Embracing Causality in Specifying the Indirect Effects of Actions*

Fangzhen Lin
Department of Computer Science
University of Toronto
Toronto, Canada M5S 1A4
Email:fl@cs.toronto.edu

Abstract

This paper considers the problem of specifying the effects of actions in the situation calculus using domain constraints. We argue that normal state constraints that refer to only the truth values of fluents are not strong enough for this purpose, and that a notion of causation needs to be employed explicitly. Technically, we introduce a new ternary predicate *Caused* into the situation calculus: *Caused(p,v,s)* if the proposition *p* is caused (by something unspecified) to have the truth value *v* in the state *s*. Using this predicate, we can represent not only *action-triggered* causal statements such as that the action *load* causes the gun to be loaded, but also *fluent-triggered* ones such as that the fact that the switch is in the up position causes the lamp to be on. The former is convenient for representing the direct effects of actions, and the latter the indirect effects.

1 Introduction

We consider the problem of formalizing the effects of actions in the situation calculus [McCarthy and Hayes, 1969]. To motivate the needs for a notion of causality in this endeavor, let us consider the following problem.

Imagine a suitcase with two locks and a spring loaded mechanism which will open the suitcase when both of the locks are in the up position. Apparently, because of the spring loaded mechanism, if an action changes the statuses of the locks, then this action may also cause, as an indirect effect, the suitcase to open.

The problem of how to describe the spring loaded mechanism by a constraint and use it to derive the indirect effects of actions is an instance of the *ramification* problem [Finger, 1986], which has been recognized as one of the central problems in reasoning about the effects of actions. In the situation calculus, the constraint that this spring loaded mechanism gives rise to can be represented as follows:[1]

$$up(L1, s) \wedge up(L2, s) \supset open(s). \quad (1)$$

Although summarizing concisely the relationship among the truth values of the three relevant propositions at any particular instance of time, this constraint is too weak to describe the indirect effects of actions. For instance, suppose that initially the suitcase is closed, the first lock in the down position, and the second lock in the up position. Suppose an action is then performed to turn up the first, lock. Then this constraint is ambiguous about what will happen next. According to it, either the suitcase may spring open or the second lock may get turned down. Although we have the intuition that the former is what will happen, this constraint is not strong enough to enforce that because there is a different mechanism that will yield a logically equivalent constraint. For instance, a mechanism that will turn down the second lock when the suitcase is closed and the first lock is up will yield the following logically equivalent one:

$$up(L1, s) \wedge \neg open(s) \supset \neg up(L2, s).$$

So to faithfully represent the ramification of the spring loaded mechanism on the effects of actions, something stronger than the constraint (1) is needed. The proposal of this paper is to appeal to causality: (through the spring loaded mechanism) the fact that both of the locks are in the up position *causes* the suitcase to open. The goal of this paper is then to make precise causal statements like this, and show how they can be used effectively to describe the effects of actions.

This paper is organized as follows. Section 2 briefly describes the version of the situation calculus used in this paper. Section 3 shows in detail how the suitcase example is solved using causality. Section 4 generalizes the method used in section 3 to a general class of theories. Section 5 applies the method of section 4 to an example to further illustrate some of the fine points of our approach. To simplify our presentation, we shall defer discussions of related work, in particular that of [Lifschitz, 1987] and that of [Haugh, 1987], to section 6. Finally section 7 concludes this paper.

"This research was supported by grants from the Government of Canada Institute for Robotics and Intelligent Systems, and from the National Science and Engineering Research Council of Canada.

[1] We use the convention that in displayed formulas, free variables are implicitly universally quantified. See the next section for a precise description of the version of the situation calculus used in this paper.

2 The Situation Calculus

We have already seen the situation calculus in action in the introduction section. We hope the notations used there are familiar and/or intuitive enough for everyone to follow. In any event, they are defined in this section.

The language of the situation calculus is a many sorted first order one. We assume the following sorts: *situation* for situations, *action* for actions, *fluent* for propositions! fluents, *truth-value* for truth values *true* and *false,* and *object* for everything else.

We use the following domain independent predicates and functions:

- The binary function *do* - for any action *a* and any situation s, *do(a,s)* is the situation resulting from performing *a* in *s.*
- The binary predicate *Holds* - for any propositional fluent *p* and any situation s, *Holds(p, s)* is true if p holds in *s.*
- The binary predicate *Poss* - for any action *a* and any situation s, *Poss(a,s)* is true if *a* is possible (executable) in s.
- The ternary predicate *Caused* - for any fluent p, any truth value v, and any situation s, *Caused(p, v, s)* is true if the fluent p is caused (by something unspecified) to have the truth value *v* in the situation *s.*

Notice that in the introduction, we wrote, for instance, $up(L1, s)$ instead of $Holds(up(L1), s)$. We consider the former to be a shorthand for the latter. We shall continue to do so in an effort to improve the readability of our formulas. Formally, if F is a fluent name of arity $object^n \rightarrow fluent$, then we define the expression $F(t_1, ..., t_n, t_s)$ to be a shorthand for the formula $Holds(F(t_1, ..., t_n), t_s)$, where $t_1, \cdots, t_n$ are terms of sort *object*, and t_s a term of sort *situation*.

We assume that all theories in this paper will include the following basic axioms:

- For the predicate *Caused*, the following two basic axioms:

$$Caused(p, true, s) \supset Holds(p, s), \quad (2)$$
$$Caused(p, false, s) \supset \neg Holds(p, s). \quad (3)$$

- For the truth values, the following unique names and domain closure axiom:

$$true \neq false \wedge (\forall v)(v = true \vee v = false). \quad (4)$$

- The unique names assumptions for fluent and action names (we assume there are only finitely many of them). Specifically, if F_1,..., F_n are all the fluent names, then we have:

$$F_i(\vec{x}) \neq F_j(\vec{y}), \ i \text{ and } j \text{ are different},$$
$$F_i(\vec{x}) = F_i(\vec{y}) \supset \vec{x} = \vec{y}.$$

  Similarly for action names.

- The *foundational axioms* in [Lin and Reiter, 1994] for the *discrete* situation calculus. These axioms characterize the structure of the space of situations. For the purpose of this paper, it is enough to mention that they include the following unique names axioms for situations:

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

3 An Example

Consider again the suitcase example described in the introduction section. Suppose that *flip(x)* is an action that flips the status of the lock *x.* Its direct effect can be described by the following axioms:[2]

$$Poss(flip(x), s) \supset$$
$$up(x, s) \supset Caused(up(x), false, do(flip(x), s)), \quad (5)$$
$$Poss(flip(x), s) \supset$$
$$\neg up(x, s) \supset Caused(up(x), true, do(flip(x), s)). \quad (6)$$

To perform *flip(x),* the object *x* must be a lock:

$$Poss(flip(x), s) \supset lock(x), \quad (7)$$

where *lock* is a unary predicate symbol. We assume that LI and *L2* are two distinct locks:

$$L1 \neq L2 \wedge lock(L1) \wedge lock(L2) \quad (8)$$

The spring loaded mechanism is now represented by the following causal rule:

$$up(L1, s) \wedge up(L2, s) \supset Caused(open, true, s). \quad (9)$$

Notice that this causal rule, together with the basic axiom (2) about causality, entail the state constraint (1). Notice also that the physical, spring loaded mechanism behind the causal rule has been abstracted away. For all we care, it may just as well be that the device is not made of spring, but of bombs that will blow open the suitcase each time the two locks are in the up position. It then seems natural to say that the fluent *open* is caused to be true by the fact that the two locks are both in the up position. This is an instance of what we have called *fluent-triggered* causal statements in the abstract. In comparison, causal statements like the effect axioms (5) and (6) are *action-triggered.*

This finishes describing our starting theory for the domain. To describe fully the effects of the actions, we need to add suitable frame axioms. Our version of the frame axioms is: Unless caused otherwise, a fluent's truth value will persist:

$$Poss(a, s) \supset \{\neg(\exists v) Caused(p, v, do(a, s)) \supset$$
$$[Holds(p, do(a, s)) \equiv Holds(p, s)]\}. \quad (10)$$

For this frame axiom to make sense, obviously, we need to minimize the predicate *Caused.* We shall circumscribe the predicate *Caused* in the set of axioms introduced so far. We shall do so with all the other predicates *(Poss* and *Holds)* fixed, because we want to condition

[2]Recall that, for instance, *up(x, s)* is defined to be a shorthand for *Holds(up{x), s).*

Caused on them. It is easy to see that the circumscription [McCarthy, 1986] yields the following *causation axioms:*

$$Caused(open, v, s) \equiv$$
$$v = true \wedge up(L1, s) \wedge up(L2, s), \quad (11)$$
$$Caused(up(x), v, s) \equiv$$
$$\{v = true \wedge (\exists s')[s = do(flip(x), s') \wedge Poss(flip(x), s') \wedge \neg up(x, s')] \vee \quad (12)$$
$$v = false \wedge (\exists s')[s = do(flip(x), s') \wedge Poss(flip(x), s') \wedge up(x, s')]\}.$$

Notice that these axioms entail the two direct effect axioms (5, 6) and the causal rule (9). In fact, they are obtained by applying the Clark completion [1978] to the predicate *Caused* in these clauses. As we shall sec in the next section, this will be true for a general class of theories.

Having computed the causal relation, we now use the frame axiom (10) to compute the effects of actions. It is easy to see that from the frame axiom (10) and the two basic axioms (2, 3) about causality, we can infer the following pseudo *successor state axiom* [Reiter, 1991]:

$$Poss(a, s) \supset \{Holds(p, do(a, s)) \equiv$$
$$Caused(p, true, do(a, s)) \vee \quad (13)$$
$$Holds(p, s) \wedge \neg Caused(p, false, do(a, s))\}$$

From this axiom and the causation axiom (12) for the fluent *up,* we then obtain the following real successor state axiom for the fluent *up:*

$$Poss(a, s) \supset \{up(x, do(a, s)) \equiv$$
$$(a = flip(x) \wedge \neg up(x, s)) \vee (up(x, s) \wedge a \neq flip(x))\}$$

Similarly for the fluent *open,* we have

Poss(a,s)) {open(do(a, s)) =
[up(L1,do(a,s)) A up(L2, do{a, s))] V open(s)}.

Now from this axiom, first eliminating up(Ll,do(a,s)) and *up(L2,do(a,s))* using the successor state axiom for *up,* then using the unique names axioms for actions, and the constraint (1) which, as we pointed out earlier, is a consequence of our axioms, we can deduce the following successor state axiom for the fluent *open:*

$$Poss(a, s) \supset \{open(do(a, s)) \equiv$$
$$a = flip(L1) \wedge \neg up(L1, s) \wedge up(L2, s) \vee$$
$$a = flip(L2) \wedge \neg up(L2, s) \wedge up(L1, s) \vee$$
$$open(s)\}.$$

Obtaining these successor state axioms solves the frame and the ramification problems for the suitcase example.

Finally it remains to be shown how *Poss* is computed. This is an instance of the *qualification problem* [McCarthy, 1977], and the standard default assumption one uses is that an action is always executable unless explicitly ruled out by the theory. In other words, we want to maximize the predicate *Poss* in the theory we have so far, including the frame axiom and the causality axioms.

To determine what policy we shall use in maximizing *Poss,* we first have to be clear what we expect from the maximization. For us, after the maximization, we want, for each action *A(x),* an *action precondition axiom* [Reiter, 1991] of the following form:

$$Poss(A(\vec{x}), s) \equiv \Phi(\vec{x}, s),$$

where Φ(x, *s)* is a formula which refers to only the truth values of the fluents in s. In other words, the preconditions of an action should be expressed in terms of the truth values of the fluents in the current state. This means that we should maximize *Poss(a, s)* with *Holds(p, s)* fixed while allowing *Caused* and *Holds(p,s')* ^ ≠ *s'* to vary. Model-theoretically, this means that an intended model is one in which the extension of *Poss(a, s)* cannot be made larger by changing the interpretation of the predicate *Caused* and the truth values of the fluents at states other than *s.*

It turns out there is a procedure based on the Clark completion for computing the maximization. The procedure is outlined in the next section. For this example, it yields the dark completion of (7):

$$Poss(flip(x), s) \equiv lock(x),$$

Having this action precondition axiom solves the qualification problem for the suitcase example.

This concludes our solution to the suitcase problem. The method used in this example can be generalized to a general class of theories, as we shall see in the next section.

4 The Method

The procedure we followed in solving the suitcase problem can be summarized as follows:

1. Start with a theory *T* that includes all the effect axioms and state constraints.
2. Minimize *Caused* in *T.* Let T" be the resulting theory.
3. Add to *V* the frame axiom (10). Let *T"* be the resulting theory.[3]
4. Maximize *Poss* in *T"* to obtain the final action theory.

Clearly, the tractability of this approach depends crucially on the form of the initial theory *T.* In the following we consider a special class of theories for which the Clark completion is enough to compute the result of minimizing *Caused* and that of maximizing *Poss.*

Before we describe this special class of theories, we first define a terminology that will be used throughout this section. Let *s* be a situation variable. We call a formula *Φ($)* a *simple state formula about s* if Φ does not mention *Poss, Caused,* or any situation term other than possibly the variable *s.* For example, $lock(L1) \wedge L1 \neq L2$ (mentions no situation term) and *up(x, s)* A *a = flip(x)*

[3] Alternatively, instead of simply adding (10), one can minimize the formula:

->(Holds(p,s) = Holds(p,do(a,s)))

in the theory *T'.* See [Lin and Reiter, 1994].

(mentions only s) are simple state formulas about s, but $up(x, do(a, s))$ and $up(x, s')$ are not since the former mentions the compound situation term $do(a, s)$, and the latter mentions the situation variable s' which is different from s.

Step 1. For each action $A(\vec{x})$, formalize the *direct effects* of A by axioms of the form:

$$Poss(A(\vec{x}), s) \supset \Phi(s) \supset Caused(F(\vec{y}), v, do(A(\vec{x}), s)), \quad (14)$$

where F is a fluent name, and $\Phi(s)$ is a simple state formula about s. For instance, the direct effect axiom (5) for the action $flip$ can be rewritten as;

$$Poss(flip(x), s) \supset x = y \wedge v = false \wedge up(y, s) \supset Caused(up(y), v, do(flip(x), s)).$$

Step 2. For each action $A(\vec{x})$, formalize the *explicit qualifications* of A by axioms of the form:

$$Poss(A(\vec{x}), s) \supset \Phi(s) \quad (15)$$

where $\Phi(s)$ is a simple state formula about s. For instance, the qualification axiom (7) for the action $flip$ is of this form.

Step 3. Formalize all *causal rules*, the constraints that will be used to derive the indirect effects of actions, by axioms of the form:

$$\Phi(s) \wedge Caused(p_1, v_1, s) \wedge \cdots \wedge Caused(p_n, v_n, s) \supset Caused(F(\vec{x}), v, s), \quad (16)$$

where *F* is a fluent, and Φ(s) a simple state formula about *s.* The causal rule (9) in the suitcase example can be rewritten straightforwardly as axioms of this form. For an example of a causal rule in which the predicate *Caused* also appears in the left hand side of the implication, suppose that we add a new fluent called *down* to our suitcase example so that *down(x, s)* is true if the lock *x* is in the down position. Clearly this new fluent is an antonym of up, so one of them is caused to be true iff the other is caused to be false:

C aused(up(x), true, s) = Caused(down(x), false, s),
Caused(up(x), false, s) = Caused(down(x), true, s).

Notice that adding the fluent *down* and the above causal rules to the suitcase example will not affect the causation axiom for the fluent *up,* although the minimization will no longer be computable by the Clark completion (see Step 5).

Step 4. Formalize all other domain knowledge by axioms of the form:

$$(\forall s)C(s), \quad (17)$$

where *C(s)* is a simple state formula about *s.* In the suitcase example, the axiom (8) will be included in this step. State constraints that are used to derive implicit action qualifications will also be included here. For instance, in the blocks world, the state constraint $(\forall x)\neg on(x, x, s)$, which says that a block cannot be on top of itself, will be included in this step. This constraint implies that the action *stack(x, y),* which puts *x* on *y,* will not be executable when *x = y.* For more examples of such *qualification state constraints,* see [Ginsberg and Smith, 1988; Lin and Reiter, 1994].

The above four steps yield a starting theory *T* about the dynamic domain of interest. Of course, in addition to the axioms given in steps 1 to 4, *T* also contains the axioms given in section 2. The remaining steps apply nonmonotonic logic to *T* to solve the frame, the ramification, and the qualification problems.

Step 5. The goal of this step is to compute for each fluent *F* a *causation axiom* of the following form:

$$Caused(F(\vec{x}), v, s) \equiv \Psi, \quad (18)$$

where is a formula which does not mention the predicate *Caused.*

This is achieved by circumscribing the predicate *Caused* in *T* with all the other predicates fixed. However, since all the axioms in *T* other than those given in Step 1 and Step 3 either do not mention *Caused* or mention it negatively, by a standard result in circumscription, the circumscription of *Caused* in *T* is equivalent to the union of *T* and the circumscription of *Caused* in the set of axioms given in Step 1 and Step 3. But these axioms are definite clauses about *Caused.* So the circumscription always has a unique minimal model. Furthermore, if the theory *T* is stratified, then the circumscription can be computed using the Clark completion. Formally, we say that *T* is *stratified* if there are no fluents $F_0, F_1, \ldots, F_n$ such that F_0 -> F_1 -> •■--> F_n -▶ F_0, where for any fluents *F* and *F', F' —➤ F (F* depends on *F')* if there is a causal rule in *T* such that *F* appears in the right hand side, and *F'* appears in the left hand side of the causal rule. For instance, the theory in the suitcase example is stratified because *up —> open* is the only dependency relation that holds for this theory. However, if we add the fluent *down,* and the above causal rules relating *up* and *down,* then the resulting theory will no longer be stratified because we'll have *up —> down —> up.*

Proposition 4.1 *Let T' be the union of the axioms in section 2, the axioms given in Step 2 and Step 4, and the Clark completion of the predicate Caused in the clauses given in Step 1 and Step 3. If the theory T is stratified, then T' is the result of circumscribing Caused in T with the other predicates fixed, i.e. for any structure M, M is a minimal model of T iff M is a model of T'.*

However, as it is well known in logic programming community, the Clark completion may be too weak if there are cycles or recursion in the causal rule (16). The following is an example of cycles:

$$Caused(up(x), v, s) \supset Caused(up(x), v, s)$$

and the following an example of recursion:

Caused(heap(x), true, s) ⊃ Caused(heap(f(x)),true, s).

This is only natural because the Clark completion is first-order, but to capture cycles and recursion we need second-order logic.

Step 6. Assume that for each fluent *F* we have computed a causation axiom of the form (18), the goal of this step is to compute for each fluent *F* a *successor state axiom* [Reiter, 1991] of the form:

$$Poss(a,s) \supset F(\vec{x}, do(a,s)) \equiv (\Phi(s) \vee (F(\vec{x},s) \wedge \Phi'(s))), \quad (19)$$

where Φ and Φ are simple state formulas about *s.*

This is achieved by replacing the occurrences of *Caused* in the pseudo successor state axiom (13) according to (18), as we have done in the suitcase example. Generally, we can obtain a successor state axiom for each fluent this way if the theory *T* is stratified:

Proposition 4.2 *If T is stratified, then there is a simple rewriting procedure by which we can obtain a successor state axiom for each fluent using the causation axioms and the pseudo-successor state axiom.*

Having computed for each fluent a successor state axiom solves the frame and the ramification problems. Successor state axioms are desirable because they have many appealing computational properties (see [Reiter, 1991]), and are the foundations underlying much of the Cognitive Robotics research project at the University of Toronto.[4]

We want to point out that even if we have computed for each fluent *F* a causation axiom of the form (18), it is not guaranteed that we'll have for each fluent a successor state axiom of the form (19). This is due to cycles such as

$$open(s) \supset Caused(open, true, s), \quad (20)$$

and recursion such as the following transitive closure axiom:

$$R(x,z,s) \wedge R(z,y,s) \supset Caused(R(x,y), true, s).$$

To handle recursion, we'll have to use second-order successor state axioms which are ones still of the form (19) but with the formulas Φ and Φ' allowed to be second-order. However, cycles cause problems in this case. For instance, given (20), we'll have a causation axiom for *open* of the form:

$$Caused(open, v, s) \equiv v = true \wedge open(s) \vee \cdots$$

So the pseudo successor state axiom will yield something like

$$Poss(a,s) \supset \{open(do(a,s)) \equiv open(do(a,s)) \vee \cdots\}$$

which is not very useful. Fortunately, there seems to be a sense that these cycles should never arise. Some authors, for example Shoham [1990], have insisted that causation be anti-reflexive. In other words, the causes of a fact should never include the fact itself. So it is wrong to write causal rules like (20). In any event, we have yet to see an example where this kind of cycle arises naturally.

Step 7. Assume that for each fluent *F* we have computed a successor state axiom of the form (19), the goal of this final step is to compute for each action *A* an *action precondition axiom* [Reiter, 1991] of the form:

$$Poss(A(\vec{x}), s) \equiv \Phi(s), \quad (21)$$

where Φ is a simple state formula.

For each action *A,* this is achieved by maximizing the relation *Poss(A(x*),s)* with *Holds(p,s)* fixed but *Caused* and *Holds(p,s')* As' $\neq$ *s* allowed to vary. The precise definition of this maximization policy will be given in the full version of this paper.[5] It turns out that there is a procedure to compute the result of the maximization using regression and the Clark completion. The procedure is a generalization of that in [Lin and Reiter, 1994], and can only be outlined here due to the space limitation. But we'll see an example in the next section.

From each causation axiom of the form (18), deduce first the following two axioms by eliminating the predicate *Caused:*

$$\Psi(v/true) \supset F(\vec{x}, s),$$
$$\Psi(v/false) \supset \neg F(\vec{x}, s),$$

where $\Psi(v/true)$ is the result of replacing the free variable *v* in Ψ by *true,* and similarly for $\Psi(v/false)$. Now for each action *A,* apply regression to these axioms and the constraints (17) given in Step 4 to generate all (non-vacuous) qualification axioms of the following form:

$$Poss(A(\vec{x}, s) \supset \Phi(s),$$

where Φ is a simple state formula about s, and is not a consequence of the axioms we have so far. Finally apply the Clark completion to these axioms to obtain an action precondition axiom of the form (21).

So to summarize, we have:

Theorem 1 *// the theory T given by steps 1 to 4 is stratified, then there is a procedure by which we can obtain a successor state axiom for each fluent, and an action precondition axiom for each action. The procedure is based on simple syntactic manipulations, and is provably correct with respect to our nonmonotonic semantics.*

5 Another Example

Having described the general procedure in last section, we now apply it to an example involving walking and shooting. This example is of interest because it has a constraint which yields an indirect effect for one action, but an implicit qualification on another.

Imagine an agent who can perform the following three actions: *start-walk, end-walk,* and *shoot.* We follow the seven-step procedure given in the last section.

Step 1. *Direct effect axioms:*

Poss(start-walk, s) ⊃
Caused(walking, true, do(start-walk, s))

Poss (end-walk, s) ⊃
Caused(walking, false, do(end-walk, s)),

Poss(shoot, s) ⊃ *Caused(dead, true, do(shoot, s)).*

[4] See ftp://ftp.ca.toronto.edu/pub/cogrob/README.html

[5] But see [Lin and Shoham, 1991] or [Lin and Reiter, 1994] for the definition of a similar policy.

Step 2. *Explicit action qualification axioms:*

$$Poss(start\text{-}walk, s) \supset \neg walking(s),$$
$$Poss(end\text{-}walk, s) \supset walking(s).$$

Step 3. *Causal rules:*

$$dead(s) \supset Caused(walking, false, s). \quad (22)$$

Step 4. There are no other domain constraints.

Step 5. Clearly, the theory generated by the above steps is stratified, so Proposition 4.1 is applicable. The Clark completion of *Caused* in the clauses in steps 1 and 3 yields:

$$Caused(dead, v, s) \equiv$$
$$v = true \wedge (\exists s')[s = do(shoot, s') \wedge Poss(shoot, s')],$$
$$Caused(walking, v, s) \equiv$$
$$v = false \wedge dead(s) \vee$$
$$v = false \wedge (\exists s')[s = do(end\text{-}walk, s') \wedge Poss(end\text{-}walk, s')] \vee$$
$$v = true \wedge (\exists s')[s = do(start\text{-}walk, s') \wedge Poss(start\text{-}walk, s')].$$

Step 6. From the pseudo successor state axiom (13) and the above causation axiom for the fluent *dead*, we obtain the following successor state axiom:

$$Poss(a, s) \supset \{dead(do(a, s)) \equiv a = shoot \vee dead(s)\}.$$

For the fluent *walking*, similar reasoning yields the axiom:

$$Poss(a, s) \supset \{walking(do(a, s)) \equiv$$
$$a = start\text{-}walk \vee$$
$$walking(s) \wedge a \neq end\text{-}walk \wedge \neg dead(do(a, s))\}.$$

Using the successor state axiom for *dead*, we can obtain the following successor state axiom for *walking*:

$$Poss(a, s) \supset \{walking(do(a, s)) \equiv$$
$$a = start\text{-}walk \vee$$
$$walking(s) \wedge \neg dead(s) \wedge a \neq end\text{-}walk \wedge a \neq shoot\}$$

Notice that the causal rule (22) has been used to derive the following indirect effect of *shoot*:

$$Poss(shoot, s) \supset \neg walking(do(shoot, s)). \quad (23)$$

Step 7. From the causation axioms for *dead* and *walking*, we can deduce the following axioms:

$$(\exists s')[s = do(shoot, s') \wedge Poss(shoot, s')] \supset dead(s),$$
$$(\exists s')[s = do(end\text{-}walk, s') \wedge Poss(end\text{-}walk, s')] \supset \neg walking(s),$$
$$(\exists s')[s = do(start\text{-}walk, s') \wedge Poss(start\text{-}walk, s')] \supset walking(s),$$
$$dead(s) \supset \neg walking(s).$$

Consider the first one. This axiom is equivalent to

$$Poss(shoot, s) \supset dead(do(shoot, s)).$$

However, this is not an action qualification axiom since *dead(do(shoot, s))* is not a simple state formula about *s*. So we replace it using its successor state axiom (regression), and obtain:

$$Poss(shoot, s) \supset (shoot = shoot \vee dead(s)).$$

But this yields a vacuous action qualification axiom since the right hand side of the implication is a tautology.

Similar efforts on the second and the third axioms end up in vain as well. So we are left with the fourth one. To see if it entails any implicit action qualifications, we instantiate it to *do(a, s):*

$$dead(do(a, s)) \supset \neg walking(do(a, s)). \quad (24)$$

Now regress both *dead(do(a, s))* and *walking(do(a, $))* according to their respective successor state axioms, we get

$$Poss(a, s) \supset \{[a = shoot \vee dead(s)] \supset$$
$$\neg[a = start\text{-}walk \vee$$
$$walking(s) \wedge \neg dead(s) \wedge a \neq end\text{-}walk \wedge a \neq shoot]\}$$

So we have:

$$Poss(start\text{-}walk, s) \supset \neg dead(s). \quad (25)$$

But this is a non-vacuous qualification axiom on *start-walk*. It can be explained intuitively as follows: If *dead* holds in state *s,* then *start-walk* must not be possible, otherwise, since it does not affect the truth value of *dead* but makes *walking* true, the resulting state will violate the constraint (24). But this constraint is derived from the causal rule (22), so we see here that this causal rule is used to derive both the indirect effect (23) for the action *shoot,* and the implicit qualification (25) on the action *start-walk.*

Now since we have considered all four consequences of the causation axioms, and there are no other constraints in step 4, the correctness of the procedure in the last section guarantees that (25) is the only implicit action qualification axiom. So we obtain the following action precondition axioms using the Clark completion:

$$Poss(start\text{-}walk, s) \equiv \neg walking(s) \wedge \neg dead(s),$$
$$Poss(end\text{-}walk, s) \equiv walking(s),$$
$$Poss(shoot, s).$$

6 Related Work

Much of the work on reasoning about action concerns causality. We shall attempt to review only those that make explicit and formal use of this notion.

Before us, Lifschitz [1987] and Haugh [1987] have also proposed using causality to formalize the effects of actions in the situation calculus. However, they consider only what we called action-triggered causation, so can only represent the direct effects of actions. In fact, both of the causation predicates in [Lifschitz, 1987] and in [Haugh, 1987] take an action but no state argument, and both of the efforts have difficulties handling the ramification problem. Similar remarks apply to [Elkan, 1992] as well despite the fact that the causation predicate there has a state argument. This is because the causation

predicate in [Elkan, 1992] continues to have an action argument, and the state argument is introduced only to help expressing complex preconditions.

Whereas we treat causality as a predicate, Geffner [1990] and McCain and Turner [1995] treat it as modal operators. Nonetheless, it seems that this work and that of [McCain and Turner, 1995] have much in common. The causal theories of [Geffner, 1990], however, are aimed at, general default reasoning. Although [Geffner, 1990] includes an example of how the general framework can be applied to reasoning about action, it is not clear if this can be done in general. In particular, it is not clear if a distinction can be made between ramification and qualification constraints in this framework.

Whereas we use causality as a primitive notion, Shoham [1990], and Iwasaki and Simon [1986] attempt to derive it from an acausal theory. In particular, Iwasaki and Simon consider deriving the causal relations from a set of acausal equations. It is not clear if this approach can be ported into the situation calculus.

Pearl [1988] argues about the need for a primitive notion of causality in general default reasoning. This paper obviously echoes the same theme. In fact, the title of this paper follows that of [Pearl, 1988].

7 Conclusions

We have argued that acausal state constraints like (1) are not adequate for representing the indirect effects of actions, and proposed a solution using causal rules like (9). By embracing causality, we are able to use only simple nonmonotonic formalisms for solving the frame, the ramification, and the qualification problems. This enables us to describe a general class of theories for which our approach is computationally tractable. In fact, we are currently working on a planner that can take as input causal theories of the form specified by steps 1 to 4 in section 4.

Acknowledgements

My thanks to Alvaro del Val, Norman Foo, and Yan Zhang for stimulating discussions related to the subject of this work. This work owes much to Ray Reiter. In fact, it is an outgrowth of our ongoing discussions on the ramification problem and the situation calculus. I also thank G. Neelakantan Kartha, Yves Lesperance, Vladimir Lifschitz, Norman McCain, and Ray Reiter for very helpful comments on earlier versions of this paper.

References

[Clark, 1978] Keith L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logics and Databases,* pages 293-322. Plenum Press, New York, 1978.

[Elkan, 1992] Charles Elkan. Reasoning about action in first-order logic. In *Proc. of the 1992 Canadian Conf. on Artificial Intelligence,* 1992.

[Finger, 1986] Jeff Finger. *Exploiting Constraints in Design Synthesis.* PhD thesis, Department of Computer Science, Stanford University, Stanford, CA, 1986.

[Geffner, 1990] Hector Geffner. Causal theories for nonmonotonic reasoning. In *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90),* pages 524-530, 1990.

[Ginsberg and Smith, 1988] Matthew L. Ginsberg and David E. Smith. Reasoning about action II: the qualification problem. *Artificial Intelligence,* 35:311-342, 1988.

[Haugh, 1987] Brian Haugh. Simple causal minimizations for temporal persistence and projection. In *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87),* pages 218-223, 1987.

[Iwasaki and Simon, 1986] Yumi Iwasaki and H.A. Simon. Causality in device behavior. *Artificial Intelligence,* 29:3-32, 1986.

[Lifschitz, 1987] Vladimir Lifschitz. Formal theories of action. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87),* pages 966-972, 1987.

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

[Lin and Shoham, 1991]
Fangzhen Lin and Yoav Shoham. Provably correct theories of action: Preliminary report. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91),* Anaheim, CA, 1991. Full paper to appear in *JACM.*

[McCain and Turner, 1995] Norman McCain and Hudson Turner. A causal theory of ramifications and qualifications. In *This Volume,* 1995.

[McCarthy and Hayes, 1969] John McCarthy and Patrick 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, 1969.

[McCarthy, 1977] John McCarthy. Epistemological problems of Artificial Intelligence. In *IJCAI-77,* pages 1038-1044. Cambridge, MA, 1977.

[McCarthy, 1986] John McCarthy. Applications of circumscription to formalizing commonsense knowledge. *Artificial Intelligence,* 28:89-118, 1986.

[Pearl, 1988] Judea Pearl. Embracing causality in default reasoning. *Artificial Intelligence,* 35:259-271, 1988.

[Reiter, 1991[1] Raymond Reiter. The frame problem in the situation calculus: a simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy,* pages 418-420. Academic Press, San Diego, CA, 1991.

[Shoham, 1990] Yoav Shoham. Nonmonotonic reasoning and causation. *Cognitive Science,* 14:213-252, 1990.