Electronic News Journal on Reasoning about Actions and Change 91
Vol. 1: *91*, 31.10.1997
http://www.ida.liu.se/ext/etai/rac/notes/1997/08/

Position Statement for NRAC Panel on Ontologies for Actions and Change

Erik Sandewall
Linköping University, Sweden

The following is my idea of the topic for the panel:

By an "ontology" for actions and change, I mean a set of assumptions about the character of the world that one is reasoning about. For example, the choice of discrete vs continuous time, the choice to allow or not to allow for causation relations *between* events, and the choice to allow or not to allow for nondeterminism, are examples of such assumptions which together form an ontology.

It may be useful to distinguish between ontological and epistemological assumptions, where the latter are assumptions about *what we know* about the world. "All the actions/events are explicitly known" is an example of such an epistemological assumption.

Ontologies may be expressed formally or informally. I propose that the panel should focus on formally expressed ontologies.

One consequence of the definition is that the "frame assumption" or assumption of persistence must be built into the ontology. The situation calculus then does not represent an ontology, since commonsense scenario descriptions in sitcalc need to be complemented with additional axioms, minimization of models, or other similar devices.

The main workshop invitation mentions two families of ontologies, namely those represented by action languages (cal-A and successors) and by the features and fluents framework (that is, trajectory semantics and the approach of using underlying semantics). Ray has pointed out to me that GOLOG also represents an ontology that differs from the first two in important respects.

If you agree with me about this background, at least in its main parts, I propose that you might address the following topics (but not exhaustively!) in your introductory statements at the panel:

1) What ontologies = sets of ontological assumptions are in use at present?

2) How are they expressed formally?

3) What results have been obtained within and between those ontologies? What types of results are likely to be obtained in the near future?

Electronic News Journal on Reasoning about Actions and Change 92
Vol. 1: *92-95*, 31.10.1997
http://www.ida.liu.se/ext/etai/rac/notes/1997/09/

The Ontological Approach of Situation Calculus and GOLOG

Ray Reiter
University of Toronto, Canada

I like Erik's proposal for lots of reasons, but mostly because he wants to keep the discussion technical. No vague claims, and amen to that.

Here's what I want to say:

1. Erik's notion of an ontology seems odd to me, mainly because it requires "that the "frame assumption" or assumption of persistence must be built into the ontology". I would have thought that the frame assumption is epistemological or, as the philosophers like to say, "metaphysical". My own understanding of "ontology" is that it is determined by the *language* one uses in formulating one's theories. In any case, I think that Erik *is* making an important foundational point, namely, that there are two rather different ways to address the frame problem, one more fundamental than the other:

a. Solve it axiomatically by including in one's domain axioms suitable sentences capturing the frame assumption. Whether or not these axioms are parsimonious, or how one arrives at them, is largely irrelevant. The problem is seen simply as writing down an intuitively correct collection of axioms. This is the "classical" approach in AI. It seems to be what McCarthy and Hayes had in mind in their original formulation of the problem. In other words, this is the axiom-hacking approach. I admit to being guilty of this sin in almost all my past work on actions.

b. The second approach – which I believe Erik is advocating – is much more principled and fundamental. It requires that one's ontological assumptions (in Erik's use of the term) be formalized *semantically*, i.e. as a class of structures in the logician's sense of that word. Of course, this must be a class of structures for some logical *language*. So one's ontological assumptions (in my sense of the term) have first to be expressed in a choice of language, but that having been done, one can then define the class of structures that capture one's intuitions about persistence. Alas, a lot more remains to be done after this. Next, you have to figure out what *sentences* of the language characterize the above class of structures, and finally, prove a representation theorem stating that the models of these sentences are all and only the structures in the class. I take

it that much of Erik's work is of this kind, as is also the work on the $\mathcal{A}$-families of languages of Gelfond and Lifschitz. Circumscriptive approaches seem to lie somewhat between the axiom-hacking and semantic methodologies.

I have no quarrel with the second approach. I think that methodologically, it's the right way to go. However, most of my work, and that of my colleagues at Toronto, is motivated by quite different considerations, namely, given a (perhaps not completely general, perhaps not methodologically solid) solution to the frame problem, what can we do with it? We have been very busy answering this question during the past few years, and this has led to the GOLOG family of programming languages, as well as various extensions of the sitcalc ontology and of our solution to the frame problem to accommodate this extended ontology.

Which brings me to:

2. The extended ontology of the sitcalc for which the basic solution to the FP is sufficient.

- Concurrency.
- Time (discrete, continuous, linear, circular, whatever you want).
- Natural actions (e.g. falling objects, balls colliding, bus schedules).
- Continuous actions.
- Sensing actions and knowledge.
- Complex actions, programs, concurrent programs, interrupts, reactive behavior (GOLOG, Temporal GOLOG, RGOLOG, CONGOLOG).

I think that Erik is right in focusing on ontologies in this panel, so let me say a little bit about the sitcalc ontology, how it differs from other approaches to actions, and why these differences matter.

3. The central ontological ingredient of the sitcalc is the *situation*. Even at this late stage in AI, many people still don't understand what a situation is, so here's the secret: A situation is a finite sequence of actions. Period. It's not a state, it's not a snapshot, it's a *history*. Moreover, situations are first class objects in the sitcalc – you can quantify over them.

These features have lots of consequences:

(a) Planning is done deductively, not abductively as in linear time logics like the event calculus or the features and fluents approach.

(b) Because they are just action sequences, plans are situations; they are terms *in* the language and can therefore be inspected by suitable predicates and reasoned about. Our experience has been

that this is an essential feature of the sitcalc. See Fangzhen Lin's paper at this IJCAI for an elaboration and application of this idea.

(c) The GOLOG family of languages depends crucially on the fact that histories are first class objects in the sitcalc. The result of executing a GOLOG program is a situation representing its execution trace.

(d) The space of situations is the set of all finite sequences and therefore it is a tree rooted at [], the empty sequence. This means that the sitcalc provides branching futures. In addition, the sitcalc ontology includes a predicate for subsequence. This, together with the ability to quantify over situations means that one can express almost all the modalities that temporal logics provide like in (some, all) futures, past, next, etc.

(e) Since it supports branching futures, the sitcalc is well suited to hypothetical and counterfactual reasoning.

(f) Because situations are terms, they can function as surrogates for the possible worlds much beloved of modal logicians. This means, as Bob Moore showed years ago, and as Hector Levesque has elaborated, we can axiomatize accessibility relations on situations, and embed logics of knowledge directly into the sitcalc. As John Mccarthy likes to put it: Modalities si, modal logic no! Using this, Levesque has formulated an elegant treatment of sensing actions, knowledge and a solution to the frame problem for knowledge within the sitcalc.

4. Relationship of the sitcalc to other ontologies. (I'm flying a bit by the seat of my pants here. Corrections welcome.)

Situtaion calculus	A-languages	Linear temporal approaches
actions are terms	same?	same
histories are first class citizens	state-based no histories	no histories
branching futures	branching	linear
first order logic	propositional	first order
supports sensing actions and knowledge without explicit modalities	possible, but not yet done.	not likely

5. Finally, I'd like to say a few words about relationships to another approach to dynamical systems, namely classical discrete event control theory.

The central component of DECT is an automaton, whose transitions are defined by actions, and whose states are what we normally think of as world states, i.e. tuples of fluent truth values. The work on $\mathcal{A}$-languages comes very close to this view semantically and one can view this work as the logicization of DECT. There are lots of advantages to this logicization, not least, that sentences in a language provide a compact representation for the exponentially large state spaces that control theorists have to deal with. Also, sentences allow for incomplete information about the initial state, a serious and difficult problem for control theory. While this connection to DECT is pretty direct for the sitcalc and $\mathcal{A}$-languages, it's not so straightforward for the linear temporal logics. I think the sitcalc has lots of advantages for establishing these connections:

(a) It's first order and therefore generalizes the essentially propositional automata of DECT. (DECT can be interpreted as a version of the monadic sitcalc.)

(b) The family of GOLOG languages can be used to write controllers.

(c) Because it's all in logic, one can prove properties of these controllers (safety, fairness, etc).

(d) With its expanded ontology for continuous actions and time, the sitcalc is suitable for modeling and controlling so-called "hybrid" systems, a hot topic these days in the control theory world.

Electronic News Journal on Reasoning about Actions and Change 96
Vol. 1: *96-98*, 31.10.1997
http://www.ida.liu.se/ext/etai/rac/notes/1997/10/

Action Languages from $\mathcal{A}$ to $\mathcal{C}$: A Statement for the Panel on Ontologies

Vladimir Lifschitz
University of Texas at Austin, TX, USA

The ontology of the original action language $\mathcal{A}$ [1] is more restrictive than the ontological assumptions of its "dialects" proposed later. In this note, I compare $\mathcal{A}$ with the recent proposal called $\mathcal{C}$ [2]. (There is no discussion here of other work on action languages—that would require a long paper.)

The language $\mathcal{A}$, as well as some of its successors, has two components. Its "effect propositions" describe the effects of actions on fluents, and their meaning can be described by a transition diagram similar to the diagrams familiar from the theory of finite automata. They form the "action description part" of $\mathcal{A}$. The "value propositions" of $\mathcal{A}$ are conditions on a path in this transition diagram. They form the "query part" of $\mathcal{A}$. The task of designing the query component of an action language is pretty much orthogonal to the task of designing its description component [3], and it is convenient to address these two problems separately. In accordance with this idea, $\mathcal{C}$ was designed as a pure *description* language; there is no counterpart of value propositions in it. The semantics of $\mathcal{C}$ is defined by showing how sets of propositions describe transition diagrams.

Here are some differences between the ontology of $\mathcal{C}$ and the ontology of the effect propositions of $\mathcal{A}$.

1. $\mathcal{A}$ is based on propositional logic; $\mathcal{C}$ uses full classical logic (even higher-order, if we wish). Accordingly, the ontology of $\mathcal{C}$ includes *families* of actions and fluents; they are represented by symbols with arguments. Furthermore, fluents can be non-Boolean; then they are represented by terms rather than formulas. We do not have to represent fluents by expressions like *Block5IsOnTable*, as in $\mathcal{A}$; instead, we can write $On(Block5, Table)$ or $Location(Block5) = Table$. We can also use variables and write expressions like $Location(x) = l$.

2. In $\mathcal{A}$, every action is executable in every state, and in exactly one way. In $\mathcal{C}$, actions can be nondeterministic. A transition diagram described in $\mathcal{C}$ looks like a transition diagram of a nondeterministic finite automaton: It can have several edges beginning in the same state and having the same action label. This set of edges can be

empty, which means that the action is impossible to execute in the given state.

3. There is no way to talk in $\mathcal{A}$ about the concurrent execution of actions. In $\mathcal{C}$, actions can be performed concurrently. Accordingly, an edge in a transition diagram is labeled by a *set* of actions, rather than a single action. In particular, this set of actions can be empty—the world can change even when we do nothing. In this way, we can describe causal relationships between states, as opposed to causal relationships between an action and a state. (Having \$100 in my bank account today causes the balance to become \$100.01 tomorrow even when I do nothing.) Syntactically, this is achieved by using *predicate* symbols to represent (families of) actions, instead of function symbols as in the situation calculus. The formula

$$Move(Block5, Table) \wedge \neg\exists l Move(Block6, l)$$

expresses that *Block5* is moved onto the table while *Block6* is not moved anywhere.

4. In $\mathcal{A}$, every fluent that has a name in the language is "inertial," that is to say, tends to keep the value that it had before. In $\mathcal{C}$, the assumption that a fluent is inertial can be easily expressed, but it is not "built-in." Some fluents are not inertial. For instance, the balance of my bank account tends to change in accordance with a certain formula. This is similar to the inertia assumption, but not quite the same. Whether a string on my guitar is producing a sound is not an inertial property either. It is "momentary," in the sense that it tends to have the value false.

Syntactically, the main feature of $\mathcal{C}$ is the use of "dynamic causal laws" of the form

caused F **if** G **after** H

("there is a cause for F to be true if G is true and H was true before"). The semantics of these expressions is defined by a translation into classical logic and is, in this sense, similar to circumscription. Its main idea comes from [4]. Here are some examples:

1. The effect of *Move* on *Location* can be expressed by

caused $Location(x) = l$ **after** $Move(x, l)$

which is shorthand for

caused $Location(x) = l$ **if** $True$ **after** $Move(x, l)$.

2. The impossibility of moving a heavy object can be expressed by

nonexecutable $Move(x, l)$ **if** $Heavy(x)$

which is shorthand for

$$\textbf{caused } \mathit{False} \textbf{ if } \mathit{True} \textbf{ after } \mathit{Move}(x,l) \wedge \mathit{Heavy}(x).$$

3. The inertiality of *Location* can be expressed by

$$\textbf{inertial } \mathit{Location}(x) = l$$

which is shorthand for

$$\textbf{caused } \mathit{Location}(x) = l \textbf{ if } \mathit{Location}(x) = l \textbf{ after } \mathit{Location}(x) = l$$

("there is a cause for x to be at location l if x is there now and was there before.").

Two ontological ideas that I do not know how to express in $\mathcal{C}$ are

- continuous time, and
- causal relations between actions.

Do they require that we go beyond the framework of transition diagrams?

References

1. Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17:301–322, 1993.

2. Enrico Giunchiglia and Vladimir Lifschitz. An action language based on causal logic. Unpublished draft. Available at `http://www.cs.utexas.edu/users/vl/mypapers/c-short.ps`, 1997.

3. Vladimir Lifschitz. Two components of an action language. *Annals of Mathematics and Artificial Intelligence*, 1997. To appear.

4. Norman McCain and Hudson Turner. Causal theories of action and change. In *Proc. AAAI-97*, pages 460–465, 1997.

Electronic News Journal on Reasoning about Actions and Change 99
Vol. 1: *99-115*, 30.9.1997
http://www.ida.liu.se/ext/etai/actions/nj/9710-2/

Protocol of Panel Debate About Ontologies for Actions and Change

Edited by: Erik Sandewall
Linköping University, Sweden

This on-line panel debate is a continuation of the workshop panel on Ontologies for Actions and Change that was held at the NRAC workshop during the IJCAI 1997 conferece, and chaired by Erik Sandewall. The online session started with position statements by the three panelists, namely Erik Sandewall, Ray Reiter, and Vladimir Lifschitz. (In the session at NRAC, Michael Thielscher participated instead of Vladimir Lifschitz). These position statements are included in the present News Journal. Earlier position statements in an ECSTER on-line discussion (reproduced in the May, 1997 issue of this News Journal) are also relevant for the present topic.

The subsequent discussion up to the end of the month was as follows. Some of the contribution were received as part of the concurrent panel on theory evaluation, but were transferred to the present discussion. The discussion is expected to continue into November; continued contributions will then be included in next month's News Journal.

Murray Shanahan on 23.10.1997

I'm sympathetic with most of Pat Hayes's criticisms of the situation calculus, but not when he writes ...

> *Why is it that the only people who feel at all bothered by the frame/ ramification/ qualification problems are philosophers (who mostly dont even understand what they are) and people working in this rather isolated part of KR?*

The frame problem seems to arise in any logic-based formalism in which the effects of actions are described. It certainly arises in the event calculus, which has a very different ontology to the situation calculus. It also arises in the ontologies of Yoav Shoham's and Erik Sandewall's books, which is why those book took the shape they have. The YSP, in some guise, arises in all these formalisms too. And (at the risk of reviving an old debate Pat had with Drew McDermott),

the frame problem seems to arise in Pat Hayes's histories formalism too.

Ray Reiter on 23.10.1997

When Pat Hayes speaks, one is well advised to listen, because he usually gets it right. But when the godfather of the sitcalc, and a parent of the frame problem says such surprising things about his own creations, I can't restrain myself.

> *First, its based on an overly simplistic view of the way things happen in the everyday world, one obviously inspired by reasoning about what happens inside computers. The everyday world just doesnt consist of static states and functions between them: its not organised like a series of snapshots. Sitcalc belongs with SHAKEY, in a world where only the robot can move and nothing else is happening.*

False. Situations are simply finite sequences of actions. These need not be just the actions under the robot's control; they can, and in interesting cases do, involve exogenous actions (Fido ate the sandwich that the robot is asked to fetch.) Writing controllers to deal correctly with such exogenous event occurrences has long been the meat-and-potatoes of control theory, and this is certainly possible also in the sitcalc. Indeed, the sitcalc can easily be seen to be a generalization of discrete event control theory.

> *Second, sitcalc only works properly if we are careful only to mention processes which can be acted upon; that is, it confuses change with action.*

I can't figure out what Pat means by this, even with the help of his grow(s) example. I suspect that he wants to distinguish between processes, that evolve in time, and actions, but I'm not sure. So I'll simply say here that there is a sitcalc story for continuous processes, and leave it at that.

> *Third, it confuses action with inference. The way that actions are described in the sitcalc involves asserting conditions on the past and inferring conclusions about the future: axioms have the general form ...(s) =...(action(s)). But common-sense reasoning often involves reasoning from the present to the past (as when we infer an explanation of something we see) or more generally, can move around in time quite freely, or may have nothing particularly to do with time or action. We are able not just to say that if the trigger is pulled then the target will be dead, but also, given the corpse, that someone must have pulled*

the trigger. In the sitcalc this would require giving necessary and sufficient conditions for every action description, and Reiter's recent attempt to rejuvenate it does.

So what's the point here? With a suitable solution to the frame problem, one can, in the sitcalc, reason in all directions.

Which brings us to the fourth thing wrong with sitcalc: it has many fatal, or at any rate very intractable, technical problems. Why is it that the only people who feel at all bothered by the frame/ramification/qualification problems are philosophers (who mostly dont even understand what they are) and people working in this rather isolated part of KR? Why hasnt the FP become a central difficulty in, say, natural language work, or qualitative physics, or planning (as used in industrial applications)? Because those fields typically dont use this clumsy ontology, that's why. These problems are all artifacts of the sitcalc ...

Absolutely false! I can't speak to how the natural language community treats actions, but qualitative physics and planning have no difficulty with the FP because, without exception, they adopt the STRIPS sleeping dogs strategy. Which is to say, **they assume they have complete information about world states**. If you don't believe that, here's a challenge. Give a theory of planning for incompletely specified worlds, in any formalism you like, that does not require a solution to the FP.

Now sleeping dogs are great, when applicable. But robots must necessarily function in incompletely specified worlds; otherwise, why do they need sensors? In the absence of a good story of how to reason about the effects of actions in open worlds without solving the FP, I'll put my money on the Lifschitzs, Sandewalls, Shanahans and Thielschers of our community.

Mikhail Soutchanski on 27.10.1997

The following reply addresses comments by P. Hayes, V. Lifschitz, R. Miller and others against the situation calculus.
Pat wrote:

"Bad theory" [the sit. calc.] isnt really right: it was a really neat theory for a while, and better than anything going, and its still useful. But it has some pretty dreadful properties; and yet not only has it lasted a long time, but its almost considered to be inviolable by many people in the field.

There are reasons why the situation calculus (SC) has been successful for so long time. Here are some of them.

1. The SC is simple and easy to understand. It is formulated in the classical many-sorted first-order (or second-order, if necessary) logic with the standard semantics. I want to stress here the difference between the classical logic approach and other logics (with non-standard syntax and semantics) proposed for formalization of reasoning about actions. If at a later time, somebody will propose a new (non-standard) logic for reasoning, say, about shapes, and somebody else will propose yet another (non-standard) logic, e.g., for reasoning about materials (or substances), it would be a difficult task to combine all those proposals in a one logical framework.

2. The situation calculus is a foundation for general purpose high-level programming languages. Reminder: This idea is proposed in the 1969 paper "Some philosophical problems from the standpoint of artificial intelligence" (J.McCarthy & P.Hayes). Note that it is an easy exercise to formalize the Turing machine in the SC.

Moreover, thanks to the explicit situational argument, as long as the SC-based program proceeds, the information about the sequence of actions performed so far, can be used to direct the further execution of a program. For example, if (in the real world) during the execution of a primitive action robot 'fails', analyzing the list of primitive actions performed so far, the robot can (sometimes) infer conclusions regarding what caused the failure. As we know from the control theory and the game theory, the history of the interaction of an agent with an environment (that may include other agents with possibly contradictory goals) may provide useful guidelines when the agent decides how to recover from a 'failure'. From the other hand, the event calculus and other "narrative time-line languages" do not have any term that would keep record of what part of the narrative had been done before the moment when a failure happened.
Pat continued:

> *here are a few of the things that are wrong with sitcalc. First, its based on an overly simplistic view of the way things happen in the everyday world, one obviously inspired by reasoning about what happens inside computers. The everyday world just doesnt consist of static states and functions between them: its not organised like a series of snapshots.*

A. We should distinguish between situations (which are uniquely associated with sequences of actions) and snapshots (which are equivalence classes over situations). B. The SC of 1997 can handle very sophisticated views of the way things happen in the everyday world.

> *Most intuitive reasoning done by humans lies entirely outside the purview of the situation calculus.*

Note that your objection can be easily rephrased as: "Most intuitive reasoning done by humans lies entirely outside the purview of

the formal logic".

I'm not sure whether we must have the same concerns that the cognitive science has. Most of the people do not think in terms of C, LISP, PROLOG, but all these languages are still useful for writing programs that will exhibit an intended behavior. Similarly, the SC is useful as the basis for the high-level programming language.

> *Yet so firm has been the grip of the sitcalc ontology on people's thinking that examples which do not immediately fit into it are routinely ignored,*

Please, formulate those examples in technical terms.
Murray wrote:

> *I'm sympathetic with most of Pat Hayes's criticisms of the situation calculus*

Erik wrote:

> *With respect to your second point, concerning the situation calculus as an example of a theory with staying power but considerable weaknesses, exactly those observations have led to the work on reasoning about actions using first-order logic with explicit metric time [...] We can certainly discuss whether the shortcomings in the basic sitcalc can be fixed by add-ons, or whether a metric-time approach is more fruitful, and this discussion is likely to go on for a while (see also Ray Reiter's comments, next contribution). However, since we agree about the shortcomings of sitcalc, it might also be interesting to discuss why* it *has such remarkable inertia.*

Please provide formal arguments why the SC of 1997 cannot be used for high-level programming of robots and for providing operational semantics of programming languages and explain what frameworks will work better.

The following is a reply to Rob Miller's note [1]["Comparing Action Formalisms: A Preliminary Position Statement"]

> *A good example of a (nevertheless interesting) problem which is the product of a particular ontology (rather than being fundamental) is the difficulty of distinguishing between observations and causal rules in the Situation Calculus [...] Neither the problem nor the solution translate to other (ontologically different) approaches. We need to be careful to distinguish between this type of issue and more fundamental problems such as dealing with ramifications or continuous change.*

[1] Ref: http://vir.liu.se/brs/news/96deb/03/debit.html

In the 1997 version of the SC, there are *no* causal rules. Toronto's version of the SC has instead of them successor state axioms specifying the evolution of a dynamical system (for example, composed from robot, other agents and the nature) and precondition axioms which specify when primitive actions are possible. Let's understand "observation" as a SitCalc formula that contains occurrences of only one (current) situational term. There are no problems with any observation as long as observations and robot's beliefs about the world (deduced from an initial description by successor-state axioms) coincide with each other. If they do not, it means only that an exogenous (with respect to robot's mind) action changed value of one or several fluents. However, there is a straightforward and technically sound approach to incorporate "unexpected" observations using successor state axioms. Note that event calculus will have exactly the same problem if the robot believes regarding a fluent f that it was *InitialisedTrue(f)* and was not *Clipped(0,f,t)* at the moment t, but nevertheless, a sensor reports that this fluent does not hold at t (due to some external reasons).

The following is a reply to "Approaches to Reasoning About Actions: A Position Statement" by Vladimir Lifschitz.
Vladimir wrote:

> *1. Explicit time vs. the situation calculus. The following situation calculus formula seems to have no counterpart in languages with explicit time:*
>
> ```
> value(f,result(a1,s)) = value(f,result(a2,s)).
>                                                  (1)
> ```
>
> *It says that the value of f at the next instant of time does not depend on which of the actions a1, a2 is going to be executed. For instance, if I now send an e-mail message to Erik Sandewall, the total number of messages sent by me since this morning will be the same as if I send a message to Ray Reiter instead. This is an argument in favor of the situation calculus.*
>
> *But there is a little problem here. What is the meaning of (1) if the effects of a1 and a2 on f are nondeterministic? I have a few coins in my pockets; let a1 stand for getting a coin from my left pocket, let a2 stand for getting a coin from my right pocket, and let f stand for the value of the coin that I have in my hand. We can interpret (1) as a counterfactual, but this seems less interesting than assertions involving some kind of quantification over the outcomes of a1 and a2, for instance:*
>
> - *there exist an outcome of a1 and an outcome of a2 such that (1) holds,*

- *for any outcome of a1 and any outcome of a2, (1) holds,*
- *for any outcome of a1 there exists an outcome of a2 such that (1) holds.*

The situation calculus has no mechanism for expressing these distinctions.

1). Consider nondeterministic actions as concurrent executions of two actions: one action is performed by an agent (like a1 and a2 in the example above), another action is performed by the nature. These concurrent executions seem nondeterministic for the agent (or any other external observer) only because there is no information what particular action is selected by the nature. Thus, we distinguish two separate activities: Vladimir extracts an object from a pocket and nature makes this object into the coin of the particular value. Let n1 be nature's action of turning a coin from the left pocket into the coin of the particular value, n2 - the corresponding action for the right pocket. Consider now new sort "c" for sets of actions performed concurrently. Let constants C1 and C2 represent activities in corresponding pockets, then the formula

```
IN(a1,C1) & IN(n1,C1)
```

says that a1 - a physical action performed by Vladimir is included in "C1" and n1 - action chosen by nature is also included in "C1". Similarly,

```
IN(a2,C2) & IN(n2,C2)
```

represents a concurrent activity (a2 and n2) in the right pocket. Assuming some additional axioms like unique name axioms and like

```
\forall a. IN(a,C1) <=> a=a1 or a=n1
\forall a'. IN(a',C2) <=> a'=a2 or a'=n2
```

the formula (1) can be rewritten as:

```
IN(a1,C1) & IN(n1,C1) & IN(a2,C2) & IN(n2,C2) &
   [value(f,res(C1,s))= value(f,res(C2,s))]
```

I will denote the resulting formula by "Formula(a1,n1,a2,n2,s)". The assertions involving some kind of quantification over the outcomes are represented in the following way:

```
(i)     \exists n1, n2. Formula(a1,n1,a2,n2,s)
(ii)    \forall n1, n2. Formula(a1,n1,a2,n2,s)
(iii)   \forall n1, exists n2. Formula(a1,n1,a2,n2,s)
```

2. (by R.Reiter)

Instead of the function *result(a,s)* consider the relation *do(a,s,s')*: do(a,s,s') means that s' is one of the situations you can reach from s by performing action a. It's just like Golog's *do(δ,s,s')*. Then we can represent Vladimir's three distinctions by:

```
(i)     (\exists s',s''). do(a1,s,s') & do(a2,s,s'') &
                value(f,s')=value(f,s'').
(ii)    (\forall s',s''). do(a1,s,s') & do(a2,s,s'') ->
                value(f,s')=value(f,s'').
(iii)   (\forall s'). do(a1,s,s') ->
             (\exists s''). do(a2,s,s'') &
                            value(f,s')=value(f,s'').
```

Murray Shanahan on 28.10.1997

Stop! Enough! This discussion has quickly degenerated into childish bickering. There is little value in a debate of the form:

A: You can't do X in B's formalism.

B: Yes you can. But you can't do Y in A's formalism.

A: Yes you can. But you can't do Z in B's formalism.

and so on. . . No doubt, suitably modified, you can do whatever you need to in any of the formalisms. (Why does Ray write "sensing actions in the event calculus: not likely"? Rob Miller has work in progress on this theme. History should tell us that such claims are dangerous. A few years ago we were saying "continuous change in the situation calculus: not likely".)

Why this possessivenss about formalisms? I'm proud to say I've written papers using both situation calculus and event calculus, and my book covers both extensively. It would be so much more valuable if we sought relations between different formalisms and tried to understand the space of possible action formalisms.

The most pertinent comment I've read in this debate so far was Pat Hayes's when he wrote:

> *One of the biggest failures of the KR community generally is that it is virtually impossible to actually publish a knowledge representation itself! One can talk about formalisms and semantics and equivalences etc. etc., . . . but this is all part of the *metatheory* of knowledge representation. But when it comes to actually getting any representing done, we hardly hear about that at all.*

It's as if we were violinists in an orchestra who, instead of making music, spent all their time arguing over who has the nicest violin. Let's make some music. Let's use our formalisms to build theories, and then let's see how those theories fare when used in anger. Then perhaps we'll actually make some progress in common sense reasoning.

Erik Sandewall on 28.10.1997

Murray,

I agree with you that possessiveness about formalisms is a bad thing, but let's not give up this discussion so hastily. After all, it *is* important to understand what is the range of expressiveness of our current major formalisms. What we need, I think, is

- Concrete, well founded arguments (not just "X has recently showed that formalism F can do phenomenon A")
- A structure to the topic, maybe along the lines of the table in Ray Reiter's position statement for the ontologies debate
- A broader scope, considering other properties of a formalism than just its expressiveness. Are there entailment methods which work correctly for the whole range of expressiveness that is being claimed?

Wrt the first item, a concrete and well founded argument may need a little more space than just a few lines in a discussion, while on the other hand it does not require a full paper. The **notes** structure of the present Newsletter and News Journal may come in nicely here. In the Newsletter web pages where the present two panels started (21.10 and 27.10), clicking the title of a position statement leads one to a postscript file for that statement; that presentation of the statement will also go into the monthly News Journal edition. These notes have a journal-like "look and feel" and will be citable; they are one step more formal than what you find in a newsgroup. All newsletter participants are invited to submit their comments in that form (latex preferred).

Wrt structure of the topic, why don't we build on Ray's table - contributions addressing specific combinations of "representation aspect" and "ontology" (possibly correlated with a formalism) are invited. I'll try to set up a web-page structure where every such combination obtains its own thread of messages.

One reason why this discussion will be useful is to clear up some misunderstandings. For example, Michail, when you write

> *From the other hand, the event calculus and other "narrative time-line languages" do not have any term that would keep record*

of what part of the narrative had been done before the moment when a failure happened...

you express a misunderstanding bordering on an mistake. Since each interpretation in a narrative time-line approach contains one history of the world along the time-line, it can also contain the actions that are (were) performed in that history, or up to a point in time in that history. Then the history of past events is not expressed as a *term*, of course, but why would that matter?

In the work on range of applicability for entailment methods, as reported in the "Features and Fluents" book, I started out with a narrative timeline approach simply because it seemed more *natural* for dealing with events with extended duration and overlapping intervals, and with continuous change. However, it became clear during the work that a simple generalization of the time-domain definition made it possible to include situation calculus as a special case, and that virtually all the formal results about the properties of various nonmonotonic methods carried over without any difficulty. In that sense there is no contradiction between sitcalc and narrative timeline approaches, although I still like to think of the former as a special case of the latter.

On the other hand, I have also noticed that it is apparently much easier to *get articles published* if they use situation calculus. This may possibly be due to notational chauvinism (a natural consequence of possessiveness) on the side of some reviewers: If one really believes that (e.g.) the situation calculus is the best answer to all problems, then why accept a paper from someone that hasn't seen the truth?

If our research area is going to conserve an older approach to such an extent that essential new results can't make it through the publication channels, then the whole area will suffer. There, in fact, is an additional reason why we may have to sweat out this discussion about the capabilities of different ontologies and formalisms: not in order to bicker, but to *increase* the acceptance of each other's approaches.

Pat Hayes on 30.10.1997

Before responding to the responses to my comments about the situation calculus, a note on terminology.

The 'situation calculus', 'event calculus', etc., are all just styles of writing axioms in a first-order logic (with suitable modifications to allow circumscription, etc..) The word 'calculus' doesnt point to anything more substantial than a choice of vocabulary and an axiomatic style. (Contrast the useage in 'lambda calculus', for example.) This isn't anything to regret in itself, but it does mean that to talk about something being an 'extension' to a calculus becomes rather fuzzy. There is no end to the relations and axioms one might add to a first-order theory; and if we also allow the axioms of the theory to be

altered and the intuitions which motivated them to be replaced by different intuitions, then we can make any set of axioms into any other set of axioms, so all talk of this or that 'calculus' becomes meaningless. Ray Reiter seems to have done this for the situation calculus. Whatever it is, this 'extended ontology' that Ray describes [see this issue of ENRAC, page 93 f.] bears almost no similarity to the ontology and axiomatic style expounded by McCarthy about 30 years ago (and still used by Ray, along with everyone else, as late as 1991 in his paper in the McCarthy festschrift). It has a different vocabulary, different axioms and is based on different intuitions (which are directly opposed to those motivating the original situation calculus) and has different formal properties. Contrast, for example, Reiter and McCarthy on what a 'situation' is meant to be:

McCarthy (1969): "A situation is the complete state of the universe at an instant of time."

Reiter (1997): "Even at this late stage in AI, many people still don't understand what a situation is, so here's the secret: A situation is a finite sequence of actions. Period. It's not a state, it's not a snapshot, it's a *history*."

Evidently Ray is talking about something different from McCarthy. Nothing wrong with this, of course: I've done it myself from time to time. (Consider my old naive physics 'histories' ontology. World-states are a special case of histories, and there's a systematic translation of situation-vocabulary into history-vocabulary; does that mean that the 'liquids' axiomatisation is written in an "extended" situation calculus?)

Now, it may be said that the field has advanced, and its up to old fogies like me to adapt ourselves to the new terminological consensus. Just as 'frame problem' now means almost everything from Hume's problem of induction to the cost of memory, the meaning of 'situation calculus' has moved with the times. (As Mikhail Soutchanski says, "the SC of 1997" is different from the SC of, say, 1991.) I've made a similar point to Erik, who carelessly used 'ontology' to mean what it meant for about a thousand years, thus risking confusion with the new West-coast sense of 'ontology' (ie. a sorted first-order theory presented in an object-oriented notation, with comments in Managerese.) But, as Erik said, we still need a name for the old sense; and we still need a name for the situation calculus as it was everywhere from 1965 until around 1994 and still is in most of the world outside Toronto. How about 'gofsitcalc'? Whatever we call it, in any case, that's what I was talking about.

More substantive comments to follow.

Pat Hayes on 31.10.1997

As I expected, criticising the sitcalc on this newsletter is rather like farting in church. Much of the spluttering seems to be motivated more by the event than by the content of what I said, however. Murray wrote:

> *I'm sympathetic with most of Pat Hayes's criticisms of the situation calculus, but not when he writes ...*
>
> > Why is it that the only people who feel at all bothered by the frame/ramification/qualification problems are philosophers (who mostly dont even understand what they are) and people working in this rather isolated part of KR?
>
> *The frame problem seems to arise in any logic-based formalism in which the effects of actions are described. It certainly arises in the event calculus, which has a very different ontology to the situation calculus. It also arises in the ontologies of Yoav Shoham's and Erik Sandewall's books, which is why those book took the shape they have. The YSP, in some guise, arises in all these formalisms too. And (at the risk of reviving an old debate Pat had with Drew McDermott), the frame problem seems to arise in Pat Hayes's histories formalism too.*

Well, I'm afraid I disagree. Id be interested to have this explained to me in a little more detail, if anyone can. The histories ontology has its problems, some of them very serious; but the FP isnt among them. Its hard to see how it could be, since the ontology doesnt even refer to states, situations or actions.

One criticism of histories was that some kind of frame-problem-tackling 'continuation principle' was needed to make sure, for example, that the history of a ball's trajectory didnt just stop in the middle of the table somewhere. This was just a misunderstanding. The description of a history includes its boundary conditions, and in the case of a trajectory history the temporal boundaries (assuming no friction) must involve an impact, which in turn requires an impacting surface. So one can infer directly that a trajectory will continue until the ball hits something; no nonmonotonic principles are involved. Ray wrote:

> *When Pat Hayes speaks, one is well advised to listen, because he usually gets it right. But when the godfather of the sitcalc, and a parent of the frame problem says such surprising things about his own creations, I can't restrain myself.*
>
> > First, its based on an overly simplistic view of the way things happen in the everyday world, one obviously inspired by reasoning about what happens inside computers. The everyday

> world just doesnt consist of static states and functions between them: its not organised like a series of snapshots. Sitcalc belongs with SHAKEY, in a world where only the robot can move and nothing else is happening.

False. Situations are simply finite sequences of actions. These need not be just the actions under the robot's control; they can, and in interesting cases do, involve exogenous actions (Fido ate the sandwich that the robot is asked to fetch.) Writing controllers to deal correctly with such exogenous event occurrences has long been the meat-and-potatoes of control theory, and this is certainly possible also in the sitcalc. Indeed, the sitcalc can easily be seen to be a generalization of discrete event control theory.

Part of why we arent communicating here may be that the term 'sitcalc' has become ambiguous, and Ray helped it get that way. The Reiter-sitcalc is a very different beast than the original sitcalc, both in the way it is supposed to describe the world and in how it sets about doing it: so different, in fact, that it is misleading to call it by the same name. (Alright, I concede that maybe the world has come to use 'sitcalc' to refer to Reiter's calculus, and so I ought to change my own vocabulary to avoid misunderstandings. But now what do we call the old situation calculus? Let me distinguish gof-sitcalc, for the situation calculus before Reiter, from R-sitcalc, just to avoid confusion.)

> Second, sitcalc only works properly if we are careful only to mention processes which can be acted upon; that is, it confuses change with action.

I can't figure out what Pat means by this, even with the help of his grow(s) example. I suspect that he wants to distinguish between processes, that evolve in time, and actions, but I'm not sure. So I'll simply say here that there is a sitcalc story for continuous processes, and leave it at that.

No, that wasnt my point. It was more to do with what Ray calls 'exogenous' actions. Part of what made the gof-sitcalc so attractive was the way it could be used to plan: a proof that a goal situation exists automatically gives you a plan of how to achieve it, encoded in the very *name* of the situation. You can read it off from the actions named in the term whcih refers to the goal state. This basic idea was used, with variations and elaborations, for many years and felt to be a wonderful positive feature, as Mikhail Soutchanski kindly reminds us. It is the basis of the analogy with program synthesis (the situation becomes the state of the computer) and with finite-state control theory

(the state of the controlled system.) Fine. But this works *provided* that the only state-to-state functions used in the formalism are those which correspond to state-transitions. But other interpretations of the sitcalc - notably that which claims that it can do the work of temporal modalities - lead naturally to the use of functions to simply assert a temporal relation between situations, with no corresponding implication of there being any kind of action or state-transition which can be used to achieve that new state. Suppose, to use a different example, we want to assert that whenever the stock market is high, it will be lower at some time in the future (the Greenspan axiom?). In tense logic this wouild be something like

```
highdow -> F(lowdow)
```

demodalised into the sitcalc it becomes (simplifying)

```
highdow(s) -> (exists t) later(s, t) & lowdow(t) )
```

Now however if we skolemise this we get

```
highdow(s) -> later(s, greenspan(s)) & lowdow(greenspan(s))
```

where *greenspan* is the skolem function. The ontology of the situation calculus provides no principled way to distinguish functions like this, which simply express an existential temporal relationship, from functions which are supposed to describe achievable state-transitions corresponding to actions.

Now of course we can always make this distinction by adding something. For example, we can add "do-able" as a relation between situations, or, better, as a predicate on functions whcih have been reified into objects by using a 'do' function. Yes, and we can also add Gödel-Bernays set theory, Peano arithmetic and Tarski's axiomatisation of spatial orderings. So what? We can ADD any other axioms you like. But my point was only that there was (and I think still is) a clash of intuitions which the sitcalc didnt resolve; and the ontology of the sitcalc not only doesnt resolve it, but in fact encourages a confusion between these very different ideas, by assimilating them to a single formal structure. And to repeat, I think that much of the work directed towards solutions to the frame problem - notably, almost everything connected with chronological minimisation - is entirely wrongheaded, and seemed plausible only because of this confusion between action and temporal ordering.

> *So what's the point here? With a suitable solution to the frame problem, one can, in the sitcalc, reason in all [temporal] directions.*

Well, maybe I'm not fully following this, but I dont see how its done. Heres an example, from my old liquids axioms. Theres a table

top which is dry in state s but on which theres a pool of liquid in state t, where t is later than s. Now, how did the liquid get there? In fact there are about five ways it can have done so (depending on what one counts as a 'different' way), and some of them involve intermediate stuff which has to have been removed (such as a leaky container). How does one describe this in the situation calculus? Or another example: Drew McDermott is healthy in state s but dead from a gunshot in state t. How do we infer, in the sitcalc, that somebody shot him?

> ..... Why hasnt the FP become a central difficulty in, say, natural language work, or qualitative physics, or planning (as used in industrial applications)? Because those fields typically dont use this clumsy ontology, that's why. These problems are all artifacts of the sitcalc ...

Absolutely false! I can't speak to how the natural language community treats actions, but qualitative physics and planning have no difficulty with the FP because, without exception, they adopt the STRIPS sleeping dogs strategy. Which is to say, they assume they have complete information about world states.

Sorry, just not true. Most qualitatative physics reasoning doesn't assume complete information, and certainly doesnt use a STRIPS strategy. For example, Feng Zhao's work on qualitative phase-space reasoning works with incomplete data and approximate descriptions. In contrast, I might point out that such assumptions as the unique-names axiom that many circumscriptive reasoners rely on so centrally amount to a completeness assumption in disguise, since they essentially insist that all models are isomorphic to Herbrand models, which reduces reasoning to a kind of symbolic state-simulation.
Mikhail wrote:

> *There are reasons why the situation calculus (SC) has been successful for so long time. Here are some of them.*
>
> *1. The SC is simple and easy to understand. ...*
>
> True.
>
> *2. The situation calculus is a foundation for general purpose high-level programming languages. Reminder: This idea is proposed in the 1969 paper "Some philosophical problems from the standpoint of artificial intelligence" (J.McCarthy & P.Hayes). Note that it is an easy exercise to formalize the Turing machine in the SC.*

Thanks for the reminder. It is pretty easy to formalize a Turing machine in almost anything, so this doesnt mean much. But in any case, what has this got to do with the topic we are all concerned with? How did Turing machines come to be relevant here?

> *Moreover, thanks to the explicit situational argument, as long as the SC-based program proceeds, the information about the sequence of actions performed so far, can be used to direct the further execution of a program. For example, if (in the real world) during the execution of a primitive action robot 'fails', analyzing the list of primitive actions performed so far, the robot can (sometimes) infer conclusions regarding what caused the failure. As we know from the control theory and the game theory, the history of the interaction of an agent with an environment (that may include other agents with possibly contradictory goals) may provide useful guidelines when the agent decides how to recover from a 'failure'. From the other hand, the event calculus and other "narrative time-line languages" do not have any term that would keep record of what part of the narrative had been done before the moment when a failure happened.*

I think theres a good point lurking here, but its not quite right as stated. Its not the situational argument as such that makes this possible, but the fact that the terms that get bound to it contain explicit sytactic information about the actions performed. But that's also a problem, as Ive already explained. I think the real advantage of the situational argument is that by having explicit terms which denote the outcomes of actions (not the actions themselves), it allows alternatives to be represented as conjunctions rather than as disjunctions. To say that one might do P (with result A) or one might do Q (with result B) in sitcalc style, one asserts a conjunction: A(do P) & B(do Q) . If we have to talk about one single temporal world (be it 'linear' or not), this has to be a disjunction: either (P, then A) happens, or (Q, then B) does. These alternatives get out of hand very fast.
I wrote and Mikhail answered as follows:

> Most intuitive reasoning done by humans lies entirely outside the purview of the situation calculus.
>
> *Note that your objection can be easily rephrased as: "Most intuitive reasoning done by humans lies entirely outside the purview of the formal logic".*

I have no idea what this response is supposed to mean. Do you identify formal logic with the situation calculus? Or do you mean only that much of intuitive reasoning is outside the scope of our subject? Or what??
Mikhail continued:

> *I'm not sure whether we must have the same concerns that the cognitive science has. Most of the people do not think in terms of C, LISP, PROLOG, but all these languages are still useful for writing programs that will exhibit an intended behavior. Similarly, the SC is useful as the basis for the high-level programming language.*

And again, I'm at a loss to understand your point. What is 'the high-level programming language' you refer to here? For the record, I think that unless our subject is part of cognitive science, then it's just empty formula-hacking. See earlier comments in reply to Erik.

> Yet so firm has been the grip of the sitcalc ontology on people's thinking that examples which do not immediately fit into it are routinely ignored,

Please, formulate those examples in technical terms.

I can't formulate them in technical terms. If I could, they would already be formalised. But for a good start, try looking at

`http://www.dcs.qmw.ac.uk/~rsm/CS98/CS98Problems.html`

Pat Hayes