The Situation Calculus and Event Calculus Compared

Robert Kowalski and Fariba Sadri
Department of Computing, Imperial College,
180 Queen's Gate, London SW7 2BZ
rak@doc.ic.ac.uk fs@doc.ic.ac.uk

ILPS 94

Abstract

The situation calculus and event calculus both formalise common sense reasoning about the initiation and persistence of properties and relationships over the course of time. The main difference is that, whereas the situation calculus deals with transitions between global situations, the original event calculus was designed to deal with the effect of actions on local states of affairs. In this paper, to facilitate comparison, we deal with a special case of the event calculus where, like the situation calculus, actions bring about transitions between global situations.

Another difference is that, whereas the event calculus was intended primarily for reasoning about actual events (narratives), the situation calculus was designed primarily for reasoning about hypothetical actions and situations. In this paper we consider variants of the two calculi which combine the use of deduction to reason about narratives with abduction to reason about hypothetical events.

With these and a number of other minor modifications to the original situation and event calculi, we show that the event calculus logically implies the situation calculus and that the situation calculus augmented with induction logically implies the event calculus. To facilitate the proof, we formulate both calculi as logic programs and use the technique of proving properties of logic programs by reasoning with their completions augmented with induction.

We also explore extensions to the two calculi to deal with ramifications.

1 Introduction

It has long been known that the situation calculus ([20], [21]) can be formulated as a logic program (e.g. [14]). An obvious advantage of such a formulation is that it facilitates implementation. Another is that through the use of negation as failure it provides both an implementation and a variety of sound and useful semantics for default reasoning. In particular, it has been shown that the interpretation of negation as negation as failure solves the Yale shooting problem ([10], [11], [2]).

In this paper we illustrate another advantage of the logic programming formulation, namely that it allows well established techniques for proving properties of logic programs to be applied both to the situation calculus and to the event calculus. One of the most powerful of such techniques and the one we shall use in this paper is to regard the meaning of a logic program as represented by its iff-completion augmented by axioms of induction. This technique was developed by Clark and Tarnlund [5] who observed that the use of this technique for logic programs is analogous to the use of Peano axioms for reasoning about arithmetic. A similar formulation of the situation calculus has recently been developed by Reiter [25].

In contrast to the situation calculus, the event calculus [19] was originally formulated as a logic program, although it was derived from a formulation in standard first-order logic. The original version of the event calculus was very general and allowed both reasoning with maximal periods as well as forward and backward persistence. This version of the event calculus has some similarities [26] with Allen's interval calculus [1].

Later, simplified versions of the event calculus focused on the special case which allows only forward persistence ([9], [27], [17]). Although the original event calculus was intended only for narratives, Eshghi showed that the simplified event calculus could be used for planning by allowing the occurrence of events and the temporal relationships between them to be "abducible". Shanahan showed that abduction could also be used for explanation. Kowalski, on the other hand, discussed an implementation of the simplified event calculus, and informally compared it with the situation calculus. More recently, Pinto and Reiter [24] criticised this comparison and argued that the event calculus is not sufficiently precise to allow a formal comparison with the situation calculus.

In this paper we partly address the Pinto-Reiter criticism by presenting a formal comparison of the situation calculus and the simplified event calculus. For this purpose we formulate the two calculi as logic programs sharing a common vocabulary and common axioms, except for a core axiom in each, which defines what sentences hold in a given situation. In both cases, event descriptions are given explicitly for narratives and are abduced for hypothetical reasoning.

In the following sections we present the logic programming formulations of both calculi, present their "semantics" in terms of iff-completions and axioms of induction, and then show that the situation calculus with induction implies the event calculus, whereas the event calculus without induction implies the situation calculus. We also show how actual and hypothetical events can be combined and how ramifications can be dealt with in both calculi.

2 The situation calculus

The following two clauses constitute the core of our formulation of the situation calculus as a logic program. Together they define what sentences *P* hold in the situation *result(A, S)* that results from performing an action of type *A* in situation *S*.

$$\text{holds}(P, \text{result}(A, S)) \leftarrow \text{happens}(A, S) \wedge \text{initiates}(A, S, P) \qquad \text{S1}$$

$$\mathrm{holds}(P, \mathrm{result}(A, S)) \leftarrow \mathrm{happens}(A, S) \wedge \mathrm{holds}(P, S) \wedge \neg\, \mathrm{terminates}(A, S, P) \qquad \text{S2}$$

Terms starting in the upper case are variables. Unless otherwise indicated, all variables are assumed to be universally quantified in front of the formula in which they occur.

The first clause defines what sentences hold because they are initiated by an action. The second clause, called the *frame axiom*, defines what sentences hold because they held before the action and were not terminated by it.

The predicate *holds(P, S)* can be understood as a metapredicate which expresses that the sentence named *P* holds explicitly (i.e. as an axiom) in the situation named *S*. A situation can be thought of either as a time point or as the set of all sentences, namely the theory, that hold at a given time point. Under this latter interpretation, *holds(P, S)* is a special case of the predicate *demo(S, P)* used to represent the proof predicate in metaprogramming (e.g. [14]).

A definition of *demo* for the simple case of propositional Horn sentences *P* and the relationship of *demo* to *holds* can be given simply by three clauses:

$$\mathrm{demo}(S, P) \leftarrow \mathrm{holds}(P, S) \qquad \text{D1}$$
$$\mathrm{demo}(S, P) \leftarrow \mathrm{demo}(S, P \leftarrow Q) \wedge \mathrm{demo}(S, Q) \qquad \text{D2}$$
$$\mathrm{demo}(S, P \wedge Q) \leftarrow \mathrm{demo}(S, P) \wedge \mathrm{demo}(S, Q) \qquad \text{D3}$$

Of course, the definition of *demo* can be extended to handle more general forms of object level sentences. Of special interest are the additional clauses:

$$\mathrm{demo}(S, \neg P) \leftarrow \neg\, \mathrm{demo}(S, P) \qquad \text{D4}$$
$$\mathrm{demo}(S, P) \leftarrow \mathrm{demo}(S, \forall X\, P) \qquad \text{D5}$$

which we will use later. Clause D5 can be regarded as relating the so-called ground and non-ground representations of variables as discussed in [16].

This interpretation of the relationship between *holds* and *demo* appears to improve upon the conventional formulations of the situation calculus. The most obvious improvement is that it allows actions to initiate and terminate arbitrary sentences. This enables the treatment of such actions as updating a deductive database, or as passing or repealing legislation. The conventional formulations are normally restricted to actions that update atomic sentences, as in "closed world" relational databases, or that update atomic sentences or their negations in "open world" domains. We will show later in the paper that combining actions that initiate or terminate arbitrary sentences with the definition of *demo* in terms of *holds* provides a simple treatment of the *ramification problem.*

The definitions of the predicates *initiates* and *terminates* are domain-dependant. For example, the fact that an act of donor X giving object Y to recipient Z initiates the property of Z possessing Y and terminates the property of X possessing Y can be defined by the two clauses:

initiates(give(X, Y, Z), S, possess(Z, Y))
terminates(give(X, Y, Z), S, possess(X, Y))

Similarly, in the Yale shooting problem, the fact that if the gun is loaded then an act of shooting initiates the death of the turkey and terminates its life can be represented by the conditional clauses:

initiates(shoot, S, dead) ← demo(S, loaded)
terminates(shoot, S, alive) ← demo(S, loaded)

(The use of the conditions *demo(S, loaded)* instead of *holds(loaded, S)* facilitates the treatment of ramifications.)

A possibly more familiar formulation of the situation calculus uses a single predicate *abnormal* ([3], [28]) instead of the two predicates *initiates* and *terminates*. The frame axiom in this formulation is often given in the form:

$$[\text{holds}(P, S) \leftrightarrow \text{holds}(P, \text{result}(A, S))] \leftarrow \neg\text{abnormal}(A, S, P)$$

In our formulation we can derive an analogous formula

$$[\ \text{holds}(P, S) \leftrightarrow \text{holds}(P, \text{result}(A, S))\ ] \leftarrow \text{happens}(A, S) \wedge \neg\text{initiates}(A, S, P) \wedge \neg\text{terminates}(A, S, P)$$

by using the iff-completion of clauses S1 and S2, which is exactly the semantics we use for our formulation of the situation calculus. This semantics is discussed in section 3.

We use the predicates *initiates* and *terminates* instead of *abnormal* because they facilitate the comparison with the event calculus. Moreover, it can be argued that they provide a semantically more meaningful formulation of the situation calculus itself, distinguishing between abnormalities due to initiation of sentences and those due to termination.

The inclusion of the condition *happens(A, S)* in S1 and S2 is a more significant deviation from standard formulations. It is this condition which allows the same axioms, S1 and S2, to be used both for narratives and for hypothetical reasoning. For narratives it suffices simply to add assertions that certain events have actually happened. For hypothetical reasoning it suffices to prove that goals can be achieved or observations can be explained provided that appropriate, relevant events can consistently be assumed to happen. Such assumptions can be determined by means of abduction.

Although our formulation of the situation calculus can be used for planning and for many other kinds of hypothetical reasoning, there is one particular kind, namely counterfactual reasoning, which needs further investigation. Counterfactual reasoning in this case has the form:

"Supposing that an event e1 had happened resulting in outcome p, what would have been the outcome if event e2 had happened instead?"

Some formulations of the situation calculus can reason with such counterfactuals relatively easily, at the expense of not being able to deal with narratives. We conjecture that our formulation could perform this kind of counterfactual reasoning either by employing an appropriate form of abduction or by using metalevel reasoning. Further discussion of this issue is beyond the scope of this paper.

Before accepting an actual or hypothetical event, it is necessary to verify that all its preconditions can hold. This can be done by verifying that the following integrity constraint is maintained:

demo(S, P) ← happens(A, S) ∧ precondition(A, S, P) I1

In the case of narratives the constraint should be verified whenever an update asserts that an event has happened. In the case of hypothetical reasoning it should be checked whenever an event is assumed to happen.

In general, an integrity constraint can be any first-order formula. There are many different views of integrity satisfaction and different techniques for integrity verification. For the main purpose of this paper, namely the theorem of section 5, these differences are not significant, and we shall not discuss them further.

As with the domain-specific predicates *initiates* and *terminates*, the definition of the predicate *precondition* is domain dependant, for example:

precondition(give(X, Y, Z), S, possess(X, Y))

In the standard formulations of the situation calculus, either the preconditions of actions are ignored or they are catered for by including extra conditions in the analogues of clauses S1 and S2. In [18] we showed how a theory with integrity constraints can be transformed into an equivalent theory without constraints by including extra conditions in some of the rules. The proof of equivalence was based on the answer set semantics of Gelfond and Lifschitz [12]. Whether this equivalence can also be demonstrated for the completion semantics, which we use in this paper, has still to be investigated.

In addition to I1, other integrity constraints are needed, including:

A1 = A2 ← happens(A1, S) ∧ happens(A2, S) I2

I2 is necessary to prevent different actions from happening in the same situation. Although the original formulation of the event calculus allows concurrent events, we will also apply I2 to the special case of the event calculus which we investigate later in this paper. This will facilitate our comparison of the situation calculus with the event calculus.

Before we present the semantics of the situation calculus logic program in the next section, we need to discuss the representation of the initial state, s0. There are at least two possibilities. We can define the properties of the initial state by means of a clause

holds(P, s0) ← initially(P)

together with appropriate clauses defining the predicate *initially.*

Alternatively, we may assume, without loss of generality, that nothing holds in the initial state *s0* and that all properties of the initial state are actually properties of the successor of the initial state, *result(creation, s0),* where *creation* is defined by clauses of the form *initiates(creation, s0, P).* For our purpose, the second alternative is more convenient than the first.

Because a *creation* event can initiate any property and has no preconditions, we need an integrity constraint

S = s0 ← happens(creation, S) I3

to prevent abducing the occurrence of *creation* in any situation other than the initial situation. Alternatively, we could simply formulate the relevant *initiates* facts so that *creation* initiates properties only in the initial situation.

3 The semantics of the situation calculus program

Many different semantics have been defined for proving properties of logic programs. To compare the situation and event calculi, it is convenient to use the completion semantics [4] augmented with axioms of induction [5].

The completion of axioms S1 and S2 can be represented in the form

holds(P, S2) ↔
∃A1,S1 [[S2= result(A1, S1) ∧ happens(A1, S1) ∧ initiates(A1, S1, P)] ∨
[S2= result(A1, S1) ∧ happens(A1, S1) ∧ holds(P, S1) ∧
¬ terminates(A1, S1, P)]] SC

together with the Clark equality theory (CET), which consists of the *unique name axioms* including

[A1=A2 ∧ S1=S2] ←result(A1, S1) = result(A2, S2) CET1
¬ ∃ S [s0 = result(A, S)] CET2

and the usual axioms of equality.

In fact, the proof of the theorem in section 5 requires no further unique names axioms additional to CET2.

Similarly, we need to complete the definitions of the *initiates*, *terminates*, *precondition* and *demo* predicates as well as any other non-abducible predicates.

Alternatively, we may "selectively" complete only certain instances of predicates and treat the remaining instances as abducible. This is particularly useful for combining actual and hypothetical events. For example, a theory can contain a completed clause

$happens(X, s0) \leftrightarrow X = load$

to record the actual occurrence of an event, while leaving other instances of the predicate abducible.

We will use the technique of selective completion later in sections 6 and 7. Another approach [7], which achieves similar results is to treat all instances of incompletely defined predicates as abducible and to represent information about incomplete predicates by means of integrity constraints.

In all of these approaches, the fact that a predicate or an instance of a predicate is abducible is represented by excluding it from the completion and thereby leaving it undefined. Console et al [6] have shown that, with this representation, abduction using the if-form of a logic program becomes deduction using the iff-completion.

One of the limitations of most formulations of the situation calculus is that situations need to be named explicitly in terms of a sequence of actions following the initial state. As a result, it is virtually impossible to represent narratives with incomplete information. This limitation can be overcome by relating situations to one another by means of an ordering relation. Such ordering relations have been defined for the situation calculus by Shanahan [28] and Reiter [25].

$S < result(A, S) \leftarrow happens(A, S)$
$S < result(A1, S1) \leftarrow happens(A1, S1) \wedge S < S1$

The "semantics" of this program is given by its completion:

$S1 < S2 \leftrightarrow \exists A,S\ [\ S2=result(A, S) \wedge happens(A, S) \wedge S1 \leq S\ ]$ LESS

where $S1 \leq S$ abbreviates $S1=S \vee S1 < S$.

Here the condition *happens(A, S)* ensures that the only situations considered are those that result from actions that actually happen in the case of narratives or that can happen in the case of hypothetical actions.

To ensure that situations have the intended form of a finite sequence of actions following the initial state, we need the further integrity constraint

$s0 \leq S \leftarrow happens(A, S)$ I4

Here (and elsewhere) the expression $s0 \leq S$ functions as a type expression, stating that S is of the type "situation".

Induction can now be expressed in the form

$\forall S\ [\Phi(S) \leftarrow s0 \leq S] \leftarrow \Phi(s0) \wedge \forall A,S\ [\Phi(result(A, S)) \leftarrow \Phi(S) \wedge s0 \leq S]$ IND

Φ is either a metavariable standing for any first-order formula (as in Peano arithmetic) or a universally quantified second-order variable (as in complete axiomatisations of arithmetic). In practice, the metavariable version is sufficient for our purposes.

Similarly to ([24], [25]), given LESS, CET2 and IND we can show the following properties of the predicate < . We need most of these in the proof of the main theorem in section 5.

a) $\neg \exists S\ [S < s0]$
b) $\forall S1,S2,S3\ [S1 < S3 \leftarrow S1 < S2 \wedge S2 < S3]$
c) $\forall S1,S2\ [\neg S2 \leq S1 \leftarrow S1 < S2]$
d) $\forall S,A\ \neg \exists S1\ [S < S1 \wedge S1 < result(A,S)]$
e) $\forall S\ \neg S < S$

4 The simplified event calculus

The original version of the event calculus [19] allowed concurrent and partially ordered events as well as the persistence of properties both backwards and forwards over time. It also catered for properties holding both during time periods and at time points.

To compare the event calculus with the situation calculus we shall consider a special case of the former that is analogous to the restrictions of the latter. We can obtain this special case by pre-processing away the time periods and removing any consideration of backward persistence of properties. Such a special case was introduced in ([15], [17], [27]). This version of the event calculus has at its core a single axiom:

$$\begin{aligned} holds(P, T2) \leftarrow\ & happens(E1, T1) \wedge initiates(E1, P) \wedge T1 < T2 \wedge \\ & \neg \exists E^*,T^*[happens(E^*, T^*) \wedge terminates(E^*, P) \wedge T1 < T^* \wedge T^* < T2] \end{aligned}$$

which expresses that a property holds at a time point if it was initiated by an event which happened at an earlier time point and no intervening event happened to terminate the property.

The condition $T1 < T2$ has the effect of ensuring that properties hold only *after* their initiating event happens. The inequality $T^* < T2$ ensures that properties , including preconditions of events, hold at the time they are terminated by some event. The inequality $T1 < T^*$, instead of the stronger inequality $T1 \leq T^*$, builds in the assumptions that two different events do not occur at the same time (expressed by I2), and that no event initiates and terminates the same property.

We further restrict our version of the event calculus to the special case where the < relation on time points is identified with the < relation on situations. In this case, event tokens are unnecessary because individual events can be identified by their action or event types and the situations in which they occur. We may, therefore, instantiate the event calculus axiom by means of a particular representation for events, representing event tokens E by pairs (A, S) consisting of an action or event type A and a situation S. This reformulates the event calculus predicates in the situation calculus vocabulary, replacing

happens(E,T)	by	happens(A, S, S)
	which simplifies to	happens(A, S)
initiates(E, P)	by	initiates(A, S, P)
terminates(E, P)	by	terminates(A, S, P)

The resulting completion of the event calculus axiom has the iff-form:

$$\begin{aligned}&\text{holds}(P, S2) \leftrightarrow\\ &\exists A1,S1 \; \{ \text{ happens}(A1, S1) \wedge \text{initiates}(A1, S1, P) \wedge S1 < S2 \wedge\\ &\qquad \neg \exists A^*,S^* \; [\text{ happens}(A^*, S^*) \wedge \text{ terminates}(A^*, S^*, P) \wedge S1 < S^* \wedge S^* < S2 \;] \; \} \qquad \text{EC}\end{aligned}$$

Like our version of the situation calculus, this formulation of the simplified event calculus can be used either for narratives by letting *happens* assertions be given as input, or for hypothetical reasoning by letting them be generated by abduction. In both cases, integrity of the happening of events can be checked by verifying that such assertions satisfy the integrity constraints I1-I4.

Our formulations of the situation calculus and of the simplified event calculus differ only in the core axioms SC and EC. The remaining axioms, IND, LESS, CET, D1-D5 and constraints I1-I4 are common to both calculi. Although the axiom EC itself is noncommittal with respect to the notion of time (which could be continuous or discrete), the axiom LESS identifies time with the discrete situations of the situation calculus.

5 The relationship between the situation and event calculi

The relationship between our formulations of the situation and event calculi can now be stated as follows.

Theorem: i) COMM, EC |= SC
ii) COMM, SC, IND |= EC

where SC and EC are the iff-forms of the situation calculus and event calculus core axioms respectively, IND is the axiom schema of induction and COMM consists of I2, I4, LESS and the properties (a)-(d) common to the two calculi.

Proof:

i) Given COMM, EC is equivalent to

$\text{holds}(P, S2) \leftrightarrow (\text{event1}) \lor (\text{event2})$

where (event1) has the form

$\exists A1,S1,A,S$
$\{ \text{happens}(A, S) \land \text{happens}(A1, S1) \land S2=\text{result}(A, S) \land$
$\text{initiates}(A1, S1, P) \land S1=S \land$
$\neg \exists A^*,S^* [\text{happens}(A^*, S^*) \land \text{terminates}(A^*, S^*, P) \land S1< S^* \land S^*< S2] \}$

and (event2) has the form

$\exists A1,S1,A,S$
$\{ \text{happens}(A, S) \land \text{happens}(A1, S1) \land S2=\text{result}(A, S) \land$
$\text{initiates}(A1, S1, P) \land S1< S \land$
$\neg \exists A^*,S^* [\text{happens}(A^*, S^*) \land \text{terminates}(A^*, S^*, P) \land S1< S^* \land S^*< S2] \}$

This equivalence can be shown by replacing the condition S1< S2 in EC by its definition in LESS and distributing disjunction over conjunction.

On the other hand, SC is equivalent to

$\text{holds}(P, S2) \leftrightarrow (\text{sit1}) \lor (\text{sit2})$

where (sit1) has the form

$\exists A,S [S2 = \text{result}(A, S) \land \text{happens}(A, S) \land \text{initiates}(A, S, P)]$

and (sit2) has the form

$\exists A,S [S2 = \text{result}(A, S) \land \text{happens}(A, S) \land \text{holds}(P, S) \land \neg \text{terminates}(A, S, P)]$

It is not difficult to show that (event1) and (sit1) are equivalent in the theory COMM using lemma (d), stating that there is no situation S* between S and result(A, S), and the assumption I2, that only one event occurs in any situation. Also (event2) is equivalent to

$\exists A1,S1,A,S$
$\{ \text{happens}(A, S) \land \text{happens}(A1, S1) \land S2=\text{result}(A, S) \land$
$\text{initiates}(A1, S1, P) \land S1< S \land \neg \text{terminates}(A, S, P) \land$
$\neg\exists A^*,S^* [\text{happens}(A^*, S^*) \land \text{terminates}(A^*, S^*, P) \land S1< S^* \land S^*< S] \}$