which the effects of all actions on all fluents are determined. The resulting axiomatization has a very simple form permiting an analysis of goal regression for plan synthesis. For such axiomatizations, we have proved the soundness and completeness of regression, as well as a relative consistency theorem. The ideas of this chapter can be extended in several ways:

1. Composite actions are sequences of primitive actions. For any such composite action, we wish to determine the action precondition and successor state axioms for each fluent. It turns out (Reiter [8]) that these axioms can be characterized in terms of the regression operator, which also provides a means for computing them. It is also possible to prove various properties of composite actions, e.g., equivalence, impossibility, etc.

2. In the theory of databases, the evolution of a database is determined by *transactions*, whose purpose is to update the database with new information. For example, in an educational database, there might be a transaction specifically designed to change a student's grade. This would normally be a procedure which, when invoked on a specific student and grade, first checks that the database satisfies certain preconditions (e.g., that there is a record for the student, and that the new grade differs from the old), and if so, records the new grade. Any attempt to formalize the effects of transactions immediately encounters the frame problem. Thus, in the example, it would be necessary to state that the grade-changing transaction does not affect a teacher's salary. In this setting, the situation calculus provides an ideal vehicle for representing states of the database; transactions then correspond exactly to actions. Since for databases (unlike the "real" world) the effects of all transactions on all database relations will be known, the assumptions underlying our solution to the frame problem are justified. It follows that the axiomatization of this chapter provides a specification for database updates, and that

Theorem 2 yields a sound and complete query evaluation mechanism for databases after arbitrary updates. A companion paper (Reiter [7]) will describe in greater detail how the formalism described here characterizes update transactions in databases.

Acknowledgments

Many people helped out on this one. My thanks to Leo Bertossi, Charles Elkan, Joe Halpern, Hector Levesque, Vladimir Lifschitz, Wiktor Marek, Alberto Mendelzon, John Mylopoulos, Javier Pinto, and Len Schubert for comments and suggestions in connection with these ideas.

Afterword

Writing this chapter for John McCarthy's festschrift was particularly rewarding for me because its subject draws on some of his earliest and most fundamental research in AI. The very idea of using logic as a knowledge representation language, and the situation calculus itself, stem from John's 1959 "Programs with Common Sense." This was among the first papers in the field, and it remains the most influential of those early papers. The frame problem also has a long (for AI) history. If AI can be said to have a classic problem in the way that, say, Hilbert's tenth problem was for mathematics, then the frame problem is it. We really should be calling it McCarthy's (first?) problem, since it is so strongly associated with his name. Like all good open problems, it is subtle, challenging, and it has led to significant new technical and conceptual developments in the field. Indeed, it was one of the most important motivating factors in the development of nonmonotonic reasoning, including John's own circumscription theory. So on this occasion I am especially gratified that my contribution to John's festschrift can be seen as a continuation of the research tradition that began with his 1959 paper, and that it addresses a problem that has informed much of his distinguished career.

Someone once asked John – I forget the circumstances – when AI will achieve its goals. His answer was "sometime between four and four hundred years." However accurate this prediction might be, one thing is certain: without John's contributions to the field, we would have a much longer wait.

Bibliography

[1] K.L. Clark. Negation as Failure. In H. Gallaire and J. Minker, editors,

Logic and Data Bases, pages 292–322. Plenum Press, New York, 1978.

[2] C. C. Green. Theorem Proving by Resolution as a Basis for Question-Answering Systems. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 183–205. American Elsevier, New York, 1969.

[3] A. R. Haas. The Case for Domain-Specific Frame Axioms. In F. M. Brown, editor, *The frame problem in artificial intelligence. Proceedings of the 1987 workshop*, pages 343–348, Los Altos, California. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1987.

[4] J. McCarthy. Epistemological Problems of Artificial Intelligence. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pages 1038–1044, Cambridge, MA, 1977.

[5] J. McCarthy and P. Hayes. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, Edinburgh, Scotland, 1969.

[6] E.P.D. Pednault. ADL: Exploring the Middle Ground between STRIPS and the Situation Calculus. In R.J. Brachman, H. Levesque, and R. Reiter, editors, *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR'89)*, pages 324–332. Morgan Kaufmann Publishers, Inc., San Mateo, California, 1989.

[7] R. Reiter. On Database Updates. Technical report, Department of Computer Science, University of Toronto, in preparation.

[8] R. Reiter. A Simple Solution to the Frame Problem (Sometimes). Technical report, Department of Computer Science, University of Toronto, in preparation.

[9] L.K. Schubert. Monotonic Solution of the Frame Problem in the Situation Calculus: an Efficient Method for Worlds with Fully Specified Actions. In H.E. Kyberg, R.P. Loui, and G.N. Carlson, editors, *Knowledge Representation and Defeasible Reasoning*, pages 23–67. Kluwer Academic Press, Boston, Mass., 1990.

[10] R. Waldinger. Achieving Several Goals Simultaneously. In E. Elcock and D. Michie, editors, *Machine Intelligence 8*, pages 94–136. Ellis Horwood, Edinburgh, Scotland, 1977.