Non-Markovian Control in the Situation Calculus

Alfredo Gabaldon
Department of Computer Science
University of Toronto
Toronto, Canada M5S 3G4
alfredo@cs.toronto.edu

September 28, 2000

Abstract

The property that the executability and the effects of an action are determined entirely by the current state or situation is known as the Markov property and is assumed in most formalizations of action. The fact is, however, that it is not difficult to run into scenarios where the Markov property is not present. We consider removing this assumption from the situation calculus based formalization of actions of Reiter, which forms the basis of the programming language GOLOG, and define an operator for regressing formulas that quantify over past situations with respect to such non-Markovian basic action theories.

1 Introduction

At the core of research in Cognitive Robotics and the logic approach to AI in general is the problem of reasoning about the effects of actions and its accompanying frame, qualification, ramification problems, etc. During the last decade a considerable amount of effort was devoted to this problem and several proposals have been put forward. There are the Situation Calculus [7, 8], the Event Calculus [5], the action language $\mathcal{A}$ [4] and its extensions, and the Features and Fluents approach [12] among others.

All of these proposals assume what is known in systems and control theory as the *Markov property*: whether an action can be executed and what are its effects is determined entirely by the current state or situation. The fact is, however, that it is not difficult to run into scenarios when the Markov property is not present and the executability and effects of an action depend not only on what holds in the current situation, but also on whether some conditions held and some actions occurred at points arbitrarily far into the past. For example, a robot can only attempt a login into a computer if it hasn't performed the same action in each of the previous few situations. Or the effect of entering a pin number at an ATM machine will be that the bank card is confiscated if an incorrect pin number was used three times in a row. Of course, one can represent this with a Markovian theory by introducing

more fluents, but the resulting theory can be considerably larger. In the ATM example, for instance, one would need to introduce fluents $pinAttempt1$ and $pinAttempt2$ and axioms to define when they change truth value.

In this paper, we consider relaxing the requirement of the Markov property from the situation calculus based formalization of actions of Reiter [10] and modifying the regression operator to work with non-Markovian basic action theories and formulas that quantify over past situations. The situation calculus based programming language GOLOG [6] can then be extended to take advantage of the additional flexibility. As an example, consider a scenario where a robot is not supposed to start cleaning an office unless the student that inhabits it has gotten out of the room sometime in the past since the robot arrived. This information can be incorporated into the action precondition axioms of the robot's non-Markovian action theory.

An additional strong motivation behind this work is the need to accommodate qualifications on actions expressed in the form of temporal logic constraints. These constraints are intended to further reduce the number of actions available to the robot at a particular situation based on whether the resulting history satisfies the constraints in some sense. In the case of the cleaning robot mentioned above, for modularity and other reasons, it may be better to incorporate the precondition on the action of start-cleaning in the form of a temporal constraint. In planning, domain dependent knowledge expressed as temporal constraints has been used for search control with very good results [1, 2]. Furthermore, not surprisingly, this problem has a Database analog: dynamic integrity constraint checking (see e.g. [11, 3]). We believe our work can lead to contributions in these areas as well.

2 The language of the Situation Calculus

In this section we briefly review the language of the situation calculus. For a complete description see [9].

The language $\mathcal{L}_{sitcalc}$ is a second order language with equality and with three disjoint sorts: *action*, *situation* and *object*. In addition to $\wedge, \neg, \exists$ and definitions in terms of these for the other standard logical symbols, the alphabet of $\mathcal{L}_{sitcalc}$ includes a countably infinite number of variable symbols of each sort and predicate variables of all arities. A constant symbol S_0 and a function do of sort : $action \times situation \rightarrow situation$, a binary predicate symbol $\sqsubset$ used to define an ordering relation on situations, a binary predicate symbol $Poss$: $action \times situation$, and for each $n \geq 0$ a countably infinite number of function symbols of sort $(action \cup object)^n \rightarrow action$ called *actions*, a countably infinite number of predicate symbols of sort $(action \cup object)^n \times situation$ called *relational fluents*, and a countably infinite number of function symbols of sort $(action \cup object)^n \times situation \rightarrow action \cup object$.

Intuitively, situations are finite sequences of actions (sometimes referred to as *histories*) and this intuition is captured by a set of four *Foundational Axioms* [9][1]:

$$do(a_1, s_1) = do(a_2, s_2) \supset a_1 = a_2 \wedge s_1 = s_2, \tag{1}$$

$$(\forall P).P(S_0) \wedge (\forall a, s)[P(s) \supset P(do(a, s))] \supset (\forall s)P(s), \tag{2}$$

[1] Lower case Roman characters denote variables. Free variables are implicitly universally prenex quantified.

$$\neg s \sqsubset S_0, \tag{3}$$

$$s \sqsubset do(a, s') \equiv s \sqsubset s' \vee s = s'. \tag{4}$$

The initial situation or empty history is denoted by constant S_0. Non-empty histories are built by means of the function *do*.

3 Basic Non-Markovian Theories of Action

In this section we introduce the notion of situation-bounded formulas. Intuitively, an $\mathcal{L}_{sitcalc}$ formula is bounded by situation term σ if all the situation variables it mentions are restricted, through equality or the $\sqsubset$ predicate, to range over subsequences of σ. This notion is useful because in order to apply regression on a formula, one needs to know how many actions there are in each situation, i.e. how many regression steps to apply. A formula that mentions a situation variable can be regressed provided that the variable is restricted to be a subsequence of some situation term with a known number of actions in it.

The following notation is used through out: for $n \geq 0$, we write $do([\alpha_1, \ldots, \alpha_n], \lambda)$ to denote the term of sort *situation* $do(\alpha_n, do(\alpha_{n-1}, \ldots, do(\alpha_1, \lambda)\ldots))$ where $\alpha_1, \ldots, \alpha_n$ are terms of sort *action* and λ stands for a variable s of sort *situation* or the constant S_0.

Definition 1 For $n \geq 0$, define the *length* of the situation term $do([\alpha_1, \ldots, \alpha_n], \lambda)$ to be n.

Definition 2 For $n \geq 0$, let $\alpha_1, \ldots, \alpha_n$ be terms of sort *action*. A term $do([\alpha_1, \ldots, \alpha_n], s)$ is *rooted at* s iff s is the only variable of sort *situation* mentioned by $\alpha_1, \ldots, \alpha_n$ or no variable of that sort is mentioned. A term $do([\alpha_1, \ldots, \alpha_n], S_0)$ is *rooted at* S_0 iff $\alpha_1, \ldots, \alpha_n$ mention no variables of sort *situation*.

Definition 3 For $n \geq 0$, let σ be a term $do([\alpha_1, \ldots, \alpha_n], \lambda)$ rooted at λ. The formulas of $\mathcal{L}_{sitcalc}$ *bounded* by σ are the smallest set of formulas such that:

1. If t_1, t_2 are terms of the same sort whose subterms of sort *situation* (if any) are all rooted at λ, then $t_1 = t_2$ is a formula bounded by σ.
2. If σ' is a term of sort *situation* rooted at some situation variable or constant S_0, then $\sigma' \sqsubset \sigma$ is a formula bounded by σ.
3. For each $n \geq 0$, each n-ary non-fluent predicate P, each n+1-ary fluent F and each n-ary action function symbol A, if $t_1, \ldots, t_n$ are terms of sort *action* or *object* whose subterms of sort *situation* are all rooted at λ, then $P(t_1, \ldots, t_n)$, $F(t_1, \ldots, t_n, \sigma)$ and $Poss(A(t_1, \ldots, t_n), \sigma)$ are formulas bounded by σ.
4. If σ' is a term of sort *situation* rooted at λ' and W is a formula bounded by a possibly different term of sort *situation* also rooted at λ', then $\sigma' \sqsubset \sigma \wedge W$ and $\sigma' = \sigma \wedge W$ are formulas bounded by σ.
5. If W_1, W_2 are formulas bounded by situation terms rooted at λ, then $\neg W_1$, $W_1 \wedge W_2$ and $(\exists v)W_1$, provided σ is not rooted at v, are formulas bounded by σ.

Example 1 The sentence

$$(\exists a, s).do(a, s) \sqsubset do([B_1, B_2, B_3], S_0) \wedge P(do([C_1, C_2], s))$$

is bounded by $do([B_1, B_2, B_3], S_0)$, with subformula $P(do([C_1, C_2], s))$ bounded by $do([C_1, C_2], s)$. Here, variable s is restricted to be equal to one of the situations S_0 or $do(B_1, S_0)$.

Example 2 Let σ, σ' be two different terms of sort *situation* rooted at two different variables, and let $do(\vec{\alpha}, s')$ and $do(\vec{\beta}, s'')$ be terms of the same sort rooted at s' and s'' respectively. The formula

$$(\sigma' \sqsubset do(\vec{\alpha}, s') \vee \sigma' \sqsubset do(\vec{\beta}, s'')) \wedge do(\vec{\alpha}, s') \sqsubset \sigma \wedge do(\vec{\beta}, s'') \sqsubset \sigma$$

is bounded by σ with subformulas

$$\neg(\sigma' \sqsubset do(\vec{\alpha}, s') \wedge do(\vec{\alpha}, s') \sqsubset \sigma \wedge do(\vec{\beta}, s'') \sqsubset \sigma)$$

and

$$\neg(\sigma' \sqsubset do(\vec{\beta}, s'') \wedge do(\vec{\alpha}, s') \sqsubset \sigma \wedge do(\vec{\beta}, s'') \sqsubset \sigma)$$

bounded by σ.

Definition 4 Let W be a formula of $\mathcal{L}_{sitcalc}$ bounded by σ rooted at s. Then W is *strictly bounded* by σ if

- the only term of sort *situation* rooted at s that is mentioned by W and is different to σ is s.
- for every maximal[2]term σ' of sort *situation* mentioned by W that is different to σ, W mentions an atom $\sigma' \sqsubset \sigma''$ or $\sigma' = \sigma''$.
- W does not mention the situation constant S_0.

The intuition behind the above definition is that the situation terms mentioned by a formula strictly bounded by a term σ be restricted to subhistories of σ. The sentence in Example 1 for instance is not strictly bounded by $do([B_1, B_2, B_3], S_0)$ since the term $do([C_1, C_2], s)$ does not satisfy the second condition and in fact it is not a subhistory of $do([B_1, B_2, B_3], S_0)$.

Example 3 The Past Temporal Logic connectives of Chomicki [3] can be expressed in the situation calculus with the following strictly bounded formulas:

1. Previously α: $(\exists a, s').s = do(a, s') \wedge \alpha(s')$.
2. α since β: $(\exists s').s' \sqsubset s \wedge \beta(s') \ \wedge (\forall s'').s' \sqsubset s'' \sqsubseteq s \supset \alpha(s'')$.

[2]A situation term is *maximal* if it is not a proper subterm of another situation term in the formula [9].

3. Sometime in the past α: $(\exists s').s' \sqsubset s \wedge \alpha(s')$.

4. Always in the past α: $(\forall s').s' \sqsubset s \supset \alpha(s')$.

Definition 5 An *action precondition axiom* is a sentence of the form:

$$Poss(A(x_1, \ldots, x_n), s) \equiv \Pi_A(x_1, \ldots, x_n, s),$$

where A is an n-ary function symbol and $\Pi_A(x_1, \ldots, x_n, s)$ is a first order formula with free variables among $x_1, \ldots, x_n, s$ that is bounded by a situation term rooted at s and does not mention the predicate symbol $Poss$.

Example 4 Consider the blocks world where you have a fluent $on(x, y, s)$ saying that block x is on top of block y in situation s, and an action $unstack(x, y)$ which has the effect of making this fluent false when it is executed. Assuming that in this domain the robot's goal is always to build towers, this action is never used to construct part of a goal tower, but it may be used to remove blocks from towers we don't need. Since the robot does not construct unnecessary towers, we can assume that if there is the need to $unstack(x, y)$ then it must be the case that x has always been on y and thus restrict this action to be possible only in this case. The following Action Precondition Axiom captures this:

$$Poss(unstack(x, y), s) \equiv (\forall s').s' \sqsubseteq s \supset on(x, y, s').$$

Definition 6 A *successor state axiom* for an $(n+1)$-ary relational fluent F is a sentence of the form:

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

where $\Phi_F(x_1, \ldots, x_n, a, s)$ is a first order formula with free variables among $x_1, \ldots, x_n, a, s$ that is strictly bounded by s and does not mention the predicate symbol $Poss$.

A *successor state axiom* for an $(n+1)$-ary functional fluent f is a sentence of the form:

$$f(x_1, \ldots, x_n, do(a, s)) = y \equiv \phi_f(x_1, \ldots, x_n, y, a, s),$$

where $\phi_f(x_1, \ldots, x_n, y, a, s)$ is a first order formula with free variables among $x_1, \ldots, x_n, y, a, s$ that is strictly bounded by s and does not mention the predicate symbol $Poss$.

Example 5 Consider the ATM example mentioned in the introduction. A Successor State Axiom for a fluent $card\_confiscated(s)$ could have the form:

$$\begin{aligned} card\_confiscated(do(a, s)) \equiv\ & card\_confiscated(s) \vee \\ & [(\exists s_1, s_2, p).a = enter\_pin(p) \wedge invalid\_pin(p) \wedge \\ & \quad do(a, s_1) \sqsubset do(a, s_2) \sqsubseteq s.] \end{aligned}$$

Relaxing the strictly bounded condition in successor state axioms to be simply bounded, complicates regression. Consider the following successor state axioms:

$$P(do(a,s)) \equiv (\exists s').s' \sqsubset s \wedge Q(do([B_1,B_2,B_3],s'))$$

$$Q(do(a,s)) \equiv (\exists s').s' \sqsubset s \wedge P(do([C_1,C_2,C_3],s'))$$

Intuitively, "regressing" $P(do[A_1,A_2],S_0)$ with respect to the above axioms would result in $Q(do([B_1,B_2,B_3],S_0))$ and this in turn in $P([C_1,C_2,C_3],S_0) \vee P([B_1,C_1,C_2,C_3],S_0)$. Clearly, regression is not working here since the situation terms are growing.

A *Non-Markovian Basic Action Theory* $\mathcal{D}$ is a theory of $\mathcal{L}_{sitcalc}$ consisting of the following set of axioms:

- the foundational axioms Σ.
- a set of successor state axioms $\mathcal{D}_{ss}$.
- a set of action precondition axioms $\mathcal{D}_{ap}$.
- a set of unique name axioms for actions $\mathcal{D}_{una}$.
- a set of first order sentences $\mathcal{D}_{S_0}$ that mention no situation terms other than S_0 and represent the initial theory of the world.

$\mathcal{D}$ must also satisfy some consistency properties on fluents (see [9] for details).

4 Regression

In this section we define, building on the work of Reiter [10], a *regression operator* $\mathcal{R}$ for regressing bounded formulas of $\mathcal{L}_{sitcalc}$ with respect to a non-Markovian basic action theory.

Definition 7 A formula W of $\mathcal{L}_{sitcalc}$ is *regressable* iff

1. W is first order.
2. W is bounded by a term of sort *situation* rooted at S_0.
3. For every atom of the form $Poss(\alpha,\sigma)$ mentioned by W, α has the form $A(t_1,\ldots,t_n)$ for some n-ary action function symbol A of $\mathcal{L}_{sitcalc}$.

Definition 8 *(Regression)* Let W be a regressable formula of $\mathcal{L}_{sitcalc}$.

1. If W is a regressable atom[3] of one of the following forms:

   - an equality atom of the form

   $$do([\alpha'_1,\ldots,\alpha'_m],S_0) = do([\alpha_1,\ldots,\alpha_n],S_0),$$

[3]Note that a regressable atom that is not an equality or a $\sqsubset$-atom mentions only situation terms rooted at S_0.

- a $\sqsubset$-atom of the form

$$do([\alpha'_1, \ldots, \alpha'_m], S_0) \sqsubset do([\alpha_1, \ldots, \alpha_n], S_0),$$

- an atom $Poss(A(\vec{t}), \sigma)$ where $A(\vec{t})$ and σ are terms of sort *action* and *situation* respectively,
- an atom whose only situation term is S_0,
- an atom that mentions a functional fluent term of the form $g(\vec{t}, do(\alpha, \sigma))$ and is not of the form $\sigma' = \sigma''$ or $\sigma' \sqsubset \sigma''$ where σ' is rooted at a variable of sort *situation*,
- a relational fluent atom $F(\vec{t}, do(\alpha, \sigma))$,

then $\mathcal{R}[W]$ is defined exactly as it was in [9] for theories with the Markov property.

2. Suppose W is a regressable formula of the form

$$do([\alpha_1, \ldots, \alpha_m], s) \sqsubset do([\alpha'_1, \ldots, \alpha'_n], S_0) \wedge W'$$

where W' may be empty.

If $m \geq n$, then $\mathcal{R}[W] = false$.

If $m < n$, then

$$\begin{array}{l} \mathcal{R}[W] = \\ \neg\{\ \neg\mathcal{R}[do([\alpha_1, \ldots, \alpha_m], s) = do([\alpha'_1, \ldots, \alpha'_{n-1}], S_0) \wedge W'] \wedge \\ \neg\mathcal{R}[do([\alpha_1, \ldots, \alpha_m], s) \sqsubset do([\alpha'_1, \ldots, \alpha'_{n-1}], S_0) \wedge W']\ \}. \end{array}$$

3. Suppose W is a regressable formula of the form

$$do([\alpha_1, \ldots, \alpha_m], s) = do([\alpha'_1, \ldots, \alpha'_n], S_0) \wedge W'$$

where $m \geq 1$ and W' may be empty.

If $m > n$, then $\mathcal{R}[W] = false$.

If $m \leq n$, then

$$\begin{array}{rl} \mathcal{R}[W] = & \mathcal{R}[\ (\alpha_1 = \alpha'_{n-m+1} \wedge \ldots \wedge \alpha_m = \alpha'_n \wedge \\ & \quad s = do([\alpha'_1, \ldots, \alpha'_{n-m}], S_0)\ \wedge W'\ ]. \end{array}$$

4. Suppose W is a regressable formula of the form

$$s = do([\alpha_1, \ldots, \alpha_n], S_0) \wedge s = do([\alpha'_1, \ldots, \alpha'_m], S_0) \wedge W'$$

where W' may be empty.

If $n \neq m$ then $\mathcal{R}[W] = false$.

If $n = m$ then

$$\mathcal{R}[W] = \mathcal{R}[do([\alpha_1, \ldots, \alpha_n], S_0) = do([\alpha'_1, \ldots, \alpha'_m], S_0) \wedge W'].$$

5. Suppose W is a regressable formula of the form

$$s = do([\alpha_1, \ldots, \alpha_n], S_0) \wedge W'$$

such that W' may be empty and it does not mention any other equality atoms between s and a situation term rooted at S_0. Then

$$\mathcal{R}[W] = \mathcal{R}[W'|^s_{do([\alpha_1,\ldots,\alpha_n],S_0)}].$$

6. For the remaining possibilities, regression is defined as follows:

$$\mathcal{R}[\neg W] = \neg\mathcal{R}[W],$$

$$\mathcal{R}[W_1 \wedge W_2] = \mathcal{R}[W_1] \wedge \mathcal{R}[W_2].$$

For a variable v of any sort other than *situation*:

$$\mathcal{R}[(\exists v)W] = (\exists v)\mathcal{R}[W].$$

For a variable s of sort *situation*:

$$\mathcal{R}[(\exists s)W] = \mathcal{R}[W].$$

Theorem 1 Suppose W is a regressable formula of $\mathcal{L}_{sitcalc}$ and $\mathcal{D}$ is a basic non-Markovian action theory. Then,

1. $\mathcal{R}[W]$ is a formula uniform in S_0.[4]
2. $\mathcal{D} \models (\forall)W \equiv \mathcal{R}[W]$.

Proof 1 We adapt the proof of soundness and completeness of regression for Markovian theories of action from [9]. The proof is by induction based on a binary relation $\prec$ similar to that in [9] whose definition we omit here.

Given a bounded regressable formula W, let $L(W)$ be the sum of the lengths of all situation terms σ rooted at some situation variable such that W does not mention any atom $\sigma = \sigma'$ or $\sigma \sqsubset \sigma'$, and define

$$index(W) = ((C, E, I, \lambda_1, \lambda_2, \ldots), P)$$

where C is the total number of connectives and quantifiers in W, E is the number of equality atoms on situation terms in W, I is the number of $\sqsubset$-atoms on situation terms in W, for

[4] A formula is uniform in σ iff it is first order, does not mention *Poss*, $\sqsubset$, situation variables, equality on situations, and σ is the only situation term mentioned by fluents in their situation argument. For the formal definition see [9].

$m \geq 1$, λ_m is the number of occurrences in W of maximal situation terms of length $m - L(W)$ rooted at S_0, and P the number of atoms of the form $Poss(\alpha, \sigma)$ mentioned by W.

Our definition of $index(W)$ differs from the one used by Reiter and Pirri in two ways. Parameters E and I appear now before the λs because regressing a fluent may introduce new equality and $\sqsubset$-atoms. More noticeably, the λs here are "shifted" right by $L(W)$, e.g. if there is one term of length k then $\lambda_{k+L(W)} = 1$. The reason behind this is that after a regression step on a formula with a situation variable, it is possible that a situation term is replaced by a longer one. For instance, the formula $s = do(A, S_0) \wedge P(do(B, s))$ would be regressed to $P(do([A, B], S_0))$.

If W has index $((0, 0, \ldots), 0)$, then W must be an atom which does not mention $Poss$, $\sqsubset$ or equality between situation terms, and W must be uniform in S_0, which implies by definition of $\mathcal{R}$ that $\mathcal{R}[W] = W$, and hence the theorem holds.

Consider W such that $\nu = index(W) \succ ((0, 0, \ldots), 0)$ and assume the theorem for all regressable formulas with index $\prec \nu$.

1. In the cases when W is an equality atom of the form $do([\alpha_1, \ldots, \alpha_m], S_0) = do([\alpha'_1, \ldots, \alpha'_n], S_0)$, a $\sqsubset$-atom of the form $do([\alpha_1, \ldots, \alpha_m], S_0) \sqsubset do([\alpha'_1, \ldots, \alpha'_n], S_0)$ or an atom whose only situation term is S_0, the argument is identical to that in [9] so we omit it here.

2. Suppose W is a regressable atom of the form $Poss(A(\vec{t}), \sigma)$ for terms $A(\vec{t})$ and σ of sorts *action* and *situation* respectively. Then there must be an action precondition axiom (5) in $\mathcal{D}$ for action A. By definition, $\Pi_A(\vec{x}, s)$ does not mention predicate $Poss$. Therefore, $index(\Pi_A(\vec{t}, \sigma)) \prec index(W)$. Furthermore, since $Poss(A(\vec{t}), \sigma)$ is a regressable atom, σ and all subterms of sort *situation* mentioned by $\vec{t}$ must be rooted at S_0. From this and the definition of action precondition axioms it follows that $\Pi_A(\vec{t}, \sigma)$ is regressable. The argument then follows as in [9].

3. Suppose W mentions a functional fluent term $f(\vec{t}, do(\alpha, \sigma))$, with corresponding successor state axiom

   $$f(\vec{x}, do(a, s)) = y \equiv \phi_f(\vec{x}, y, a, s),$$

   in $\mathcal{D}_{ss}$, that is to be regressed by the primality criterion described in [9]. Let us show that

   $$(\exists y).\phi(\vec{t}, y, \alpha, \sigma) \wedge W|_y^{f(\vec{t}, do(\alpha, \sigma))} \tag{5}$$

   is regressable and has index $\prec index(W)$. First, since W is an atom and it is bounded by a situation term rooted at S_0, then either σ is rooted at S_0 or W is an atom of the form $\sigma_1 \sqsubset \sigma_2$ where σ_1 and σ are rooted at the same variable and σ_2 is rooted at S_0. Since, by definition, $\phi_f(\vec{x}, y, a, s)$ is bounded by s, in either case formula (5) is bounded by a situation term rooted at S_0 and hence is regressable.

   Second, $\phi_f(\vec{x}, y, a, s)$ does not mention the predicate $Poss$ and since $f(\vec{t}, do(\alpha, \sigma))$ is a prime functional fluent term (see [9]) the only term of sort *situation* mentioned by $\vec{t}, \alpha$ is S_0. Moreover, since $\phi_f(\vec{x}, y, a, s)$ is strictly bounded by s, all terms of sort *situation*

mentioned by $\phi(\vec{t}, y, \alpha, \sigma)$ which are different to σ are rooted at a situation variable and appear in an equality or $\sqsubset$-atom. Therefore,

$$index((\exists y).\phi(\vec{t}, y, \alpha, \sigma) \wedge W|_y^{f(\vec{t}, do(\alpha, \sigma))}) \prec index(W).$$

The argument then follows as in [9].

4. Suppose W is a relational fluent atom of the form $F(\vec{t}, do(\alpha, \sigma))$ that does not mention any functional fluent term of the form $g(\vec{t}, do(\alpha, \sigma))$, and with the following successor state axiom:

$$F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a, s).$$

Since $F(\vec{t}, do(\alpha, \sigma))$ is regressable, σ and all subterms of sort *situation* mentioned by $\vec{t}$ must be rooted at S_0. Since, by definition, $\Phi_F(\vec{x}, a, s)$ is bounded by s, $\Phi(\vec{t}, \alpha, \sigma)$ is regressable. Furthermore, $\Phi_F(\vec{x}, a, s)$ does not mention the predicate $Poss$ and, since it is strictly bounded by s, all terms of sort *situation* mentioned by $\Phi(\vec{t}, \alpha, \sigma)$ which are different to σ are rooted at a situation variable and appear in an equality or $\sqsubset$-atom. Therefore,

$index(\Phi(\vec{t}, \alpha, \sigma)) \prec index(W)$. The argument then follows as in [9].

5. The only remaining cases are when W is an atom $\sigma' \sqsubset \sigma$ and $\sigma' = \sigma$ where σ' is rooted at some situation variable and σ is rooted at S_0.[5] These cases are covered below.

6. Suppose W is a regressable formula of the form $do([\alpha_1, \ldots, \alpha_m], s) \sqsubset do([\alpha'_1, \ldots, \alpha'_n], S_0) \wedge W'$

where W' may be empty. If $m \geq n$, $\mathcal{R}[W] = false$ and the theorem is immediate. For $m < n$, it is easy to see that the indices of formulas

$$do([\alpha_1, \ldots, \alpha_m], s) = do([\alpha'_1, \ldots, \alpha'_{n-1}], S_0) \wedge W' \tag{6}$$

$$do([\alpha_1, \ldots, \alpha_m], s) \sqsubset do([\alpha'_1, \ldots, \alpha'_{n-1}], S_0) \wedge W' \tag{7}$$

are $\prec index(W)$. Furthermore, since W is regressable, W' is bounded by a situation term rooted at S_0 or by one rooted at s. Therefore, formulas (6) and (7) are regressable. This concludes part 1 of the theorem. Part 2 follows from the induction hypothesis and Foundational Axiom (4).

7. Suppose W is a regressable formula of the form $do([\alpha_1, \ldots, \alpha_m], s) = do([\alpha'_1, \ldots, \alpha'_n], S_0) \wedge W'$

where W' maybe empty. If $m > n$, $\mathcal{R}[W] = false$ and the theorem is immediate. In the case $m \leq n$, the formula $\alpha_1 = \alpha'_{n-m+1} \wedge \ldots \wedge \alpha_m = \alpha'_n \wedge s = do([\alpha'_1, \ldots, \alpha'_{n-m}], S_0) \wedge W'$ clearly has an index $\prec index(W)$ and is regressable and equivalent to W.

[5] If σ is also rooted at some situation variable, the atom is not regressable.

8. Suppose W is a regressable formula of the form $s = do([\alpha_1, \ldots, \alpha_m], S_0) \wedge s = do([\alpha'_1, \ldots, \alpha'_n], S_0) \wedge W'$

   where W' maybe empty. If $m \neq n$, then $\mathcal{R}[W] = false$ and the theorem is immediate. If $m = n$, it is easy to see that the formula $do([\alpha_1, \ldots, \alpha_m], S_0) = do([\alpha'_1, \ldots, \alpha'_m], S_0) \wedge W'$ has index $\prec index(W)$ and is regressable and equivalent to W.

9. Suppose W is a regressable formula of the form $s = do([\alpha_1, \ldots, \alpha_n], S_0) \wedge W'$ where W' maybe empty.

   If W' does not mention variable s, then the theorem is immediate. If W' mentions variable s then it mentions a term σ of sort *situation* rooted at s which does not appear in an equality atom or an atom $\sigma \sqsubset \sigma'$ of W. This implies that $L(W) \geq L(W'|^s_{do([\alpha_1,\ldots,\alpha_n],S_0)})$. Therefore, $index(W'|^s_{do([\alpha_1,\ldots,\alpha_n],S_0)}) \prec index(W)$. The two formulas are clearly equivalent.

10. The cases when W is a regressable formula of the form $\neg W_1$, $W_1 \wedge W_2$ and $(\exists v)W_1$ are straightforward.

Corollary 1 Suppose W is a regressable formula of $\mathcal{L}_{sitcalc}$ and $\mathcal{D}$ is a basic non-Markovian theory of actions.

$$\mathcal{D} \models W \text{ iff } \mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models \mathcal{R}[W].$$

The proof of this corollary requires a Relative Satisfiability Theorem similar to Theorem 1 in [9], establishing that a non-Markovian basic action theory $\mathcal{D}$ is satisfiable iff $\mathcal{D}_{una} \cup \mathcal{D}_{S_0}$ is satisfiable.

5 Conclusion

As we mentioned in the introduction, most of the proposals that have been introduced for representing dynamical systems assume the Markov Property. Removing this assumption from the version of the situation calculus discussed throughout this paper without major changes to the formalism was possible thanks to the fact that histories are first order objects in these theories. Removing this assumption from other formalizations where this is not the case would require considerable more effort. The action languages based on $\mathcal{A}$ [4] have semantics based on transition systems. This makes it difficult to refer to paths of the transition system when defining the transition system itself, although once it is defined, one can refer to histories in queries. Event calculus formulas can refer to the past by means of time points, but additional machinery is necessary to extract a sequence of actions from the information given in terms of time points.

We have extended Reiter's situation calculus based formalization of actions [10, 9] to allow non-Markovian action theories, i.e. theories where action precondition and successor state axioms may refer to situations that precede the current situation, defined a class of formulas bounded by a finite history and which may quantify over situation variables ranging over the subhistories of the binding situation, and defined a sound and complete regression

operator which can be used to obtain a logically equivalent formula whose only situation term is S_0 from a formula from this class.

In the future, we plan to implement regression of formulas based on the non-Markovian action theories introduced here, analyze the complexity of regression based on such theories, extend GOLOG with the added functionality of non-Markovian control, and consider the use of temporal logic constraints within this framework.

Acknowledgements: Special thanks to Ray Reiter for many suggestions and advice on this work. I am also grateful to Fahiem Bacchus, Gero Iwan and Lucia Moura for useful discussions related to the subject of this paper. Thanks also to the anonymous reviewers for their comments.

References

[1] Fahiem Bacchus and Froduald Kabanza. Using temporal logic to control search in a forward chaining planner. In M. Ghallab and A. Milani, editors, *New Directions in Planning*, pages 141–153. IOS Press, 1996.

[2] Fahiem Bacchus and Froduald Kabanza. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence*, 16:123–191, 2000.

[3] Jan Chomicki. Efficient checking of temporal integrity constraints using bounded history encoding. *ACM Transactions on Database Systems*, 20(2):148–186, 1995.

[4] Michael Gelfond and Vladimir Lifschitz. Representing Actions and Change by Logic Programs. *Journal of Logic Programming*, 17:301–322, 1993.

[5] R.A. Kowalski and M.J. Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95, 1986.

[6] Hector J. Levesque, Raymond Reiter, Ives Lespérance, Fangzhen Lin, and Richard B. Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1–3):59–83, April–June 1997.

[7] J. McCarthy. Situations, actions and causal laws. Technical report, Stanford University, 1963. Reprinted in Semantic Information Processing (M. Minsky ed.), MIT Press, Cambridge, Mass., 1968, pp. 410–417.

[8] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969. Also appears in N. Nilsson and B. Webber (editors), Readings in Artificial Intelligence, Morgan-Kaufmann.

[9] Fiora Pirri and Ray Reiter. Some contributions to the metatheory of the Situation Calculus. *Journal of the ACM*, 46(3):325–364, 1999.

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

[11] G. Saake and U. W. Lipeck. Foundations of Temporal Integrity Monitoring. In C. Rolland, F. Bodart, and M. Leonard, editors, *Proc. of the IFIP Working Conf. on Temporal Aspects in Information Systems*, pages 235–249, Amsterdam, 1988. North-Holland.

[12] E. Sandewall. *Features and Fluents: The Representation of Knowledge about Dynamical Systems.* Oxford University Press, 1994.