RJ 10272 (A0211-091) November 19, 2002
Computer Science

IBM Research Report

Decidability of the Basic Situation Calculus, Blocksworld, and Some Solutions to the Frame Problem

Tom Costello
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

Decidability of the basic Situation Calculus, Blocksworld, and some solutions to the Frame Problem

Tom Costello
IBM Almaden

Abstract

The knowledge representation community has often stressed the primacy of first order logic, as a maximal sound and complete system. The results of this paper show that another *strictly incomparable* system, monadic second order logic (which is sound, complete, and *decidable*), is more natural in A.I.'s most classic domains, reasoning about action, non-monotonicity, and blocksworld.

The situation calculus is a classic logical A.I. domain. However, it contains binary predicates, and thus determining theoremhood is *possibly* undecidable. Its now standard formalization contains second order logic, to state induction over situations, and thus might be highly undecidable.

The classic domain of logical A.I., blocksworld, is not axiomatizable in first order logic. We present a sound and complete axiomatization of all its domain constraints—we are unaware of a previous completeness proof.

The projection problem has also been attacked with second order logic, via circumscription. Though the basic situation calculus might be axiomatizable, these solutions could make theoremhood highly undecidable.

In all these cases we show that the problem of theoremhood remains decidable by embedding in monadic second order logic.

1 Introduction

This paper considers some questions concerning the decidability of axiomatizations in the situation calculus.

Logical A.I. is founded on the idea that facts about the world, and an agent's beliefs, intentions, and goals can be formalized in logic. The agent then infers what to do in this logical language. A major problem of logical A.I. is how to represent various kinds of information. There are many possible ways to axiomatize any given domain. Therefore, some method of evaluating when one axiomatization is better than another is essential.

One of the classical ways of better understanding axiomatization in mathematics is to consider certain properties of the axiomatization. Mathematical logic has defined many properties that an axiomatization can have, including completeness, decidability, finite axiomatizability, being equational, being essentially undecidable, etc. These properties allow mathematicians to judge the various claims of different proposals. The basic properties that any axiomatization must have to be worthy of the name are soundness and completeness. That is the axiomatization should imply only facts true of the domains, and it should imply all such facts.

First order logic has a special place in logical A.I. as a sound and complete logic. It is the most expressive language that contains the usual connectives ($\wedge, \neg, \exists$) and is compact (and obeys Löwenheim-Skolem). Compactness is (almost) a necessary condition for a proof theory, as (normal) proofs can only contain finitely many sentences. Thus if a domain requires more than first order logic, it might seem to be inexpressible.

With this in mind, we consider some classical logical A.I. domains, the situation calculus, blocksworld and the projection problem. Theses are know to require extensions to first order logic. We show that despite the need for second order features these domains are expressible in a decidable fragment of logic. Thus we show that an incomparable sound and complete logic is more natural than first order logic for some of logical A.I.'s most hallowed domains.

The paper that introduced situations [McCarthy and Hayes 1969] defined them as follows, "A situation s is the complete state of the universe at an instant of time". The basic mechanism used in the calculus to define a new situation is the result function, $s' = \text{result}(e, s)$.

In this formula s is a situation, and e is an event, and s' is a new situation that results when e occurs. A function or predicate of a situation is called a *fluent*.

We first consider the basic situation calculus, and some proposed axiomatizations.

We then consider the favorite A.I. domain—blocksworld [Manna and Waldinger 1986; Lifschitz 1987]. We show that it is not first order axiomatizable. We give a second order axiomatization, prove a natural completeness result, and show that the theory is decidable.

We then consider a basic problem, first stated in the situation calculus, the *frame problem*, and briefly sketch one proposed solution. We then show that under certain natural conditions this solution to the frame problem is decidable.

Thus we show that an alternative to first order logic exists which is natural for knowledge representation.

1.1 Axioms for the situation calculus

Reiter has suggested that the situations in the situation calculus be defined axiomatically.

He suggests the following axioms[1],

$$\forall \phi . \phi(S0) \wedge (\forall s, a . \phi(s) \rightarrow \phi(do(a,s))) \rightarrow \forall s . \phi(s) \quad (1)$$

$$\begin{array}{l} \forall s_1, s_2, a . s_1 < do(a, s_2) \equiv s_1 \leq s_2 \\ \forall a_1, a_2, s_1, s_2 . do(a_1, s_1) = do(a_2, s_2) \rightarrow \\ \quad a_1 = a_2 \wedge s_1 = s_2 \\ \forall s_1, s_2 . s_1 < s_2 \rightarrow \neg(s < s_1). \end{array} \quad (2)$$

Rather than use $do(a, s)$ we use $result(a, s)$. These axioms are categorical, that is relative to an interpretation of equality of actions, there is a unique model of situations.

This does not mention how actions are described. One way to axiomatize a set of actions is to introduce a finite set of action constants, to state that they are all distinct, and to write that every action is equal to one of these constants. This can axiomatize any finite set of actions.

We can introduce infinite sets of actions by allowing functions that take actions to actions[2].

Proposition 1 *Given an interpretation of equality on actions, there is a unique interpretation of situations that satisfies the above axioms 1 and 2.*

Proof: The unique model is a tree of situations, whose branching factor is the set of actions, and all of whose paths are finite. This tree is unique. ∎

If a theory is axiomatizable and complete, then it is decidable. However, it is not clear that this theory is *axiomatizable* (i.e. has a recursively enumerable set of theorems), as second order logic is not *axiomatizable.*

The axiomatization of the situation calculus above is decidable if the set of actions is finite or a Herbrand universe. We prove this by reducing the problem to the decidability of monadic second order logic with two successors, which is known to be decidable, by Rabin's Tree Theorem [Rabin 1969; Thomas 1994].

[1] He takes $s_1 \leq s_2$ to be a shorthand to mean $s_1 < s_2 \vee s_1 = s_2$

[2] We then introduce a finite set of actions constants, that are distinct. We then state that the functions are surjective, that all action are made from constants by application of functions, and that the action constants are not equal to any function applied to actions. This makes a countable set of actions. . This corresponds to taking the set of actions to be the Herbrand universe.

Definition 2 *Monadic second order logic over two successors has as its language, terms made from a countable set of individual variables $\{x_n | n \in \omega\}$, and the two unary successor functions r_0 and r_1, atomic formulas made from monadic second order variables $\{X_n | n \in \omega\}$ applied to terms, and equality applied to terms, and formulas made from atomic formulas using negation $\neg$ universal quantification over individuals $\forall x$, universal quantification over second order monadic variables $\forall X$, and conjunction $\wedge$.*

Theorem 3 (Rabin) *The monadic second order theory of two successor functions is decidable.*

This theorem can be used to show that the theory of totally ordered countable sets is decidable [Rabin 1969], and thus the theory of well-ordered countable sets if decidable.

We can extend the situation calculus to allow defined relations on actions. In particular, we often well-order our actions with a relation. We shall see examples of this in later sections.

A set A is well-ordered with order type α by a relation R, if in all models A can be put into one to one correspondence with the ordinals below α by a function f, such that for all $a \in A$, $R(a, b)$ if and only if $f(a) < f(b)$.

We introduce the notion of well-ordering, as we will need to state that the set of actions can be well-ordered by a relation definable in the language. This is not the same as the set of actions being countable which is a strictly stronger condition, as we can only describe countable ordinals.

Theorem 4 *The axiomatization of the situation calculus given by Reiter, with an axiomatization of the equality of actions that is complete, is complete and decidable, if the set of actions is finite, or is countable and well-ordered with order type α by a relation definable in the language of the situation calculus.*

The proof encodes the set of actions into combinations of the two successors r_0 and r_1. Each action is associated with a unique finite sequence of applications of r_0 and r_1. As the set of actions can be well-ordered, this can be done in the obvious way. We interpret situations as a subset of the domain, that subset that contains a particular element we interpret at $S0$, and which is closed under application of the functions associated with actions. The set of situations is thus definable. We interpret statements in the situation calculus as statements relativized to this set. Thus the situation calculus is definable in this case. ∎

Thus the basic situation calculus is decidable. Next we consider what happens when we add some unary fluents.

Reiter has suggested representing the effects of actions on fluents by writing explanation closure axioms. Similar ideas were earlier suggested by [Pednault 1989] and [Haugh 1987]. All three proposed solving the frame problem by succinctly writing down the frame axioms.

Normally effect axioms are written in the form:

$$\forall s.\phi^+_A(s) \rightarrow \text{holds}(F, \text{result}(A,s))$$
$$\forall s.\phi^-_A(s) \rightarrow \neg\text{holds}(F, \text{result}(A,s))$$

They noted that both effect axioms and the frame axioms can be written down in the form of *Explanation Closure Axioms*:

$$\begin{array}{l} \forall as.\text{holds}(F, \text{result}(a,s)) \equiv \\ (a = A_1 \wedge \phi^+_{A_1}(s)) \vee \cdots \\ \vee(a = A_k \wedge \phi^+_{A_k}(s)) \vee \\ \text{holds}(F,s) \wedge \quad \neg(a = A_1 \wedge \neg\phi^-_{A_1}(s)) \wedge \cdots \\ \qquad\qquad \wedge\neg(a = A_n \wedge \neg\phi^-_{A_n}(s)) \end{array}$$

Theorem 5 *Given a set of actions $A = \{a_n | n \in \omega\}$ and a relation $R_a(a_n, a_m)$ which is true only if $n < m$, and a set of fluents $F = \{f_n | n \in \omega\}$ and a relation $R_f(f_n, f_m)$ which is true only if $n < m$, then the theory of the situation calculus, axiomatized by Reiter's axioms and explanation closure axioms for every fluent f_i, and axioms giving complete information about the initial situation is complete and decidable.*

The key part of the proof is to define the predicate of situations that corresponds to each fluent. We can define the fluents to be the smallest set of situations that satisfies the explanation closure axioms. We then prove that this defined set of situations is true exactly when the fluent is true of the situation. We can achieve this because the fluent is fully defined, that is the fluent is decided at each situation.

Thus, as long as our fluents are unary, and we have complete information about the initial situation, we still have decidability.

We can improve the above result, so that we no longer need full information about the initial situation, if we have only a finite number of fluents.

Theorem 6 *Given a set of actions $A = \{a_n | n \in \omega\}$ and a relation $R_a(a_n, a_m)$ which is true only if $n < m$, and a set of fluents $F = \{f_1 \dots, f_m\}$ then the theory of the situation calculus, axiomatized by Reiter's axioms and explanation closure axioms for every fluent $f_i | 0 \leq i \leq n$, is decidable (but not complete).*

We complete the initial situation, giving 2^m different possibilities. We use the previous theorem to determine the truth of a statement for each situation. A statement is true in the theory if it is true for every possible initial situation.

2 Blocksworld

The blocks-world is a classic AI domain. It has been widely used as a test-bed for planning systems, and is a favorite of logical AI.

When logicians formalize a mathematical domain, it is usual to provide some kind of completeness proof, as a check that the domain is correctly characterized. A good example of this is Euclidean geometry. This can be seen to be complete, by reducing it to the theory of addition and multiplication of real numbers. The theory of $+$ and $*$ over reals can be shown to be complete by quantifier elimination.

Another way in which we can gain confidence that a theory is appropriate is when various different axiomatizations are shown to be equivalent. The classic example of this is set theory, where many different formalizations are known to be equivalent to ZFC.

When we move to more common-sense domains it becomes harder to know when our axiomatizations are correct. However, the same tools that are used in more mathematical domains, completeness results and the equivalence of multiple axiomatizations can still be used.

While many mathematical theories are characterized by first order sentences, even simple domains like blocks world need second order logic. We prove this fact. However, this is not a cause for despair. Although second order logic is highly undecidable in general, the parts of it that we need to use to characterize blocks-world are decidable.

Even when we consider non-monotonic reasoning (circumscription) that is used to solve the frame problem, means that are expressed in second order logic, we shall see that they are sometimes still decidable.

3 Basic Axioms

Our blocks world has 4 sorts, situations s, blocks b, locations l and actions a. These are all disjoint.

We have the usual situation constant $S0$, and we have a location constant $Table$. We have a countable number of blocks. That is we have a function S on blocks, and a block constant A_0. We write A_{i+1} for $S(A_i)$, and we have,

$$\begin{array}{l} \forall a, a_1.S(a) = S(a_1) \rightarrow a = a_1 \\ \forall a.\neg(S(a) = A_0) \\ \forall \phi.\phi(A_0) \wedge \forall a.\phi(a) \rightarrow \phi(S(a))) \rightarrow \forall a.\phi(a) \end{array} \quad (3)$$

Thus in particular we have, $A_i \neq A_j | i \neq j$. We have locations, which are the *top* of a block, or are the $Table$.

$$\forall l.\exists b.top(b) = l \vee l = Table \quad (4)$$

All distinct location terms denote distinct locations.

$$\begin{array}{l} \forall b, b'.top(b) = top(b') \rightarrow b = b', \\ \forall b.top(b) \neq Table \end{array} \quad (5)$$

We have a function from actions and situations to situations, $r(a,s)$, and a function from blocks and locations to actions, $move(b,l)$, which gives the action where block b is moved to location l.

All distinct action terms are distinct, and all actions are moving blocks.

$$\begin{array}{l} \forall b, b', l, l'.move(b,l) = move(b',l') \rightarrow b = b' \wedge l = l' \\ \forall a.\exists b, l.a = move(b,l) \end{array} \quad (6)$$

We have our foundational axioms for situations. We have fluents, $On(b,l,s)$ which states that b is on location l in situation s, and $Clear(l, s)$. $Clear(l, s)$ is fully defined in terms of On.

$$\begin{array}{l}\forall l, s.Clear(l, s) \equiv \\ (\exists b.\forall b'.l = top(b') \wedge \neg On(b', top(b), s)) \vee l = Table\end{array} \quad (7)$$

We have an explanation closure axiom that determines $On(b', r(a, s))$ in terms of On at the previous situation. If we attempt to move a block which has a block on top of it, or if we attempt to move a block to a location that contains a block, or it we try to move a block to it own top, then there is no change in the locations of all the blocks.

$$\begin{array}{l}\forall b, b', l, l', s.On(b', l', r(move(b, l), s)) \equiv \\ \quad b \neq b' \wedge On(b', l', s) \vee b' = b \wedge l' = l \wedge top(b) \neq l \wedge \\ \quad (\forall b_2.\neg On(b_2, top(b), s)) \wedge ((\forall b_2.\neg On(b_2, l, s)) \\ \quad \vee l = Table) \vee \\ \quad \left(\begin{array}{l}\neg(\forall b_2.\neg On(b_2, top(b), s)) \vee \\ \neg(\forall b_2.\neg On(b_2, l, s) \wedge l \neq Table) \vee \\ (l \neq l' \wedge b \neq b') \vee top(b) = l\end{array}\right) \\ \quad \wedge On(b', l', s)\end{array} \quad (8)$$

From this explanation closure axioms, and the previous axioms, we can derive the following explanation closure axiom for $Clear(l, s)$

A location l is clear after moving b to l', if , l is the table, which is always clear, or $l' \neq l$ and l was clear before, or if b was on l, and $top(b)$ and l' are clear, and $top(b) \neq l'$, or finally, if $l = l'$ and the action fails, that is either $top(b)$ is not clear, or $l' = top(b)$.

That these conditions suffice can be seen by considering the other possibilities, and realizing that in each case, l is not clear.

$$\begin{array}{l}\forall l, b, l', s.Clear(l, r(move(b, l'), s)) \equiv \\ \quad l = Table \vee \\ \quad l \neq l' \wedge Clear(l, s) \vee \\ \quad On(b, l, s) \wedge Clear(top(b), s) \wedge Clear(l', s) \wedge \\ \qquad top(b) \neq l' \vee \\ \quad l = l' \wedge (\neg Clear(top(b), s) \vee l' = top(b))\end{array}$$

To prove this from the previous explanation closure axiom, and the other axioms, we replace $Clear$ by its definition, and then replace each occurrence of $On(b, r(a, l), s)$ by its explanation.

Up until now we have not needed any second order logic. However, our axioms are insufficient to eliminate certain unwanted situations, such as having a block be on its own top. Therefore we consider domain constraints that eliminate these possibilities.

4 Domain Constraints

We will not be able to define our domain constraints in first order logic. We will need the notion of *transitive closure*, to state that our towers cannot be infinitely tall. This cannot be captured in first order logic.

We define an operator $*$ that takes a binary predicate on blocks into its transitive closure. We write $*(P)$ as P^*. We can then define $*$ as follows. This is a definition, as we can prove by induction that $*$ is uniquely defined.

$$\begin{array}{l}(\forall b, b'.P(b, b') \rightarrow P^*(b, b')) \wedge \\ (\forall b, b', b''.P(b, b') \wedge P^*(b', b'') \rightarrow P^*(b, b'')) \wedge \\ \forall \Phi. \left[\begin{array}{l}(\forall b, b'.P(b, b') \rightarrow \Phi(b, b')) \wedge \\ (\forall b, b', b''.P(b, b') \wedge \Phi(b', b'') \rightarrow \Phi(b, b'')) \wedge \\ \forall b, b'.\Phi(b, b') \rightarrow P^*(b, b')\end{array}\right] \rightarrow \\ \qquad \forall b, b'.P^*(b, b')) \rightarrow \Phi(b, b')\end{array} \quad (9)$$

We define the predicate $above(b, b', s)$ as the transitive closure of
$\lambda b, b'.On(b, top(b'), s)$.

$$\forall b, b', s.above(b, b', s) \equiv *(\lambda b, b'.On(b, top(b'), s))(b, b') \quad (10)$$

We can now write some domain constraints about the blocks-world. No block is above itself,

$$\forall b, s.\neg above(b, b, s). \quad (11)$$

Every block is somewhere,

$$\forall b, s.\exists l.On(b, l, s). \quad (12)$$

Every block is above or equal to a block that is on the table.

$$\forall b, s \exists b'.On(b', Table, s) \wedge (above(b, b', s) \vee b = b') \quad (13)$$

Every block is on a single location.

$$\begin{array}{l}\forall b, s \neg On(b, Table, s) \rightarrow \exists b'.On(b, top(b'), s) \wedge \\ \quad \forall b_1.On(b, top(b_1), s) \rightarrow b_1 = b' \\ \forall b, b', s.On(b, top(b'), s) \rightarrow \neg On(b, Table, s)\end{array} \quad (14)$$

Every location has at most one block on it.

$$\forall b, b_1, b_2, s.On(b_1, top(b), s) \wedge On(b_2, top(b), s) \rightarrow b_1 = b_2 \quad (15)$$

There is a clear block above or equal to every block.

$$\forall b, s \exists b'.\forall b'' \neg On(b'', top(b'), s) \wedge (above(b', b, s) \vee b = b') \quad (16)$$

These last four domain constraints are enough to prove that every situation has a set of towers, where a tower is a finite list of blocks $(A_{i_0}, \ldots, A_{i_m})$ where $On(A_{i_{j+2}}, top(A_{i_{j+1}}), s)$ and $On(A_{i_0}, Table, s)$, and no other blocks are on any other locations.

Theorem 7 *Every model $\mathfrak{A}$ of the axioms 13, 14, 15,and 16 at a given situation s can be modeled by a set of totally ordered finite disjoint sets of blocks, such that their union is the entire universe of blocks. Let the set of sets be $D = \{B_i | i \in I\}$ They are disjoint, $\forall i, j.i \neq j \rightarrow B_i \cap B_j = \emptyset$. Each set is totally ordered, let these order relations on each B_i be $<_i$.*

We now interpret every model of $\mathfrak{A}$ as a set D, as follows. We let our index setI be the set of blocks on the table.

$$for\ all\ b.b \in I \stackrel{def}{=} \mathfrak{A} \models On(b, Table, s)$$

We let our sets of blocks B_i be

$$\textit{for all } b.b' \in B_b \stackrel{def}{=} \mathfrak{A} \models above(b', b, s) \vee b = b'$$

We let our order relation $<_b$ be

$$\textit{for all } b.b' <_b b'' \stackrel{def}{=} \mathfrak{A} \models \begin{array}{l}(above(b', b, s) \vee b = b') \\ \wedge above(b'', b', s)\end{array}$$

This shows that our theory of blocksworld agrees with the intuitive notion of a collection of disjoint finite towers. It also suggests the proof of the next theorem, which relies on the representation of blocksworld as a set of total orders.

Theorem 8 *The second order theory of the static blocksworld, 13, 14, 15 and 16 is decidable.*

Proof-Sketch: We reduce this to the decidability of the second order theory of linear order.

The basic idea is to introduce a unary predicate I true just of the blocks on the ground, then all the towers are ordered, and the $\leq$ relation is true of two blocks, if one is in a lower tower than the other, or lower in the same tower. Now above can be recovered from $\leq$ and our new predicate I. The other domain constraints can be stated in this language.

We also have the harder question of the dynamic blocksworld. We can also show that this is decidable. We do this by reducing it to the case we solved earlier. We have a countable set of actions and fluents, both of which are well-ordered.

5 Non-monotonicity

We now consider what happens when we try to solve the frame problem using non-monotonic machinery. To solve the frame problem we need to use a version of circumscription, a non-monotonic mechanism.

Non-monotonic logics are logics where the consequences of a set of sentences are not necessarily consequences of supersets of that set of sentences.

This paper uses circumscription as its non-monotonic machinery. Circumscription is a form of non-monotonic logic introduced by McCarthy [1986]. It expresses the non-monotonic consequences of a finitely axiomatizable theory A in a language L, under a certain circumscriptive policy, as a sentence of second order logic. A circumscriptive policy is a choice of a finite set of formulas of L to minimize[3].

6 The Projection Problem

The Projection Problem is the problem of predicting the future. It was the first problem in reasoning about action that non-monotonic reasoning attempted to solve. We first present the solution, and then we consider how this solution came about.

[3]Implicitly we vary all symbols, any particular symbol can be fixed by a suitable set of minimizations following deKleer and Konolige [1989].

When we consider the set of changes that happen when an action occurs, it would not make sense if the two situations from which we measure have different properties. For this reason, when we compare models, to see if one has fewer changes than another, we only consider those situations where the two models agree.

We call the ordering on models that solves the projection problem the projection assumption. The *projection assumption* is the default that there are as few changes from one situation to the next as possible. The critical change between this assumption, and the naive approaches that fall victim to the Yale Shooting Problem, is that situations that differ on what fluents hold, before the action we consider, are not compared.

Definition 9 *The ordering on models that models the assumption of the projection problem is defined as follows. Let $\mathfrak{A}, \mathfrak{B}$ be two models of $L_{sit}(\overline{a}, \overline{f})$. We say $\mathfrak{A} \leq_{proj} \mathfrak{B}$, if for all situations s and $R(a,s) = \mathfrak{A}[\text{result}(a,s)]$ and $R'(a,s) = \mathfrak{B}[\text{result}(a,s)]$, where $a \in \overline{a}$, then*

$$\begin{array}{l} \textit{If } \forall f \in \overline{f}.\langle f, s\rangle \in \mathfrak{A}[\text{holds}] \textit{ iff } \langle f, s\rangle \in \mathfrak{B}[\text{holds}] \textit{ then} \\ \forall a \in \overline{a}.\{f | \langle f, s\rangle \in \mathfrak{A}[\text{holds}] \not\equiv \langle f, R(a,s)\rangle \in \mathfrak{A}[\text{holds}]\} \subset \\ \quad \{f | \langle f, s\rangle \in \mathfrak{B}[\text{holds}] \not\equiv \langle f, R'(a,s)\rangle \in \mathfrak{B}[\text{holds}]\} \end{array}$$

A possible criticism of this approach is that it does not count the extra changes that might be implied by domain constraints. This criticism is invalid, as for each other change that is implied by a domain constraint, there is another effect axiom that can be derived. Because we have stayed in the language of the situation calculus, we have avoided the problems with domain constraints that plague approaches that introduce new predicates.

Theorem 10 *The assumption of the projection problem is expressible in the Π^1_1 fragment of second order logic.*

Proof: The following sentence is the required Π^1_1 sentence.

$$\begin{array}{l} A[\text{holds}, ab] \wedge \forall \text{holds}' ab'. \quad (A[\text{holds}', ab'] \wedge \\ \quad (\forall a, f, s.(\forall f'.\text{holds}(f', s) \equiv \text{holds}'(f', s))) \rightarrow \\ \quad\quad (ab(a,f,s) \rightarrow ab'(a,f,s)))) \rightarrow \\ \quad \forall a, f, s.(ab'(a,f,s) \equiv ab(a,f,s)) \blacksquare \end{array}$$

There is no regular circumscription policy that captures the projection assumption in this language. This is shown in [Costello 1997], by showing that a basis to the upper sets of the partial order $\leq_{proj}$ are not finitely first order axiomatizable. However, the above sentence is naturally in the form of generalized circumscription.

Theorem 11 *Given a set of actions $A = \{a_n | n \in \omega\}$ and a relation $R_a(a_n, a_m)$ which is true only if $n < m$, and a set of fluents $F = \{f_1 \ldots, f_m\}$ then the theory of consequences of any first order theory in this language under the the projection assumption is decidable (but not complete).*

Obviously the binary predicate $holds(f_i, s)$ can be reduced to a unary predicate on situations F_i. The key step of this proof involves showing that the truth of the ternary predicate *ab* can be reduced to statements about a set of situations by a suitable coding. We recall that we choose the situations to be the closure of a set under the functions we define to be actions. We can introduce a 2^m new functions, not already taken to be any action, and consider the image of situations under these actions to code fluent/situation pairs. We then consider the image of these objects under actions. This codes a fluent/situation/action triple, so long as we can uniquely decompose the object. We can choose our representation of actions and fluents as functions to be such that the action and fluent are retrievable from the composition of an action and a fluent. This allows us to code a triple as a single object, where the set of triples is disjoint from the set of objects that code situations. We can then state that this set if minimal in the obvious way. Then, all second order quantifiers are monadic, from, which the theorem easily follows. ∎

7 Conclusion

This paper considered classic logical A.I. domains, the situation calculus, blocksworld, and the projection problem. Each of these domain is not expressible in first order logic.

We showed that although the situation calculus contains binary predicates, such as $\leq$, and uses second order quantifiers, it is decidable.

We also show completeness and decidability results when we add effect axioms, and explanation closure to the basic situation calculus. This recalls Reiter's work [Reiter 1991] on goal regression.

We show that the blocksworld is not axiomatizable in first order logic, unless we assume a finite number of blocks. We presented an axiomatization of blocksworld which we prove sound and complete—we are unaware of a previous completeness proof. Thus here too, second order logic is needed.

The frame problem, originally posed for the situation calculus has also been attacked with second order logic. We show that one solution to the frame problem, the projection assumption, does not make the basic situation calculus undecidable.

In all these cases we show that the problem of theoremhood remains decidable. This is of importance because a major criticism of non-monotonic methods such as circumscription was that they required undecidable theories, (such as full second order logic), or were intractable. We have shown that in the standard domains the complexity of a circumscriptive solution to the frame problem is simpler than regular first order logic.

References

Baker, A. B. 1991. Nonmonotonic reasoning in the framework of situation calculus. *Artificial Intelligence* 49:5–23.

Costello, T. 1997. *Non-monotonicity and Change.* PhD thesis, Stanford University.

de Kleer, J., and K. Konolige. 1989. Eliminating the Fixed Predicates from a Circumscription. *Artificial Intelligence* 39:391–398.

Grädel, E., P. Kolaitis, and M. Vardi. 1997. On the decision problem for two-variable first-order logic. *Bulletin of Symbolic Logic* 3(1):53–69.

Harel, D. 1979. *First-Order Dynamic Logic.* Berlin/New York: Springer-Verlag.

Harel, D., A. Pnueli, and J. Stavi. 1983. Propositional dynamic logic of nonregular programs. *Journal of Computer and System Sciences* 26(2):222–243.

Haugh, B. A. 1987. Simple causal minimization for temporal persistence and projection. In *Proc. National Conference on Artificial Intelligence (AAAI '87).*

Leeuwen, J. v. (Ed.). 1994. *Handbook of Theoretical Computer Science, Volume B.* Elsevier.

Lifschitz, V. 1987. Formal Theories of Action. In F. Brown (Ed.), *The Frame Problem in Artificial Intelligence*, 121–127. Morgan Kaufmann.

Manna, Z., and R. Waldinger. 1986. Unsolved problems in the blocks world. In *Proceedings Workshop on Planning and Reasoning about Action.* AAAI.

McCarthy, J. 1986. Applications of Circumscription to Formalizing Commonsense Knowledge. *Artificial Intelligence* 28:89–116.

McCarthy, J., and P. Hayes. 1969. Some Philosophical Problems From the Standpoint of Artificial Intelligence. In D. Michie (Ed.), *Machine Intelligence 4*, 463–502. Edinburgh, UK: Edinburgh University Press.

Pednault, E. 1989. ADL: Exploring the middle ground between STRIPS and the situation calculus. In *Proc. First International Conference on Principles of Knowledge Representation and Reasoning (KR '89)*, 324–332.

Rabin, M. 1969. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society* 141:1–35.

Reiter, R. 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In V. Lifschitz (Ed.), *Artificial Intelligence and Mathematical Theory of Computation*, 359–380. Academic Press.

Scott, D. 1962. A decision method for validity of sentences in two variables. *Journal of Symbolic Logic* 27:377.

Thomas, W. 1994. Automata on Infinite Objects. In *Handbook of Theoretical Computer Science, Volume B, [Leeuwen 1994]*, 133–192. Elsevier.