Ann Math Artif Intell (2010) 58:3–83
DOI 10.1007/s10472-010-9176-z

# A description logic based situation calculus

**Yilan Gu · Mikhail Soutchanski**

Published online: 18 March 2010


**Abstract** We consider a modified version of the situation calculus built using a two-variable fragment of the first-order logic extended with counting quantifiers. We mention several additional groups of axioms that can be introduced to capture taxonomic reasoning. We show that the regression operator in this framework can be defined similarly to regression in Reiter's version of the situation calculus. Using this new regression operator, we show that the projection and executability problems (the important reasoning tasks in the situation calculus) are decidable in the modified version even if an initial knowledge base is incomplete. We also discuss the complexity of solving the projection problem via regression in this modified language in general. Furthermore, we define description logic based sub-languages of our modified situation calculus. They are based on the description logics $\mathcal{ALCO}(U)$ (or $\mathcal{ALCQO}(U)$, respectively). We show that in these sub-languages solving the projection problem via regression has better computational complexity than in the general modified situation calculus. We mention possible applications to formalization of Semantic Web services and some connections with reasoning about actions based on description logics.

**Keywords** Reasoning about action and change · Semantic web and web services · Situation calculus · Description logics · Two-variable logic with counting quantifiers

**Mathematics Subject Classification (2000)** 68T30

Y. Gu (✉)
Department of Computer Science, University of Toronto,
10 King's College Road, Toronto, ON, M5S 3G4, Canada
e-mail: yilan@cs.toronto.edu

M. Soutchanski
Department of Computer Science, Ryerson University,
245 Church Street, ENG281, Toronto, ON, M5B 2K3, Canada
e-mail: mes@cs.ryerson.ca

"*All is change; all yields its place and goes.*" — Euripides (c. 485–406 BCE)
"*Nothing is permanent but change.*" — Heraclitus (c. 540–c. 480 BCE)

## 1 Introduction

The situation calculus (SC) is a well known and popular logical theory for reasoning about changes caused by events and actions. There are several different formulations of SC. According to John McCarthy the history is the following: "[63] proposed mathematical logic as a tool for representing facts about the consequences of actions and using logical reasoning to plan sequences of actions that would achieve goals. Situation calculus as a formalism was proposed in [64] and elaborated in [68]. The name situation calculus was first used in [68] but wasn't defined there. McCarthy [65] proposed to solve the frame and qualification problems by circumscription, but the proposed solution to the frame problem was incorrect. Shanahan [90] and Reiter [82] describe several situation calculus formalisms and give references" (see the footnote 4 in [67]).

In this paper we would like to consider the SC from [82] that extends the original SC with time, concurrency, stochastic actions, etc. It serves as a foundation for the Process Specification Language (PSL) that axiomatizes a set of primitives adequate for describing the fundamental concepts of manufacturing processes (PSL has been accepted as an international standard) [35, 36]. It is used to provide a well-defined semantics for Web services and a foundation for a high-level programming language Golog [9, 51, 69, 73]. However, because the situation calculus is formulated in a general predicate logic, reasoning about effects of sequences of actions is undecidable (unless some strong restrictions are imposed on the theory that axiomatizes the initial state of the world).

The first motivation for our paper intends to overcome this difficulty. We propose to use a two-variable fragment $FO^2$ [11, 40] of the first-order logic (FOL) as a foundation for an initial theory in a modified situation calculus. Because the satisfiability problem in this fragment is known to be decidable (it is in NExpTime) [33, 76], we demonstrate that by reducing reasoning about effects of actions to reasoning in this fragment, one can always guarantee decidability for a wide useful class of formulas no matter what is the syntactic form of the theory representing the initial state of the world. Note an important caveat. We are not going to design a decidable logic for reasoning about actions (see work in this direction reviewed in Section 7) by imposing strong restrictions on the language such as allowing only action constants and disallowing more complex action terms. Instead, we consider a fragment of the SC where only particular reasoning problems become decidable, but these problems are exactly those that can be important in applications. Consequently, it should not be a surprise for the readers to see that even if an initial theory is an $FO^2$ theory, that is formulated using object variables $x$ and $y$, we include additional variables ($a$, for actions, and $s$, for situations), action terms and situation terms common in the SC. As we show in this paper, the reasoning problems that we care about can still be reduced to the theorem proving task in $FO^2$ (or in fragments of $FO^2$). At the same time, the satisfiability problem for arbitrary formulas in our modified situation calculus remains undecidable [11], unless we would commit to further restrictions on our language.

The second motivation for our paper comes from description logics. Description Logics (DLs) [5] are a well-known family of knowledge representation formalisms, which play an important role in providing the formal foundations of several widely used Web ontology languages including the Web Ontology Language (OWL) [44] in the area of the Semantic Web. Many expressive DLs can be translated to $FO^2$ (or to $C^2$ that is $FO^2$ extended with counting quantifiers [77, 80]) and offer considerable expressive power going far beyond propositional logic, while ensuring that reasoning is decidable [12]. DLs have been mostly used to describe static knowledge bases and not dynamics due to changes. However, several research groups consider formalization of actions using DLs or extensions of DLs. Following the key observation that reasoning about complex actions can be carried out in a fragment of the propositional situation calculus, in [30], an epistemic extension of DLs was given to provide a framework for the representation of dynamical systems. However, the representation and reasoning about actions in this framework are strictly propositional, which reduces the representation power of this framework. In [3], another approach was proposed for integrating description logics and action formalisms. They take the well-known description logic $\mathcal{ALCQIO}$ (and its sub-languages) as foundation and show that the complexity of executability and projection problems (two basic reasoning problems for possibly sequentially composed actions) coincides with the complexity of standard DL reasoning. However, actions (services) are represented in their paper meta-theoretically, not as first-order (FO) terms. This can potentially lead to some complications when specifications of other reasoning tasks are considered because it is not possible to quantify over actions in their framework. Other related work is reviewed in Section 7. In our paper, we take a different approach and represent actions as FO terms, but achieve integration of taxonomic reasoning and reasoning about actions by restricting the syntax of the situation calculus and by introducing additional axioms to represent a taxonomy. We do not consider stochastic actions explicitly, but using reduction of stochastic actions to constituent deterministic actions explored in [6, 13, 82] the techniques proposed in this paper can be extended as well to stochastic actions (with no more than two arguments). In addition to the aforementioned basic reasoning problems about effects of actions (executability and projection), there are several other important and challenging problems such as the ramification problem and the qualification problem. However, we feel that these challenging problems deserve a separate consideration and it remains to be seen how these problems can be addressed or if one of the existing proposals can be adapted to our setting. We are aware that there is work related to solving these problems both in the situation calculus [53] and in other action formalisms [41, 48, 62, 83, 92] (to name a few works), but a thorough discussion will take us too far from our main topic. The main contribution of our paper to the area of service composition and discovery is the following. We show that by using services that are composed from atomic services with no more than two parameters and by using only those properties of the world which have no more than two parameters (to express a goal condition), one can guarantee that the executability and projection problems for these services can always be solved even if information about the current state of the world is incomplete.

Our paper is structured as follows. In Section 2, we briefly review Reiter's situation calculus. In Section 3 we review description logics and the extension of $FO^2$ with counting quantifiers. In Section 4, we discuss details of our proposal:

the language $\mathcal{L}_{sc}^{C^2}$ of our modified SC. In Section 5.1, we consider an extension of regression (the main reasoning mechanism in the situation calculus) and investigate the computational complexity in Section 5.3. In Section 5.2, we consider an example of the modified SC that illustrates potential applications to Semantic Web Services. In Section 5.4, we consider a fragment of $FO^2$ that corresponds to a DL with better complexity properties than $FO^2$. Then we define a new SC based on this fragment, which can be considered as a sub-language of $\mathcal{L}_{sc}^{C^2}$. Moreover, in Section 6, we show a strict complexity result for the executability and projection problems of regressable formulas with ground situation terms in the modified SC without using the regression reasoning mechanism. Finally, in Section 7, we discuss briefly other related approaches to reasoning about actions.

## 2 The situation calculus

The situation calculus (SC) $\mathcal{L}_{sc}$ is a predicate language for axiomatizing dynamical systems. All dialects of the SC $\mathcal{L}_{sc}$ include three disjoint sorts: *action*, *situation* and *object*. **Actions** are FO terms consisting of an action function symbol and its arguments. Actions change the world. **Situations** are FO terms which denote world histories. A distinguished constant $S_0$ is used to denote the *initial situation*, and function $do(a, s)$ denotes the situation that results from performing action $a$ in situation $s$. Every situation corresponds uniquely to a sequence of actions. Moreover, notation $s' \preceq s$ means that either situation $s'$ is a subsequence of situation $s$ or $s = s'$.[1] **Objects** are FO terms other than actions and situations that depend on the domain of application. We assume that distinct individual names denote distinct objects, i.e., we have the unique name axioms for object constants. **Fluents** are relations or functions whose values may vary from one situation to the next. Normally, a fluent is denoted by a predicate or function symbol whose last argument has the sort *situation*. For example, $F(\vec{x}, do([\alpha_1, \cdots, \alpha_n], S_0))$ represents a relational fluent in the situation $do(\alpha_n, do(\cdots, do(\alpha_1, S_0) \cdots))$ resulting from execution of ground action terms $\alpha_1, \cdots, \alpha_n$ in $S_0$. We do not consider functional fluents in this paper.

The SC includes the distinguished predicate $Poss(a, s)$ to characterize actions $a$ that are possible to execute in $s$. For any SC formula $W$ that is a FO formula constructed using the usual FO logical operators and a term $s$ of sort *situation*, we say $W$ is a formula *uniform* in $s$ iff it does not mention the predicate $Poss$, it does not quantify over variables of sort *situation*, it does not mention equality on situations, and whenever it mentions a term of sort *situation* in the situation argument position of a fluent, then that term is $s$ (see [82]). We also introduce a notation $\phi[s]$ to represent the SC formula obtained by restoring situation $s$ back to all the fluents and/or $Poss$ predicates (if any) in $\phi$. It is obvious that $\phi[s]$ is uniform in $s$.

A *basic action theory* (BAT) $\mathcal{D}$ in the SC is a set of axioms written in $\mathcal{L}_{sc}$ with the following five classes of axioms to model actions and their effects [82]: action precondition axioms $\mathcal{D}_{ap}$, successor state axioms (SSAs) $\mathcal{D}_{ss}$, initial theory $\mathcal{D}_{S_0}$,

[1] Reiter [82] uses the notation $s' \sqsubset s$, but we use $s' \preceq s$ to avoid confusion with the inclusion relation $\sqsubseteq$ that is commonly used in description logic literature. In this paper, we use $\sqsubseteq$ to denote the inclusion relation between concepts or roles.

unique name axioms for actions $\mathcal{D}_{una}$, domain independent foundational axioms for situations $\Sigma$.[2]

**Action precondition axioms $\mathcal{D}_{ap}$:** For each action function $A(\vec{x})$, there is an axiom of the form $Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s)$, where $\Pi_A(\vec{x}, s)$ is a formula uniform in $s$ with free variables among $\vec{x}$ and $s$ at most, characterizing the preconditions of action $A$.

**Successor state axioms $\mathcal{D}_{ss}$:** For each relational fluent $F(\vec{x}, s)$, there is an axiom of the form

$$F\big(\vec{x}, do(a, s)\big) \equiv \Phi_F\big(\vec{x}, a, s\big), \tag{1}$$

where $\Phi_F(\vec{x}, a, s)$ is a formula uniform in $s$ with free variables among $\vec{x}, a$ and $s$ at most. It completely characterizes the value of $F$ in the next situation $do(a, s)$ in terms of what holds in the current situation $s$. In fact, the general syntactic form of $\Phi_F(\vec{x}, a, s)$ is

$$\Phi_F\big(\vec{x}, a, s\big) = \gamma_F^+\big(\vec{x}, a, s\big) \vee F\big(\vec{x}, s\big) \wedge \neg\gamma_F^-\big(\vec{x}, a, s\big),$$

where $\gamma_F^+(\vec{x}, a, s)$ ($\gamma_F^-(\vec{x}, a, s)$, respectively) is a formula uniform in $s$ with free variables among $\vec{x}$, $a$ and $s$ at most that completely describes the positive (negative, respectively) effects of actions on fluent $F$. Here and subsequently, we say that an action $\alpha$ has a positive effect on fluent $F$, if $F$ becomes true in the situation resulting from executing this action. Similarly, $\alpha$ has a negative effect, if $F$ becomes false. By using $(\forall a)$ in Eq. 1, Reiter solves the frame problem succinctly because all action functions not explicitly mentioned in $\Phi_F(\vec{x}, a, s)$ have neither positive, nor negative effects on $F$. For all of them, the value of $F$ in $do(a, s)$ remains the same as it was in $s$. Recall that actions in Eq. 1 that cause positive or negative effects can be arbitrary FO terms, not just action constants. In [82], Sections 3.2.4 and 3.2.5, Reiter provides a systematic way of automatically generating SSAs from effect axioms based on the causal completeness assumption and unique name axioms for actions. There, Reiter shows a precise way of constructing the normal form of $\gamma_F^+(\vec{x}, a, s)$ ($\gamma_F^-(\vec{x}, a, s)$, respectively). That is, $\gamma_F^+(\vec{x}, a, s)$ ($\gamma_F^-(\vec{x}, a, s)$, respectively) can be represented as $\bigvee_{i=1}^{h} \Psi_F^{(i)}$ for some finite index $h \geq 0$, where each $\Psi_F^{(i)}$ is a formula of the syntactic form

$$\big[\exists \vec{y}\big]\big(a = A\big(\vec{t}\big) \wedge \psi\big(\vec{x}, \vec{y}, s\big)\big) \tag{2}$$

for some action term $A(\vec{t})$ and some FOL formula $\psi(\vec{x}, \vec{y}, s)$. Note that $\vec{y}$ are those new variables which do not occur in $F(\vec{x}, do(a, s))$, if there are any. If $\vec{y}$ is empty, then there is no quantifier $[\exists \vec{y}]$ at front. Here, $\vec{t}$ is a vector of object terms with free variables (at most) among $\vec{x}$ and the quantified new variables $\vec{y}$ if there are any, $\phi(\vec{x}, \vec{y}, s)$ is uniform in $s$ and its free variables are (at most) among $\vec{x}$ and $\vec{y}$, if there

[2] McIlraith [70], Reiter [82] and Lin [53] illustrate BATs on a variety of examples. The reader can find an example of our modified situation calculus that includes $\mathcal{D}_{ap}$ and $\mathcal{D}_{ss}$ axioms in Example 1 in Section 4. Section 5.2 includes another detailed example that illustrates the expressive power of our modified SC.

are any. In other words, the SSA of fluent $F$ (Eq. 1) has the following syntactic form (integers $m_+, m_- \geq 0$):

$$F(\vec{x}, do(a,s)) \equiv \bigvee_{i=1}^{m_+} [\exists \vec{y}_i](a = PosAct_i(\vec{t}_i) \wedge \phi_i^+(\vec{x}, \vec{y}_i, s)) \vee F(\vec{x}, s)$$
$$\wedge \neg \left( \bigvee_{j=1}^{m_-} [\exists \vec{z}_j](a = NegAct_j(\vec{t'}_j) \wedge \phi_j^-(\vec{x}, \vec{z}_j, s)) \right), \tag{3}$$

where for $i = 1..m_+$ ($j = 1..m_-$, respectively), each $\vec{t}_i$ ($\vec{t'}_j$, respectively) is a vector of terms with free variables (at most) among $\vec{x}$ and the quantified new variables $\vec{y}_i$ ($\vec{z}_j$, respectively) if there are any, each *context condition* $\phi_i^+(\vec{x}, \vec{y}_i, s)$ ($\phi_j^-(\vec{x}, \vec{z}_j, s)$, respectively) is an SC formula uniform in $s$ that has free variables (at most) among $\vec{x}$ and $\vec{y}_i$ ($\vec{z}_j$, respectively) if there are any, and each $PosAct(\vec{t}_i)$ ($NegAct(\vec{t'}_j)$, respectively) is an action term that makes $F(\vec{x}, do(a,s))$ true (false, respectively) if the condition $\phi_i^+(\vec{x}, \vec{y}_i, s)$ ($\phi_j^-(\vec{x}, \vec{z}_j, s)$, respectively) is satisfied.

**Initial theory $\mathcal{D}_{S_0}$:** A set of FO formulas whose only situation term is $S_0$. It specifies the values of all fluents in the initial state. It also describes all the facts that are not changeable by any actions in the domain (static sentences). In particular, it includes unique name axioms for object constants.

**Unique name axioms for actions $\mathcal{D}_{una}$:** Includes axioms specifying that two actions are different if their names are different, and identical actions have identical arguments.[3]

**Foundational axioms for situations $\Sigma$:** The axioms for situations which characterize the basic properties of situations. These axioms are domain independent. They are included in the axiomatization of any dynamical system in the SC (see [82] for details).

Suppose that $\mathcal{D} = \mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \Sigma$ is a BAT, $\alpha_1, \cdots, \alpha_n$ is a sequence of ground action terms, and $G(s)$ is a uniform formula with one free variable $s$. One of the most important reasoning tasks in the SC is the projection problem, that is, to determine whether $\mathcal{D} \models G(do([\alpha_1, \cdots, \alpha_n], S_0))$. Another basic reasoning task is the executability problem. Let $executable(do([\alpha_1, \cdots, \alpha_n], S_0))$ be an abbreviation of the formula $Poss(\alpha_1, S_0) \wedge \bigwedge_{i=2}^{n} Poss(\alpha_i, do([\alpha_1, \cdots, \alpha_{i-1}], S_0))$. Then, the executability problem is to determine whether $\mathcal{D} \models executable(do([\alpha_1, \cdots, \alpha_n], S_0))$. Planning and high-level program execution are two important settings where the executability and projection problems arise naturally. *Regression* is a central computational mechanism that forms the basis for automated solution to the executability and projection tasks in the SC [82]. A recursive definition of the regression operator $\mathcal{R}$ on any *regressable formula* $W$ is given in [82]. A formula $W$ of $\mathcal{L}_{sc}$ is *regressable* iff (1) every term of sort situation in $W$ is starting from $S_0$ and has the syntactic form $do([\alpha_1, \cdots, \alpha_n], S_0)$, where each $\alpha_i$ is of sort action; (2) for every atom of the form $Poss(\alpha, \sigma)$ in $W$, $\alpha$ has the syntactic form $A(t_1, \cdots, t_n)$ for some $n$-ary function symbol $A$ of $\mathcal{L}_{sc}$; and (3) $W$ does not quantify over situations, and does not mention

[3] For the second type of axioms, we use $A(x_1, \cdots, x_n) = A(y_1, \cdots, y_n) \equiv x_1 = y_1 \wedge \cdots \wedge x_n = y_n$.

the relation symbols "$\prec$" or "=" between terms of situation sort. For a regressable formula $W$, we use notation $\mathcal{R}[W]$ to denote the regressed formula that results from eliminating *Poss* atoms in favor of their definitions as given by action precondition axioms, and replacing fluent atoms about $do(\alpha, s)$ by logically equivalent expressions about $s$ as given by SSAs repeatedly, until no more such replacement can be made. The formula $G(do([\alpha_1, \cdots, \alpha_n], S_0))$ is a particularly simple example of a regressable formula because it is uniform in $do([\alpha_1, \cdots, \alpha_n], S_0))$, but generally, regressable formulas can mention several different situation terms. Roughly speaking, the regression of a regressable formula $W$ through an action $a$ is a formula $W'$ that holds prior to $a$ being performed iff $W$ holds after $a$. Both precondition axioms and SSAs support regression in a natural way and are no longer needed when regression terminates. This is because each step of regression either eliminates a *Poss* atom by replacing it with an equivalent formula, or replaces a fluent with a compound situation term by a logically equivalent formula with a situation term that has one less occurrence of an action term.

The regression theorem proved in [78] shows that one can reduce the evaluation of a regressable sentence $W$ to an FOL theorem proving task in the initial theory together with unique names axioms for actions:

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

This fact is the key result for our paper: it demonstrates that an executability or a projection task can be reduced to an FOL theorem proving task. However, because $\mathcal{D}_{S_0}$ is an arbitrary FO theory, this type of reasoning is undecidable. Two of the most common ways to overcome this difficulty are to introduce the closed world assumption or introduce the domain closure assumption (i.e., assume the domain is finite). In many practical application domains these assumptions are unrealistic. Therefore, we propose a version of the SC with $\mathcal{D}_{S_0}$ based on $C^2$, or on a weaker fragment of FO$^2$. Fragments of FO$^2$ that are syntactic versions of DLs are particularly interesting because for them the satisfiability problem is more tractable than for a general FO$^2$. For this reason, in the next section, we review definitions and results relevant to DLs, FO$^2$ and $C^2$.

## 3 Description logics and two-variable first-order logic

In this section we review a few popular expressive description logics [5] and related fragments of FOL [12].

### 3.1 Description logics

We start with the language of logic $\mathcal{ALCQIO}$. Let $N_C = \{AC_1, AC_2, \ldots\}$ be a non-empty set of *atomic concepts* and $N_R = \{R_1, R_2, \ldots\}$ be a non-empty set of *atomic roles*. In $\mathcal{ALCQIO}$, *nominals* are allowed. Nominals are singleton concepts obtained by picking one of the object names. An $\mathcal{ALCQIO}$ role is either some $R \in N_R$ or an *inverse role* $R^-$ for $R \in N_R$. In addition, $(R^-)^-$ is $R$ itself. The set of $\mathcal{ALCQIO}$ concepts is the minimal set built inductively from $N_C$ and $\mathcal{ALCQIO}$ roles using the following rules: all $AC \in N_C$ are concepts, nominals are concepts, and, if $C$, $C_1$, and $C_2$ are $\mathcal{ALCQIO}$ concepts, $R$ is a role and $n \in \mathbb{N}$, then $\neg C$, $C_1 \sqcap C_2$, and $\geqslant n R.C$ are

also $\mathcal{ALCQIO}$ concepts. Concepts that are not atomic are called *complex*. A *literal* concept is a possibly negated concept name. The abbreviations for complex concepts such as $C_1 \sqcup C_2$, $\leqslant nR.C$, $\exists\, R.\, C$, $\forall\, R.\, C$ (and other complex concepts) can be easily defined. For example,

$$C_1 \sqcup C_2 \stackrel{def}{=} \neg(\neg C_1 \sqcap \neg C_2) \qquad \exists^{=n}\, R.\, C \stackrel{def}{=} (\leqslant nR.C) \sqcap (\geqslant nR.C)$$
$$\exists R.C \stackrel{def}{=} \geqslant 1 R.C \qquad \top \stackrel{def}{=} AC \sqcup \neg AC \text{ for some } AC \in N_C$$
$$\forall R.C \stackrel{def}{=} \neg\exists R.\neg C \qquad \bot \stackrel{def}{=} AC \sqcap \neg AC \text{ for some } AC \in N_C$$
$$\leqslant nR.C \stackrel{def}{=} \neg(\geqslant (n+1) R.C)$$

The semantics of description terms is given denotationally, using the notion of an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, (\cdot)^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is a domain (non-empty universe) of objects, and $(\cdot)^{\mathcal{I}}$ maps from atomic concept names to subsets of the domain (i.e., $AC^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for all $AC \in N_C$), and atomic role names to sets of pairs over the domain (i.e., $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for all $R \in N_R$). Moreover, the interpretation function $(\cdot)^{\mathcal{I}}$ is extended recursively to composite descriptions in Table 1 (see Appendix A). We assume that the interpretation of nominals has to respect the unique name assumption.

A TBox $\mathcal{T}$ is a finite set of *equality* axioms $C_1 \equiv C_2$.[4] An equality with an atomic concept on the left-hand side (LHS) is a concept *definition*. In the sequel, we always consider TBox axioms set $\mathcal{T}$ that is a *terminology*: a finite set of concept definition formulas with unique left-hand sides, i.e., no atomic concept occurs more than once as a left-hand side. We say that a *defined* concept name $C_1$ *directly uses* a concept name $C_2$ wrt $\mathcal{T}$ if $C_1$ is defined by a concept definition axiom in $\mathcal{T}$ with $C_2$ occurring on the right-hand side (RHS) of the axiom. Let *uses* be the transitive closure of directly uses, and a TBox axioms set $\mathcal{T}$ is *acyclic* if no concept name uses itself wrt $\mathcal{T}$. An ABox $\mathcal{A}$ is a finite set of assertions of the forms $C(b)$ and $R(b, b_1)$, where $b$ and $b_1$ are some object names, $C$ is a concept, and $R$ is a role. An RBox $\mathcal{H}_R$, called a *role hierarchy*, is a finite set of role inclusion axioms of the form $R_1 \sqsubseteq R_2$, where $R_1$ ($R_2$, respectively) is either an atomic role or its inverse. A DL knowledge base $KB$ in DL is a tuple $(\mathcal{T}, \mathcal{A})$ (or, a triple $(\mathcal{H}_R, \mathcal{T}, \mathcal{A})$ if role inclusion axioms are allowed). The semantics of terminological and assertional axioms are provided in Table 3 (see Appendix A).

The logic $\mathcal{ALCQI}$ is obtained by disallowing nominals in $\mathcal{ALCQIO}$. The logic $\mathcal{ALCQO}$ is obtained by disallowing inverse roles in $\mathcal{ALCQIO}$. The logic $\mathcal{ALCQO}(U)$ is obtained by adding the universal role $U$ to $\mathcal{ALCQO}$. The semantics of $U$ is given in Table 2 (see Appendix A). The logic $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ is obtained from $\mathcal{ALCQIO}$ by introducing concept identity $id(C)$ (relating each individual in $C$ with itself), and allowing complex role expressions: if $R_1$, $R_2$ are $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ roles and $C$ is a concept, then $R_1 \sqcup R_2$, $R_1 \sqcap R_2$, $\neg R_1$, $R_1^-$ and $R_1|_C$ are $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ roles too. These complex roles can be used in constructing complex concepts. The semantics of complex roles are given in Table 2 (see Appendix A). Subsequently, we call a role $R$ *primitive* if it is either $R \in N_R$ or it is an *inverse role* $R^-$ for $R \in N_R$. The logic $\mathcal{ALCHQIO}(\sqcup, \sqcap, \neg, |, id)$ allows RBox axioms based on the language of $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$. Moreover, notice that the universal role can be implicitly

[4] Sometimes, a TBox can include *general inclusion axioms* of the form $C_1 \sqsubseteq C_2$. However, we do not consider general inclusion axioms for concepts in this paper.

constructed in $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$: $U$ can be replaced using $R \sqcup \neg R$ for any $R \in N_R$.

There are different reasoning tasks in DLs, such as the *concept satisfiability problem* and the *ABox consistency problem*, etc. The concept satisfiability problem is to decide given a TBox $\mathcal{T}$ and a concept $C$, whether there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}}$ is nonempty. Given any DL KB $(\mathcal{T}, \mathcal{A})$, the ABox consistency problem is to check whether there is a interpretation that is a model for both $\mathcal{T}$ and $\mathcal{A}$. The complexity of solving the concept satisfiability problem or the ABox consistency problem has been studied for different versions of DLs [100]. For example, for any DL KB that has acyclic terminologies only, it has been shown that the complexities of solving these two problems are PSPACE-complete in $\mathcal{ALC}$, in $\mathcal{ALCO}$ or in $\mathcal{ALCQO}$ [4, 89, 95], but are EXPTIME-complete in $\mathcal{ALCQIO}$ [94, 95], in $\mathcal{ALC}(U)$ (i.e., $\mathcal{ALC}$ plus the universal role) [39, 60, 91] or in $\mathcal{SHOQ}$ [43]. Because one can implicitly represent the universal role in $\mathcal{SHOQ}$, it is also known that the complexity of solving these two problems is still EXPTIME-complete in $\mathcal{ALCO}(U)$ or in $\mathcal{ALCQO}(U)$ for any DL KB that has acyclic terminologies only. Note that for more general TBoxes, the complexities of solving these two problems are EXPTIME-complete in $\mathcal{ALC}$, in $\mathcal{ALCO}$ or in $\mathcal{ALCQO}$ [84, 86, 100].

### 3.2 $C^2$ and its relationship to description logics

The *two-variable FOL* $\text{FO}^2$ [33, 40] is a well-known fragment of ordinary FOL, whose formulas can be built with the help of predicate symbols (including equality) and constant symbols (but without general function symbols) using no more than two variable symbols $x$ and $y$ (free or bound). Note that each variable can be reused arbitrarily often. The *two-variable FOL with counting* $C^2$ extends $\text{FO}^2$ by allowing FO counting quantifiers $\exists^{\geq m}$ and $\exists^{\leq m}$ for all $m \geq 1$ [33, 47, 76, 77, 80]. Because the semantics of $\text{FO}^2$ ($C^2$, respectively) are the same as the semantics of FOL, the details are omitted here. It is well-known that modal logic is a "big brother" of DLs. In particular, it is proved that the DL $\mathcal{ALC}$ is a notational variant of a basic multi-modal logic $\mathcal{K}$ [86]. For this reason, it is important to note that the *standard translation* from a basic modal logic to FOL is proposed in [7]. Later, it was realized that a basic modal logic can be translated to $\text{FO}^2$ [8, 25]. The standard translation and other results in modal logic are extensively discussed in [10] and in [75].

Now we consider some relationships between $C^2$ and DLs. In [12], an expressive description logic $\mathcal{B}$ is defined[5], and it is shown that $C^2$ and the language $\mathcal{B}$ are *equally expressive*, that is, $C^2$ is *as expressive as* $\mathcal{B}$ and vice versa. Generally speaking, a language $\mathcal{L}_2$ is *as expressive as* language $\mathcal{L}_1$, if there is a translation function *transl* from $\mathcal{L}_1$ to $\mathcal{L}_2$ such that for every sentence $l$ in $\mathcal{L}_1$, *transl*$(l)$ *expresses the meaning* of $l$ [12]. According to [12], the *meanings* for sentences in both languages are defined uniformly as mappings from interpretations to sets of tuples over $\Delta^{\mathcal{I}}$. One can therefore say that description $D$ (a DL formula built using concepts and roles using DL constructors) has the same meaning as formula $\Psi(\vec{x})$ iff $\lambda\mathcal{I}.D^{\mathcal{I}} = \lambda\mathcal{I}.\Psi(\vec{x})^{\mathcal{I}}$,

[5] In [12], the language $\mathcal{B}$ is denoted as $\mathcal{DL} - \{\textbf{trans}, \textbf{compose}\}$, in which **trans** represents the role constructor *reflexive-transitive closure* and **compose** represents the role constructor *composition*. We change it to notation $\mathcal{B}$ in order to avoid confusion. Besides, the syntax and semantics of reflexive-transitive closure and composition can be found in Table 2, Appendix A.

where $\mathcal{I}$ are interpretations over the same schema of the two languages. One can also create hybrid sentences, which mix the two kinds of formulas. On the one hand, one can allow descriptions to appear as monadic and dyadic predicates in FOL formulas, interpreting $[D(a)]^{\mathcal{I}}$, where $D$ is a concept description, as $a^{\mathcal{I}} \in D^{\mathcal{I}}$; in this formulation, $\Psi(x)$ expresses the meaning of $D$ iff $\forall x.D(x) \equiv \Psi(x)$ is a theorem. On the other hand, we can treat $\Psi(x)$ as a description by interpreting it using $(\cdot)^{\mathcal{I}}$ whenever we encounter it, in which case $\Psi(x)$ expresses the meaning of $D$ iff $D$ implies $\Psi(x)$ and $\Psi(x)$ implies $D$. Similar hybrid formulas can be set up for the other kinds of judgments one is normally interested in for DLs. Further details can be found in [12]. Moreover, in [12], the translation in both directions between $C^2$ and $\mathcal{B}$ leads to no more than a linear increase in the size of the translated formula. His translation is similar to the standard translation earlier proposed for modal logics. Using the same approach as in [12], we prove the following theorem (the detailed proof is provided in Appendix B.1).

**Theorem 1** *The description logic $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ and $C^2$ are equally expressive (i.e., each sentence in language $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ can be translated to a sentence in $C^2$, and vice versa). In addition, translation in both directions leads to no more than a linear increase in the size of the translated formula.*

Notice that $\mathcal{ALCHQIO}(\sqcup, \sqcap, \neg, |, id)$ includes RBox in the knowledge bases, in contrast to $\mathcal{ALCQI}(\sqcup, \sqcap, \neg, id)$ that has no RBox . However, it is obvious that every axiom in RBox still can be translated into a sentence in $C^2$. Hence, the language of $\mathcal{ALCHQIO}(\sqcup, \sqcap, \neg, |, id)$ and $C^2$ are also equally expressive. In [61], another DL is considered[6] and alternative translation between that DL and $FO^2$ is proposed. It is proved that translation from $FO^2$-formulae into concepts in the considered DL involves an exponential blow-up in formula size.

This statement has an important consequence. Gradel et al. [34] and Pacholski et al. [76] show that the satisfiability problem for $C^2$ is decidable and recently in [80] it is proved that this problem is in NExpTime even when counting quantifiers are coded succinctly. Hence, the satisfiability and/or subsumption problems of concepts w.r.t. an acyclic or empty TBox in description logic $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ ($\mathcal{ALCHQIO}(\sqcup, \sqcap, \neg, |, id)$, respectively) is also decidable with the same computational complexity.[7] See additional background on DLs and discussion of connections between DLs with $C^2$ in [5, 12].

## 4 Modeling dynamical systems in a modified situation calculus

In this section, we consider dynamical systems formulated in a modification of the language of the SC so that it can be considered as an extension to $C^2$ (with an

[6] $\mathcal{ALC}$ extended with full Boolean operators on roles, the inverse operator on roles and an identity role.

[7] In [5], it is shown that the satisfiability problem and the subsumption problem can be reduced to each other; moreover, if a TBox $\mathcal{T}$ is acyclic, the reasoning problems w.r.t. $\mathcal{T}$ can always be reduced to problems w.r.t. the empty TBox.

additional situation argument).[8] The key idea is to consider a syntactic modification of the SC such that the executability and projection problems are guaranteed to be decidable as a consequence of the decidability of the satisfiability problem in $C^2$. We will denote this language $\mathcal{L}_{sc}^{C^2}$.

Firstly, the three sorts in $\mathcal{L}_{sc}^{C^2}$ (i.e., *action*, *situation* and *object*) are the same as those in $\mathcal{L}_{sc}$, except that they obey the following restrictions: (1) all terms of sort *object* are variables ($x$ and $y$) or constants, i.e., object functional symbols are *not* allowed; (2) all action functions include no more than two arguments. Each argument of any term of sort *action* is either a constant or an *object* variable ($x$ or $y$); (3) variable $s$ of sort *situation* and/or variable $a$ of sort *action* are the only additional variables allowed in $\mathcal{D}$ in addition to variables $x$, $y$.

Secondly, any fluent considered in $\mathcal{L}_{sc}^{C^2}$ is a predicate with either two or three arguments (including the one of sort *situation*). We call fluents with two arguments *dynamic concepts*, and call fluents with three arguments *dynamic roles*. Intuitively, each dynamic concept in $\mathcal{L}_{sc}^{C^2}$, say $F(x, s)$ with variables $x$ and $s$ only, can be considered as a changeable concept $F$ in a dynamical system specified in $\mathcal{L}_{sc}^{C^2}$; the truth value of $F(x, s)$ could vary from one situation to another. Similarly, each dynamic role in $\mathcal{L}_{sc}^{C^2}$, say $F(x, y, s)$ with variables $x$, $y$ and $s$, can be considered as a changeable role $R$ in a dynamical system specified in $\mathcal{L}_{sc}^{C^2}$; the truth value of $F(x, y, s)$ could vary from one situation to another. In $\mathcal{L}_{sc}^{C^2}$, *(static) concepts* (i.e., monadic predicates with no situation argument) and *(static) roles* (i.e., dyadic predicates with no situation argument), if any, are considered as unchangeable taxonomic properties and unchangeable classes of an application domain. Moreover, each concept (static or dynamic) can be either *primitive* or *defined*.

For each primitive dynamic concept, an SSA must be provided in the basic action theory for a given domain. Because defined dynamic concepts are expressed in terms of primitive concepts by axioms in an acyclic TBox , SSAs for them are not provided. In addition, SSAs are provided for dynamic roles.

Thirdly, apart from the standard FO logical symbols $\wedge$, $\vee$ and $\exists$, with the usual definition of a full set of connectives and quantifiers, $\mathcal{L}_{sc}^{C^2}$ also includes counting quantifiers $\exists^{\geq m}$ and $\exists^{\leq m}$ for all $m \geq 1$. Equality $=$ is allowed in $\mathcal{L}_{sc}^{C^2}$ too.

The dynamical systems we are dealing with here satisfy the *open world assumption* (OWA): what is not stated explicitly in an initial theory $\mathcal{D}_{S_0}$ is unknown rather than false. In this paper, the dynamical systems we are interested in can be formalized as a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ (also called $\mathcal{L}_{sc}^{C^2}$-restricted BAT below) using the following seven groups of axioms in $\mathcal{L}_{sc}^{C^2}$: $\mathcal{D} = \Sigma \cup \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_T \cup \mathcal{D}_R \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}$.[9] Five of them ($\Sigma, \mathcal{D}_{ap}, \mathcal{D}_{ss}, \mathcal{D}_{una}, \mathcal{D}_{S_0}$) are similar to those groups in a BAT in $\mathcal{L}_{sc}$, and the other two ($\mathcal{D}_T, \mathcal{D}_R$) are introduced to axiomatize description logic related facts and properties (see below). However, because $\mathcal{L}_{sc}^{C^2}$ allows only one or two object variables, all axioms must conform to the following additional requirements.[10]

[8] The reason that we call it a "modified" SC rather than a "restricted" SC is that we extend the SC with other features, such as adding acyclic TBox axioms to BATs.

[9] An example of a BAT is provided at the end of this section. Another detailed example is included in Section 5.2 below.

[10] Subsequently, we write axioms with action and situation variables, and use action and situation terms. However, we will see that they can be eliminated when we solve the projection problem.

**Action precondition axioms $\mathcal{D}_{ap}$:** For each action function $A(\vec{x})$ in $\mathcal{L}_{sc}^{C^2}$, where $\vec{x}$ can be either empty, $x$, or $\langle x, y\rangle$ if $A$ is a 0-ary, 1-ary, or 2-ary action function symbol respectively, there is an axiom of the form

$$Poss\big(A(\vec{x}), s\big) \equiv \Pi_A(\vec{x})[s], \tag{4}$$

where $\Pi_A(\vec{x})$ is a $C^2$ formula with free variables at most among $\vec{x}$ and $\Pi_A(\vec{x})[s]$ represents the result of situation-suppressed formula $\Pi_A(\vec{x})$ with situation term $s$ restored. This set of axioms characterizes the preconditions of all actions.

**Successor state axioms $\mathcal{D}_{ss}$:** Let variable vector $\vec{x}$ be either $x$, $y$ or $\langle x, y\rangle$. An SSA is specified for each dynamic primitive concept that is not defined in TBox (see below) and for each dynamic role. According to the approach of constructing SSAs provided in [82], without loss of generality, we can assume that the SSA of $F(\vec{x}, s)$ has the form

$$\begin{aligned} F\big(\vec{x}, do(a, s)\big) \equiv & \bigvee_{i=1}^{m_+} \big[\exists x\big]\big[\exists y\big]\big(a = PosAct_i\big(\vec{t}_{(i,+)}\big) \wedge \psi_i^+\big(\vec{x}_{(i,+)}\big)[s]\big) \vee F\big(\vec{x}, s\big) \\ & \wedge \neg \left( \bigvee_{j=1}^{m_-} \big[\exists x\big]\big[\exists y\big]\big(a = NegAct_j\big(\vec{t}_{(j,-)}\big) \wedge \psi_j^-\big(\vec{x}_{(j,-)}\big)[s]\big) \right). \end{aligned} \tag{5}$$

Here, each vector $\vec{t}_{(i,+)}$, $i = 1..m_+$, ($\vec{t}_{(j,-)}$, $j = 1..m_-$, respectively) represents a vector of object terms appearing in the corresponding action term, which can be either empty, $O$, $\langle O_1, O_2\rangle$, $x$, $\langle x, x\rangle$, $\langle O, x\rangle$, $\langle x, O\rangle$, $y$, $\langle y, y\rangle$, $\langle O, y\rangle$, $\langle y, O\rangle$ $\langle x, y\rangle$ or $\langle y, x\rangle$ for free variables $x$, $y$ and some object constants $O$, $O_1$, $O_2$. Each variable vector $\vec{x}_{(i,+)}$ (or $\vec{x}_{(j,-)}$, respectively), $i = 1..m_+$, $j = 1..m_-$, represents a vector of free variables appearing in the corresponding context condition, which can be either empty, $x$, $y$, $\langle x, y\rangle$ or $\langle y, x\rangle$. Moreover, $[\exists x]$ or $[\exists y]$ represents that the quantifier included in [ ] is optional; and each $\psi_i^+(\vec{x}_{(i,+)})$, $i = 1..m_+$, ($\psi_i^-(\vec{x}_{(j,-)})$, $j = 1..m_-$, respectively), is a $C^2$ formula with variables (both free and bound) among $x$ and $y$ at most. Note that when $m_+$ (or $m_-$, respectively) equals to 0, the corresponding disjunctive sub-formula is equivalent to *false*.

**Acyclic TBox axioms $\mathcal{D}_T$:** Similar to the TBox axioms in DLs, we may define new concepts using TBox axioms. Any group of TBox axioms $\mathcal{D}_T$ may include two subclasses: static TBox $\mathcal{D}_{T,st}$ and dynamic TBox $\mathcal{D}_{T,dyn}$. Every formula in static TBox is a *concept definition* formula of the form

$$G(x) \equiv \phi_G(x), \tag{6}$$

where $G$ is a monadic predicate symbol and $\phi_G(x)$ is a $C^2$ formula with a free variable $x$, and there is no fluent in it. Every formula in dynamic TBox is a *concept definition* formula of the form

$$G(x, s) \equiv \phi_G(x)[s], \tag{7}$$

where $\phi_G(x)$ is a $C^2$ formula with free variable $x$, and there is at least one fluent in it. All the concepts appeared on the left-hand side of TBox axioms are called *defined* concepts. We also require that the set of TBox axioms must be acyclic (acyclicity in $\mathcal{D}_T$ is defined exactly as it is defined for TBox). Note that the defined dynamic

concepts are not provided with SSAs. There is no need to provide an SSA for a defined concept because regression can expand TBox definitions instead of an SSA.

**RBox axioms $\mathcal{D}_R$:** Similar to the idea of RBox in DLs, we may also specify a group of axioms, called RBox axioms below, to support a role taxonomy. Each role inclusion axiom is represented as

$$R_1(x, y)[s] \supset R_2(x, y)[s], \tag{8}$$

where $R_1$ and $R_2$ are primitive roles (either static or dynamic). If these axioms are included in the BAT $\mathcal{D}$, then it is assumed that $\mathcal{D}$ is specified correctly in the sense that the meaning of any RBox axiom included in the theory is correctly compiled into SSAs. This means that an axiomatizer is responsible for writing $\mathcal{D}_{ss}$ such that axioms from RBox become logical consequences. That is, it should be provable by induction that

$$\left(\mathcal{D} - \mathcal{D}_R\right) \models \forall s. R_1(x, y)[s] \supset R_2(x, y)[s]$$

for any axiom $R_1(x, y)[s] \supset R_2(x, y)[s]$ in $\mathcal{D}_R$. This is the common approach to state constraints, e.g., it was taken in [82]. In some special (but realistic) cases, RBox axioms can be automatically compiled into $\mathcal{D}_{ss}$. Let us say $R_2(x, y, s)$ directly depends on $R_1(x, y, s)$, if $R_1(x, y, s) \supset R_2(x, y, s)$ belongs to RBox, and say $R_3(x, y, s)$ depends on $R_1(x, y, s)$, if $R_3(x, y, s)$ directly depends on $R_2(x, y, s)$, and $R_2(x, y, s)$ depends on $R_1(x, y, s)$. Then, we can say that RBox is acyclic, if there is no dynamic role $R(x, y, s)$ that depends on itself. In [70], it is proved that acyclic state constraints can be automatically compiled into SSAs. Because acyclic RBox is just a special case of state constraints considered in McIlraith's paper, her approach is applicable to acyclic TBox as well. Additional details related to state constraints in the SC can be found in [52, 54]. Although RBox axioms are not used by the regression operator, they are used for taxonomic reasoning in the initial theory.

**Initial theory $\mathcal{D}_{S_0}$:** It is a finite or countably infinite set of $C^2$ sentences that are uniform in $S_0$. It specifies the incomplete information about the initial problem state and also describes all the facts that are not changeable over time in the domain of an application. In particular, it includes static TBox axioms $\mathcal{D}_{T,st}$ as well as RBox axioms in the initial situation $S_0$ (if any). In addition, $\mathcal{D}_{S_0}$ also includes all unique name axioms for object constants. Note that this definition of $\mathcal{D}_{S_0}$ includes ABox as a special case. In the sequel, $\mathcal{D}_{S_0}$ is assumed to be finite, unless stated otherwise.

The remaining two classes (foundational axioms for situations $\Sigma$ and unique name axioms for actions $\mathcal{D}_{una}$) are the same as those in the BATs of the usual SC. Note that these axioms (as well as $\mathcal{D}_{ap}$ and $\mathcal{D}_{ss}$) use more than two variables (e.g., $\mathcal{D}_{ss}$ use action and situation variables in addition to object variables), but we will see in the next section, that these axioms will be eliminated in the process of regressing a regressable sentence to a sentence that will use no more than two object variables and no other variables. We complete this section with an example of simple web services for online shopping that illustrates all classes of axioms. Another example is provided in Section 5.2.

*Example 1* Consider a web services system in which clients can purchase CDs and books online with credit cards. The high-level features are specified as concepts and roles:

- Primitive static concepts: $person(x)$ ($x$ is a person), $cd(x)$ ($x$ is a CD), $book(x)$ ($x$ is a book), $creditCard(x)$ ($x$ is a credit card).
- Defined static concept: $client(x)$ ($x$ is a client).
- Primitive dynamic concept: $instore(x, s)$ ($x$ is in store in situation $s$).
- Defined dynamic concept: $valuableCustomer(x, s)$ ($x$ is a valuable customer in situation $s$).
- Static role: $has(x, y)$ ($x$ has $y$).
- Dynamic roles: $boughtCD(x, y, s)$ ($x$ bought CD $y$ in situation $s$), $boughtBook(x, y, s)$ ($x$ bought book $y$ in situation $s$), $bought(x, y, s)$ ($x$ bought $y$ in situation $s$.

Web services are specified as actions: $buyCD(x, y)$ ($x$ buys CD $y$), $buyBook(x, y)$ ($x$ buys book $y$), $returnCD(x, y)$ ($x$ returns CD $y$), $returnBook(x, y)$ ($x$ returns book $y$), $order(x)$ (the web service agent orders $x$ from the publisher).

The BAT of the system is as follows (most of the axioms are self-explanatory).

**Precondition Axioms:**

$$\begin{aligned}
&Poss(buyCD(x, y), s) &&\equiv client(x) \wedge cd(y) \wedge instore(y, s),\\
&Poss(buyBook(x, y), s) &&\equiv client(x) \wedge book(y) \wedge instore(y, s),\\
&Poss(returnCD(x, y), s) &&\equiv boughtCD(x, y, s),\\
&Poss(returnBook(x, y), s) &&\equiv boughtBook(x, y, s),\\
&Poss(order(x), s) &&\equiv book(x) \vee cd(x).
\end{aligned}$$

**Successor State Axioms:**

$$\begin{aligned}
&instore(x, do(a, s))\\
&\quad \equiv (\exists y)(a = returnCD(y, x)) \vee (\exists y)(a = returnBook(y, x)) \vee a = order(x)\\
&\quad \vee instore(x, s) \wedge \neg\big((\exists y)(a = buyCD(y, x)) \vee (\exists y)(a = buyBook(y, x))\big),
\end{aligned}$$

$$boughtCD(x, y, s) \equiv a = buyCD(x, y) \vee boughtCD(x, y, s) \wedge a \neq returnCD(x, y),$$

$$\begin{aligned}
boughtBook(x, y, s) \equiv\ & a = buyBook(x, y) \vee boughtBook(x, y, s)\\
& \wedge a \neq returnBook(x, y),
\end{aligned}$$

$$\begin{aligned}
bought(x, y, s) \equiv\ & a = buyCD(x, y) \vee a = buyBook(x, y) \vee bought(x, y, s)\\
& \wedge \neg\, (cd(y) \wedge a = returnCD(x, y) \vee book(y)\\
& \wedge a = returnBook(x, y)).
\end{aligned}$$

**Acyclic TBox Axioms:**

Dynamic ones: $valuableCustomer(x, s) \equiv person(x) \wedge \exists^{\geq 3} y.(bought(x, y, s))$.
Static ones: $client(x) \equiv person(x) \wedge (\exists y)(has(x, y) \wedge creditCard(y))$.

**RBox Axioms:** $boughtCD \sqsubseteq bought$, $boughtBook \sqsubseteq bought$.

## 5 Reasoning about actions via regression

After giving the definition of what is an $\mathcal{L}_{sc}^{\mathcal{C}^2}$-restricted BAT in $\mathcal{L}_{sc}^{\mathcal{C}^2}$, we turn our attention to the reasoning tasks. We want to identify reasoning problems that are decidable in $\mathcal{L}_{sc}^{\mathcal{C}^2}$. To achieve this goal, for certain type of formulas in $\mathcal{L}_{sc}^{\mathcal{C}^2}$, we expect the regressed formulas to be $\mathcal{C}^2$ formulas.

### 5.1 Modified regression with lazy unfolding

Given a formula $W$ of $\mathcal{L}_{sc}^{\mathcal{C}^2}$ in the domain $\mathcal{D}$, the definition of $W$ being regressable (called $\mathcal{L}_{sc}^{\mathcal{C}^2}$ *regressable* below) is slightly different from the definition of $W$ being regressable in $\mathcal{L}_{sc}$ (see [82]).

**Definition 1** A formula $W$ of $\mathcal{L}_{sc}^{\mathcal{C}^2}$ is $\mathcal{L}_{sc}^{\mathcal{C}^2}$ *regressable* iff

(1) Each term of sort *situation* in $W$ is ground starting from $S_0$, and has the syntactic form $do([\alpha_1, \cdots, \alpha_n], S_0)$, where each $\alpha_i$ is ground action term in the language of $\mathcal{L}_{sc}^{\mathcal{C}^2}$.
(2) Other than the action terms occurred in atoms of the form $Poss(\alpha, \sigma)$ and in situation terms, there are no function terms in $W$. Moreover, variables $x$ and $y$ (free or bound) are the only variables used in $W$, if any.
(3) For every atom of the form $Poss(\alpha, \sigma)$ in $W$, $\alpha$ has the syntactic form $A(t_1, \cdots, t_n)$ for some $n$-ary function symbol $A$ of $\mathcal{L}_{sc}^{\mathcal{C}^2}$, where $n \leq 2$. Moreover, each $t_i$ is either variable $x$, variable $y$ or some constant $O$ if there is any.
(4) $W$ does not mention the relation symbols "$\prec$" or "$=$" between terms of sort *situation*.

Definition 1 has several additional restriction conditions comparing to the regressable formulas defined in [82] (the definition is reviewed in Section 2). The condition that $W$ does not quantify over situation is ensured by condition (1). The requirements of conditions (2) and (3) are obvious, because our language is restricted to $\mathcal{L}_{sc}^{\mathcal{C}^2}$. The intuition for requiring all situation terms to be ground in condition (1) is as follows. Consider the following counterexample.

*Example 2* Consider an $\mathcal{L}_{sc}^{\mathcal{C}^2}$-restricted BAT $\mathcal{D}$, which includes an SSA

$$F(x, do(a, s)) \equiv a = A(x) \wedge \big(\exists y.G(x, y, s)\big) \vee F(x, s) \tag{9}$$

for some fluents $F(x, s)$, $G(x, y, s)$ and action function $A(x)$. Consider a formula

$$\forall x.\exists y.F(x, do(A(O), do(A_1(y), S_0))) \ \big(\text{denoted as } W_0 \text{ below}\big)$$

of $\mathcal{L}_{sc}^{\mathcal{C}^2}$. To perform a correct regression on $W_0$ in the sense that the formula resulting from regression should be logically equivalent to $W_0$ w.r.t. $\mathcal{D}$, we have to rename the quantified variable $y$ in Eq. 9 so that it is different from any variables appearing in $W_0$. That is, according to Eq. 9, the one step regression on $F$ using Reiter's regression operator $\mathcal{R}$ (the definition is reviewed in Section 2) should be

$$\forall x.\exists y.\mathcal{R}\left[A(O) = A(x) \wedge \exists z.G\big(x, z, do\big(A_1(y), S_0\big)\big) \vee F\big(x, do\big(A_1(y), S_0\big)\big)\right].$$

Then, the regressed formula is no longer a formula of language $\mathcal{L}_{sc}^{C^2}$. Otherwise, if we do not rename the quantified variable $y$ in Eq. 9 to assure the regressed formula is still of language $\mathcal{L}_{sc}^{C^2}$, then the one step regression on $F$ using Eq. 9 without renaming will result in the following formula:

$$\forall x.\exists y.\mathcal{R}\left[A(O) = A(x) \wedge \exists y.G\big(x, y, do\big(A_1(y), S_0\big)\big) \vee F\big(x, do\big(A_1(y), S_0\big)\big)\right].$$

It is obvious that the above regressed formula is not logically equivalent to $W_0$, because the variable $y$ that occurs in the situation term $A_1(y)$ should not be quantified by $\exists y$ at the front of $G$.

Hence, to avoid the problem described in Example 2, we require all situation terms to be ground in Definition 1. Below, with a carefully defined regression operator, we are able to show that for every $\mathcal{L}_{sc}^{C^2}$ regressable formula, there is an equivalent regressed $C^2$ formula uniform in $S_0$.

In the language of $\mathcal{L}_{sc}^{C^2}$, we have to be more careful with the definition of the *modified* regression operator, denoted $\mathcal{R}$ below, for two main reasons. First, to deal with TBox we have to extend regression. For an $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$, we *extend* the regression operator defined in [82] with the *lazy unfolding technique* (see [5]) to expand defined dynamic concepts. Second, $\mathcal{L}_{sc}^{C^2}$ uses only two object variables and we have to make sure that after regressing a fluent atom we still get an $\mathcal{L}_{sc}^{C^2}$ formula, i.e., that we never need to introduce new (free or bound) object variables. To deal with the two-variable restriction, we modify our regression operator in comparison to the conventional operator defined in [82], and still denote this operator as $\mathcal{R}$. The key idea is to *reuse* variables when doing replacement. For example, when replacing *Poss* atoms or fluent atoms about $do(\alpha, \sigma)$, the definition of the conventional regression operator in [82] has the assumption that the quantified variables on the right-hand side (RHS) of the corresponding axioms should be renamed to new variables different from the free variables in the atoms to be replaced. This assumption of using *new* variables for renaming ensures logical equivalence of the original formula and the formula after regression. But in $C^2$ new variables cannot be used. To avoid introducing new variables (as required by Reiter's regression operator) and to assure defined dynamic concepts being handled, we modify the regression operator. The possibility of reusing variables is guaranteed by the general format of the axioms in $\mathcal{L}_{sc}^{C^2}$-restricted BATs, given in the previous section and the additional conditions in Definition 1.

The complete formal definition of our modified regression operator $\mathcal{R}$ is as follows,[11] where $\sigma$ denotes the term of sort *situation*, and $\alpha$ denotes the term of sort *action*. Note that below, if $\Phi(\vec{x})$ represents a formula $\Phi$ whose free variables are at most among a variable vector $\vec{x}$, then for any vector of terms $\vec{t}$ such that $|\vec{t}| = |\vec{x}|$ (i.e., the number of variables in $\vec{t}$ is the same as the number of variables in $\vec{x}$), $\Phi(\vec{t})$ represents the resulting formula obtained by substituting each $x_i$ in vector $\vec{x}$ with $t_i$ in vector $\vec{t}$ if $x_i$ occurs in $\Phi$. For example, in particular, if $\Phi(\vec{x})$ has no free variables, then no substitution happens and $\Phi(\vec{t})$ is the same as $\Phi(\vec{x})$.

[11] It is also called $\mathcal{L}_{sc}^{C^2}$ *regression* sometimes below to avoid confusion.

**Definition 2** Consider an $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$ with respect to an $\mathcal{L}_{sc}^{C^2}$-restricted BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$. Then, the *modified regression operator* $\mathcal{R}$ is recursively defined as follows.

- If $W$ is not atomic, i.e., $W$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W'$, or $\mathbb{Q}v.W'$ where $\mathbb{Q}$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol, then

$$\begin{aligned}
\mathcal{R}[W_1 \vee W_2] &= \mathcal{R}[W_1] \vee \mathcal{R}[W_2], \\
\mathcal{R}[\neg W'] &= \neg \mathcal{R}[W'], \\
\mathcal{R}[W_1 \wedge W_2] &= \mathcal{R}[W_1] \wedge \mathcal{R}[W_2], \\
\mathcal{R}[Qv.W'] &= Qv.\mathcal{R}[W'].
\end{aligned}$$

- Otherwise, $W$ is an atom. There are several cases.

**a.** If $W$ is a regressable *Poss* atom, then it has the form $Poss(A(\vec{t}), \sigma)$ for terms of sort *action* and *situation* respectively in $\mathcal{L}_{sc}^{C^2}$. Then, there must be an action precondition axiom for $A$ of the form $Poss(A(\vec{z}), s) \equiv \Pi_A(\vec{z}, s)$, where the argument $\vec{z}$ of sort *object* can either be empty (i.e., $A$ is an action constant), a single variable $x$, or two-variable vector $\langle x, y \rangle$. Because of the syntactic restrictions of $\mathcal{L}_{sc}^{C^2}$ and according to the conditions in Definition 1, each term in $\vec{t}$ can only be a variable $x$, $y$ or some constant if any. Then,

$$\mathcal{R}[W] = \begin{cases}
\mathcal{R}\big[\big(\exists y.x = y \wedge \Pi_A(x, y, \sigma)\big)\big] & \text{if } \vec{t} = \langle x, x \rangle, \\
\mathcal{R}\big[\big(\exists x.y = x \wedge \Pi_A(x, y, \sigma)\big)\big] & \text{if } \vec{t} = \langle y, y \rangle, \\
\mathcal{R}\big[\widetilde{\Pi_A}(\vec{t}, \sigma)\big] & \text{if } \vec{t} \in \{y, \langle y, O \rangle, \langle O, x \rangle, \langle y, x \rangle\}, \\
\mathcal{R}\big[\Pi_A(\vec{t}, \sigma)\big] & \text{otherwise, i.e., if } \vec{t} \text{ is empty or} \\
 & \quad \vec{t} \in \big\{O, x, \langle x, y \rangle, \langle x, O \rangle, \langle O, y \rangle, \\
 & \quad \langle O, O_1 \rangle\big\},
\end{cases}$$

where $O$ and $O_1$ are constants and $\widetilde{\phi}$ denotes a *dual formula* for formula $\phi$ obtained by replacing every variable symbol $x$ (free or bound) with variable symbol $y$ and replacing every variable symbol $y$ (free or bound) with variable symbol $x$ in $\phi$, i.e., $\widetilde{\phi} = \phi[x/y, y/x]$. In this definition, in order to avoid introducing new variables but still ensure the correctness of regression in the sense that the regressed formula is logically equivalent to $W$ w.r.t. $\mathcal{D}$, we consider all the possible syntactic forms of the arguments $\vec{t}$ in action terms and treat them carefully in each of the four cases. Because of the restriction of the language of $\mathcal{L}_{sc}^{C^2}$ and the additional conditions in

Definition 1, we are able to reuse the variables $x$ and $y$ by switching their occurrences when $\vec{t}$ is either $y$, $\langle y, O\rangle$, $\langle O, x\rangle$ or $\langle y, x\rangle$.

**b.** If $W$ is a defined dynamic concept, it has the form $G(t, \sigma)$ for some object term $t$ and ground situation term $\sigma$, and there must be a TBox axiom for $G$ of the form $G(x, s) \equiv \phi_G(x, s)$. Because of the restrictions of the language $\mathcal{L}_{sc}^{\mathcal{C}^2}$, term $t$ can only be a variable $x$, $y$ or a constant. Then, we use the lazy unfolding technique as follows:

$$\mathcal{R}[W] = \begin{cases} \mathcal{R}\big[\phi_G(t, \sigma)\big] & \text{if } t \in \{O, x\}, \\ \mathcal{R}\big[\widetilde{\phi_G}(y, \sigma)\big] & \text{otherwise, i.e., if } t = y. \end{cases}$$

**c.** If $W$ is a primitive dynamic concept (or a dynamic role) of the form $F(t_1, do(\alpha, \sigma))$ (or $F(t_1, t_2, do(\alpha, \sigma))$, respectively) for some terms $t_1$ (and $t_2$) of sort *object*, ground term $\alpha$ of sort *action* and ground term $\sigma$ of sort *situation*, there must be an SSA for fluent $F$ of the form $F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a, s)$, whose detailed syntax is Eq. 5. Because of the restriction of the language $\mathcal{L}_{sc}^{\mathcal{C}^2}$, the terms $t_1$ and $t_2$ can only be a variable $x$, $y$ or some constant $O$. Then, when $W$ is a primitive dynamic concept, i.e., $W$ is of the form $F(t_1, do(\alpha, \sigma))$,

$$\mathcal{R}[W] = \begin{cases} \mathcal{R}\big[\Phi_F\big(t_1, \alpha, \sigma\big)\big] & \text{if } t_1 \in \{O, x\}, \\ \mathcal{R}\big[\widetilde{\Phi}_F\big(y, \alpha, \sigma\big)\big] & \text{otherwise, i.e., if } t_1 = y; \end{cases}$$

and, when $W$ is a dynamic role, i.e., $W$ is of the form $F(t_1, t_2, do(\alpha, \sigma))$,

$$\mathcal{R}[W] = \begin{cases} \mathcal{R}[(\exists y.y = x \wedge \Phi_F(x, y, \alpha, \sigma))] & \text{if } \langle t_1, t_2\rangle = \langle x, x\rangle; \\ \mathcal{R}[(\exists x.x = y \wedge \Phi_F(x, y, \alpha, \sigma))] & \text{if } \langle t_1, t_2\rangle = \langle y, y\rangle; \\ \mathcal{R}[\widetilde{\Phi_F}(y, t_2, \alpha, \sigma)] & \text{if } \langle t_1, t_2\rangle \in \{\langle y, x\rangle, \langle y, O\rangle, \langle O, x\rangle\}; \\ \mathcal{R}[\Phi_F(t_1, t_2, \alpha, \sigma)] & \text{otherwise, i.e, if } \langle t_1, t_2\rangle \in \\ & \big\{\langle x, y\rangle, \langle x, O\rangle, \langle O, y\rangle, \big\langle O, O_1\big\rangle\big\}. \end{cases}$$

**d.** If $W$ is of the form $A_1(\vec{t}) = A_2(\vec{t'})$ for some action function symbols $A_1$ and $A_2$, then by using axioms in $\mathcal{D}_{una}$,[12] we define the regression of $W$ as

$$\mathcal{R}[W] = \begin{cases} \mathit{false} & \text{if } A_1 \neq A_2, \\ \mathit{true} & \text{if } A_1 = A_2 \text{ and } A_1, A_2 \text{ are constant} \\ & \quad \text{action functions}, \\ \bigwedge_{i=1}^{|\vec{t}|} t_i = t'_i & \text{otherwise.} \end{cases}$$

Otherwise, if $W$ is any other situation independent atom (including equality between object terms) or $W$ is a concept or role uniform in $S_0$, then

$$\mathcal{R}[W] = W. \tag{10}$$

[12] Notice that the action functions with different number of arguments always use different function symbols (i.e., different names).

Our intention of case (**d.**) in Definition 2 is to get a $C^2$ formula (with any situation term suppressed) that has no (in)equality between action terms after regression. We therefore cannot leave (in)equalities between action terms untouched in the regressed formula unlike Reiter's definition of the regression operator that simply used Eq. 10 when dealing with (in)equalities between terms. By using unique name axioms for actions during regression, we can avoid functional terms in the resulting formula. Moreover, in case (**c.**) and case (**d.**) of Definition 2, when $\vec{t}$ is $\langle x, x\rangle$ (or $\langle y, y\rangle$, respectively), we define regression by using a quantified variable $y$ (or $x$, respectively); otherwise, we cannot ensure the correctness of regression in the sense that the regressed formula is logically equivalent to $W$ w.r.t. $\mathcal{D}$, i.e., $\mathcal{D} \models W \equiv \mathcal{R}[W]$. In particular, for any $\mathcal{L}_{sc}^{C^2}$ regressable formulas $W$ and $W'$ such that $\models W \equiv W'$, a correct definition of regression should result in $\mathcal{D} \models \mathcal{R}[W] \equiv \mathcal{R}[W']$. For example, in case (**c**), notice that

$$\models F(x, x, do(\alpha, \sigma)) \equiv \big(\exists y.x = y \wedge F(x, y, do(\alpha, \sigma)),$$

and it is easy to see that our definition of $\mathcal{R}$ ensures that

$$\models \mathcal{R}\big[F(x, x, do(\alpha, \sigma))\big] \equiv \mathcal{R}\big[\big(\exists y.x = y \wedge F\big(x, y, do(\alpha, \sigma)\big)\big)\big].$$

Consider the following counterexample if we perform regression by directly substituting $\langle x, y\rangle$ by $\langle x, x\rangle$ (or $\langle y, y\rangle$) on the RHS of SSAs or precondition axioms.

*Example 3* Consider a $\mathcal{L}_{sc}^{C^2}$-restricted BAT $\mathcal{D}$, which includes an SSA

$$F(x, y, do(a, s)) \equiv a = A(x) \wedge \big(\exists x.G(x, y, s)\big) \vee F(x, y, s). \tag{11}$$

Consider an $\mathcal{L}_{sc}^{C^2}$ regressable formula $W = F(x, x, do(A(C), S_0))$. Then if we perform regression by directly substituting $\langle x, y\rangle$ with $\langle x, x\rangle$ on the RHS of Eq. 11, we get $A(C) = A(x) \wedge (\exists x.G(x, x, S_0)) \vee F(x, x, S_0))$, which obviously will not be logically equivalent to $W$ w.r.t. $\mathcal{D}$. Indeed, in Eq. 11, the variable $y$ in $G$ is free, but once substituted by $x$ directly, it becomes bounded by $\exists x$ at the front of $G$, which should not happen.

Based on Definition 2, we are able to prove the following theorems.

**Theorem 2** *Suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable formula, then the regression $\mathcal{R}[W]$ defined above terminates in a finite number of steps.*

*Proof* This immediately follows from Definition 1, acyclicity of the TBox axioms, and from the assumption that RBox axioms are compiled into the SSAs and consequently do not participate in regression. Note also that each time, the application of $\mathcal{R}$ either goes from a formula to a sub-formula, or expands a *Poss* or a fluent atom using a corresponding precondition axiom or an SSA, but only finitely many expansions are possible because $W$ is a finite formula, mentioning only finitely many ground situation terms. □

Moreover, the following statement can be proved for $\mathcal{R}$ by induction over the structure of $W$. Since the detailed proof of Theorem 3 is rather long and mechanical, it is provided in Appendix B.2.

**Theorem 3** *Suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with the background BAT $\mathcal{D}$ in language $\mathcal{L}_{sc}^{C^2}$. Then, $\mathcal{R}[W]$ is an $\mathcal{L}_{sc}^{C^2}$ sentence uniform in $S_0$ and it is a $C^2$ sentence when the situation argument $S_0$ is suppressed. Moreover, $\mathcal{D} \models W \equiv \mathcal{R}[W]$.*

Consequently, we have the following theorem.

**Theorem 4** *Suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with the background BAT $\mathcal{D}$ in language $\mathcal{L}_{sc}^{C^2}$. Then, $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \models \mathcal{R}[W]$.*

*Proof* Part of our proof is almost word-by-word repetition of the laborious proof of the regression theorem given in [78]. Therefore, we will only briefly explain the idea of what has been done in [78] and provide details for what is different.

In [78], Pirri and Reiter first proved that "a BAT $\mathcal{D}$ is satisfiable iff $\mathcal{D}_{S_0} \cup \mathcal{D}_{una}$ is satisfiable". It is trivial that if a BAT $\mathcal{D}$ is satisfiable then $\mathcal{D}_{S_0} \cup \mathcal{D}_{una}$ is satisfiable. For the other direction, Pirri and Reiter proved it by constructing a model step by step for $\mathcal{D}$ that interprets axioms in $\mathcal{D} - (\mathcal{D}_{S_0} \cup \mathcal{D}_{una})$ properly, starting from any model of $\mathcal{D}_{S_0} \cup \mathcal{D}_{una}$. Similarly, we can also prove that "an $\mathcal{L}_{sc}^{C^2}$-restricted BAT $\mathcal{D}$ is satisfiable iff $\mathcal{D}_{S_0} \cup \mathcal{D}_{una}$ is satisfiable". We use the same idea with the following modification. For any model $\mathbb{M}$ of $\mathcal{D}_{S_0} \cup \mathcal{D}_{una}$, we also add interpretations for the defined concepts, such that the interpretations for $G(x)[s]$ are true iff those of $\phi_G(x)[s]$ are.

Subsequently, in [78], Pirri and Reiter proved the following lemma by using the above result: "Suppose $W$ is a regressable sentence with the background BAT $\mathcal{D}$ that is uniform in $S_0$, then $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models W$." This lemma is also valid for $\mathcal{L}_{sc}^{C^2}$. That is, suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with the background BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$, if $W$ is uniform in $S_0$, then $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models W$.

Now we prove the following statement: "Suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with the background BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$. If $W$ is uniform in $S_0$ and $W$ is a $C^2$ formula when $S_0$ is suppressed, then $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models W$ iff $\mathcal{D}_{S_0} \models W$." It is trivial to see that if $\mathcal{D}_{S_0} \models W$, then $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models W$. For the other direction, it is the same as proving if $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \cup \{\neg W\}$ is inconsistent, so is $\mathcal{D}_{S_0} \cup \{\neg W\}$. We can prove it by contradiction. That is, assume that $\mathcal{D}_{S_0} \cup \{\neg W\}$ is consistent, then there is a model $\mathbb{M}_0$, such that $\mathbb{M}_0 \models \mathcal{D}_{S_0} \cup \{\neg W\}$. We can then construct a model $\mathbb{M}$ such that it has the same domain on *object* sort as $\mathbb{M}_0$. For any action functions $A(\vec{x}_1)$, $B(\vec{x}_2)$, we construct $\mathbb{M}$ so that $(A(\vec{x}_1))^{\mathbb{M}} \neq (B(\vec{x}_2))^{\mathbb{M}}$ for any variable vectors $\vec{x}_1$ and $\vec{x}_2$ if symbol $A$ is different from $B$; and for any object terms $\vec{t}_1 = \langle t_{1,1} \cdots t_{1,n} \rangle$ and $\vec{t}_2 = \langle t_{1,1} \cdots t_{1,n} \rangle$ and any $n$-ary action function $A$, $(A(\vec{t}_1))^{\mathbb{M}} = (A(\vec{t}_2))^{\mathbb{M}}$ iff $(t_{1,i})^{\mathbb{M}} = (t_{2,i})^{\mathbb{M}}$ for all $i = 1..n$. Moreover, the rest interpretations of predicates or terms in $\mathbb{M}$ are the same as that of $\mathbb{M}_0$. Since $\mathcal{D}_{S_0} \cup \{\neg W\}$ has no action terms in it, $\mathbb{M}$ is well defined and is a model of $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \cup \{\neg W\}$, which is a contradiction.

Hence, all in all, we have: suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with a background BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$, if $W$ is uniform in $S_0$ and $W$ is a $C^2$ formula when $S_0$ is suppressed, then $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \models W$.

Then, by Theorem 3 and the above proved statement, we have $\mathcal{D} \models W$ iff $\mathcal{D} \models \mathcal{R}[W]$ iff $\mathcal{D}_{S_0} \models \mathcal{R}[W]$. □

We can also obtain the following theorem about decidability of the projection problem for $\mathcal{L}_{sc}^{C^2}$ regressable sentence $W$. (In particular, when $W$ is of the form $executable(S)$ for some ground situation $S$, it becomes the executability problem.)

**Theorem 5** *Suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with the background BAT $\mathcal{D}$ in language $\mathcal{L}_{sc}^{C^2}$. Then, determining whether $\mathcal{D} \models W$ is decidable.*

*Proof* According to Theorem 4, $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \models \mathcal{R}[W]$, where $\mathcal{R}[W]$ and the axioms in $\mathcal{D}_{S_0}$ are $C^2$ formulas. Therefore, determining whether $\mathcal{D} \models W$ is equivalent to determining whether $\mathcal{D}_{S_0} \wedge \neg \mathcal{R}[W]$ ($\mathcal{D}_{S_0}$ can be considered as a conjunction of all axioms in the initial theory) is unsatisfiable or not, which is a decidable problem, according to the fact that the satisfiability problem in $C^2$ is decidable. □

This theorem is important because it guarantees that the projection and executability problems in $\mathcal{L}_{sc}^{C^2}$ are decidable even if the initial KB $\mathcal{D}_{S_0}$ is incomplete. In Section 5.2, we give a detailed example that illustrates the basic reasoning tasks described above and reduction techniques for dealing with properties that need more than two variables, and show that using $\mathcal{L}_{sc}^{C^2}$, one can model realistic dynamic domains such as a student enrolment and grades management system. Below we continue our Example 1 from Section 4 about online shopping.

*Example 4* For any $S = do([a_1, \cdots, a_n], S_0)$, let $executable(S)$ be an abbreviation of the formula

$$Poss(a_1, S_0) \wedge \bigvee_{i=2}^{n} Poss(a_i, do([a_1, \cdots, a_{i-1}], S_0)).$$

We provide some examples of $\mathcal{L}_{sc}^{C^2}$ regressable formulas and the regression of some of these formulas.

$$executable(S_1),\ (\exists x) valuableCustomer(x, S_1), \qquad (\exists y) instore(y, S_2),$$
$$executable(S_2),\ boughtCD(Tom, SpiceGirls, S_1),$$

where the ground situation terms $S_1$ and $S_2$ are given as follows.

$$S_1 = do([buyCD(Tom, Classic), buyBook(Tom, Twilight), buyBook(Tom, Sail)], S_0)$$
$$S_2 = do([buyCD(Sam, SpiceGirls), returnCD(Sam, SpiceGirls)], S_0)$$

The regression of $(\exists y)instore(y, S_2)$ is as follows.

$$
\begin{aligned}
&\mathcal{R}\big[(\exists y)instore(y, S_2)\big]\\
&= (\exists y)\mathcal{R}\big[(\exists x)(x = Sam) \wedge y = SpiceGirls \vee\\
&\qquad instore(y, do(buyCD(Sam, SpiceGirls), S_0))\big]\\
&= (\exists y)(y = SpiceGirls \vee \mathcal{R}\big[instore(y, do(buyCD(Sam, SpiceGirls), S_0))\big])\\
&= (\exists y)(y = SpiceGirls \vee \mathcal{R}\big[instore(y, S_0) \wedge \neg((\exists x)(x = Sam) \wedge y = SpiceGirls)\big])\\
&= (\exists y)(y = SpiceGirls \vee instore(y, S_0) \wedge y \neq SpiceGirls)\\
&\equiv \top
\end{aligned}
$$

Here is one more example of regression.

$$
\begin{aligned}
&\mathcal{R}\big[(\exists x)valuableCustomer(x, S_1)\big]\\
&= (\exists x)\mathcal{R}\big[valuableCustomer(x, S_1)\big]\\
&= \cdots\\
&= (\exists x)\big(person(x) \wedge \exists^{\geq 3} y.\mathcal{R}\big[bought(x, y, S_1)\big]\big)\\
&= (\exists x)\big(person(x) \wedge \exists^{\geq 3} y.\mathcal{R}[x = Tom \wedge y = Sail \vee\\
&\qquad bought(x, y, do([buyCD(Tom, Classic), buyBook(Tom, Twilight)], S_0))]\big)\\
&= (\exists x)\big(person(x) \wedge \exists^{\geq 3} y.(x = Tom \wedge y = Sail \vee \mathcal{R}\big[bought(x, y, do([buyCD(Tom,\\
&\qquad Classic), buyBook(Tom, Twilight)], S_0))\big])\big)\\
&= \cdots\\
&= (\exists x)\big(person(x) \wedge \exists^{\geq 3} y.(x = Tom \wedge y = Sail \vee x = Tom \wedge y = Twilight\\
&\qquad \vee x = Tom \wedge y = Classic \vee bought(x, y, S_0))\big),
\end{aligned}
$$

which is true given that $\mathcal{D}_{S_0}$ is the conjunction of the following sentences:

$$
\begin{array}{lll}
creditCard(Visa). & creditCard(MasterCard). & person(Tom).\\
person(Sam). & cd(Classic). & cd(SpiceGirls).\\
book(Sail). & book(Twilight). & book(Java).\\
has(Tom, Visa) \vee has(Tom, MasterCard). & & \\
has(Sam, Visa) \vee has(Sam, MasterCard). & & \\
\forall x(instore(x, S_0) \vee x = Java). & &
\end{array}
$$

## 5.2 An example of a BAT and regression in $\mathcal{L}_{sc}^{C^2}$

Here, we give another detailed example of an $\mathcal{L}_{sc}^{C^2}$-restricted BAT $\mathcal{D}$ to illustrate the ideas described above in Sections 4 and 5.1. This section can be skipped if the reader does not need another example.

*Example 5* Consider an university that provides student administration and management services on the Web: admitting students, paying tuition fees, enrolling or dropping courses and entering grades.

Although the number of object arguments in the predicates can be at most two, sometimes, we are still able to handle those features of the systems which require more than two arguments. For example, the grade $z$ of a student $x$ in a course $y$ may be represented as a predicate $grade(x, y, z)$ in the general FOL (i.e., with three object arguments). Because the number of distinct grades is finite and they can be easily enumerated as "A", "B", "C" or "D", we can handle $grade(x, y, z)$ by replacing it with a finite number of extra predicates, say $gradeA(x, y)$, $gradeB(x, y)$, $gradeC(x, y)$ and $gradeD(x, y)$ such that they all have two variables only. However, the restriction on the number of variables limits the expressive power of the language if more than two arguments vary over infinite domains (such as energy, weight, time, etc). Despite this limitation, we conjecture that many web services still can be represented with at most two variables either by introducing extra predicates (just like we did for the predicate *grade*) or by grounding some of the arguments if their domains are finite and relatively small. Intuitively, it seems that most of the practical dynamical systems can be specified by using properties and actions with small arities, hence the techniques for arity reductions mentioned above and below require no more than polynomial increase in the number of axioms. The high-level features of our example are specified as the following concepts and roles.

- Primitive static concepts: $person(x)$ ($x$ is a person), $course(x)$ ($x$ is a course provided by the university).
- Primitive dynamic concepts: $incoming(x, s)$ ($x$ is an incoming student in the situation $s$, it is true when $x$ was admitted).
- Defined dynamic concepts: $eligFull(x, s)$ ($x$ is eligible to be a full-time student by paying more than 5000 dollars tuition fee), $eligPart(x, s)$ ($x$ is eligible to be a part-time student by paying no more than 5000 dollars tuition), $qualFull(x, s)$ ($x$ is a qualified full-time student if he or she pays full time tuition fee and takes at least 4 courses), $qualPart(x, s)$ ($x$ is a part-time student if he or she pays part-time tuition and takes 2 or 3 courses).
- Static role: $preReq(x, y)$ (course $x$ is a prerequisite for course $y$).
- Dynamic roles: $tuitPaid(x, y, s)$ ($x$ paid tuition fee $y$ in the situation $s$), $hasGrade(x, y, s)$ ($x$ has a grade for course $y$ in the situation $s$), $enrolled(x, y, s)$ ($x$ is enrolled in course $y$ in the situation $s$), $completed(x, y, s)$ ($x$ completes course $y$ in the situation $s$), $gradeA(x, y, s)$, $gradeB(x, y, s)$, $gradeC(x, y, s)$, $gradeD(x, y, s)$.

Web services are specified as actions: *reset* (at the beginning of each academic year, the system is being reset so that students need to pay tuition fee again to become eligible), $admit(x)$ (the university admits student $x$), $payTuit(x, y)$ ($x$ pays tuition fee with the amount of $y$), $enroll(x, y)$ ($x$ enrols in course $y$), $drop(x, y)$ ($x$ drops course $y$), $enterA(x, y)$ (enter grade "A" for student $x$ in course $y$), $enterB(x, y)$, $enterC(x, y)$, $enterD(x, y)$.

The BAT of the system is as follows (most of the axioms are self-explanatory).

**Precondition Axioms:**

$$\begin{aligned}
&Poss(reset, s) \equiv true,\\
&Poss(admit(x), s) \equiv person(x) \wedge \neg incoming(x, s),\\
&Poss(payTuit(x, y), s) \equiv incoming(x, s) \wedge \neg tuitPaid(x, y, s),\\
&Poss(drop(x, y), s) \equiv enrolled(x, y, s) \wedge \neg completed(x, y, s),\\
&Poss(enterA(x, y), s) \equiv enrolled(x, y, s) \wedge \neg completed(x, y, s),
\end{aligned}$$

and the precondition axiom for $enterB(x, y)$ ($enterC(x, y)$ and $enterD(x, y)$, respectively) is similar to the axiom for $enterA(x, y)$. Moreover, in the traditional SC, the precondition for action $enroll(x, y)$ would be equivalent to

$$(\forall z)(preReq(z, y) \wedge completed(x, z, s) \wedge \neg gradeD(x, z, s)).$$

However, in the modified SC, we only allow at most two object variables (including free or bound). Fortunately, the number of the courses offered in a university is limited (finite and relatively small) and relatively stable over years (if we manage the students in a college-wise range or department-wise range, the number of courses may be even smaller). Therefore, we can specify the precondition for the action $enroll(x, y)$ for each instance of $y$. That is, assume that the set of courses is $\{CS_1, \cdots, CS_n\}$, the precondition axiom for each $CS_i$ ($i = 1..n$) is

$$\begin{aligned}
Poss\big(enroll\big(x, CS_i\big), s\big) \equiv (\forall y)(&preReq\big(y, CS_i\big)\\
&\supset completed(x, y, s) \wedge \neg gradeD(x, y, s)).
\end{aligned}$$

On the other hand, when we do this transformation, we can omit the statements $course(x)$ for each course available at the university in the initial theory.

**Successor State Axioms:**

$$\begin{aligned}
&incoming(x, do(a, s)) \equiv a = admit(x) \vee incoming(x, s),\\
&tuitPaid(x, y, do(a, s)) \equiv a = payTuit(x, y) \vee tuitPaid(x, y, s) \wedge a \neq reset,\\
&enrolled(x, y, do(a, s)) \equiv a = enroll(x, y) \vee\\
&\qquad enrolled(x, y, s) \wedge \neg(a = drop(x, y) \vee a = enterA(x, y) \vee\\
&\qquad a = enterB(x, y) \vee a = enterC(x, y) \vee a = enterD(x, y)),\\
&completed(x, y, do(a, s)) \equiv a = enterA(x, y) \vee a = enterB(x, y)\\
&\qquad \vee a = enterC(x, y) \vee a = enterD(x, y) \vee\\
&\qquad completed(x, y, s) \wedge a \neq enroll(x, y),\\
&gradeA(x, y, do(a, s)) \equiv a = enterA(x, y) \vee gradeA(x, y, s) \wedge\\
&\qquad \neg(a = enterB(x, y) \vee a = enterC(x, y) \vee a = enterD(x, y)).
\end{aligned}$$

The SSAs for the fluents $gradeB(x, y, s)$, $gradeC(x, y, s)$ and $gradeD(x, y, s)$ are very similar to the one for fluent $gradeA(x, y, s)$. Therefore they are not repeated here. The SSA for the fluent $hasGrade(x, y, s)$ is also similar and for this reason it is omitted.

**Acyclic TBox Axioms:** (no static TBox axioms in this example)

$$\begin{aligned}
&eligFull(x, s) \equiv (\exists y)(tuitPaid(x, y, s) \wedge y > 5000),\\
&eligPart(x, s) \equiv (\exists y)(tuitPaid(x, y, s) \wedge y \leq 5000),\\
&qualFull(x, s) \equiv eligFull(x, s) \wedge (\exists^{\geq 4} y) enrolled(x, y, s),\\
&qualPart(x, s) \equiv eligPart(x, s) \wedge (\exists^{\geq 2} y) enrolled(x, y, s) \wedge (\exists^{\leq 3} y) enrolled(x, y, s).
\end{aligned}$$

An example of the initial theory $\mathcal{D}_{S_0}$ could be the conjunctions of the following sentences: $\forall x, y.\neg paidTuit(x, y, S_0), \forall x.incoming(x, S_0) \supset x = P_2 \vee x = P_3, \forall x.preReq(x, CS_4) \equiv x = CS1 \vee x = CS_3, \forall x.x \neq CS_4 \supset \neg(\exists y).prePeq(y, x), person(P_1), \cdots, person(P_m)$.

One can also imagine that some RBox axioms, for example,

$$gradeA(x, y, s) \supset hasGrade(x, y, s),$$

may be used for taxonomic reasoning in this domain.

Finally, we give an example of regression of an $\mathcal{L}_{sc}^{C^2}$ regressable formula:

$$\begin{aligned}
&\mathcal{R}\big[(\exists x).qualFull\big(x, do\big(\big[admit\big(P_1\big), payTuit\big(P_1, 6000\big)\big], S_0\big)\big)\big] \\
&\quad = \mathcal{R}\big[(\exists x).eligFull\big(x, do\big(\big[admit\big(P_1\big), payTuit\big(P_1, 6000\big)\big], S_0\big)\big) \\
&\qquad \wedge (\exists^{\geq 4} y)enrolled\big(x, y, do\big(\big[admit\big(P_1\big), payTuit\big(P_1, 6000\big)\big], S_0\big)\big)\big] \\
&\quad = \cdots \\
&\quad = (\exists x).(\exists^{\geq 4} y)enrolled\big(x, y, S_0\big) \wedge \big((\exists y)\mathcal{R}\big[y > 5000 \\
&\qquad \wedge tuitPaid\big(x, y, do\big(\big[admit\big(P_1\big), payTuit\big(P_1, 6000\big)\big], S_0\big)\big)\big]\big) \\
&\quad = \cdots \\
&\quad = (\exists x).(\exists^{\geq 4} y)enrolled\big(x, y, S_0\big) \wedge \big((\exists y).tuitPaid\big(x, y, S_0\big) \\
&\qquad \wedge y > 5000 \vee \big(x = P_1 \wedge y = 6000 \wedge y > 5000\big)\big),
\end{aligned}$$

which is false given the above initial theory.

Suppose we denote the above BAT as $\mathcal{D}$. Given goal $G$, for example $\exists x.qualFull(x)$, and a composite web service starting from the initial situation, for example, $do([admit(P_1), payTuit(P_1, 6000)], S_0)$ (we denote the corresponding resulting situation as $S_r$). We can check if the goal is satisfied after the execution of this composite web service by solving the projection problem whether $\mathcal{D} \models G[S_r]$. In our example, this corresponds to solving whether $\mathcal{D} \models \exists x.qualFull(x, S_r)$. We may also check if a given (ground) composite web service, i.e., a sequence of ground services $[A_1, A_2, \cdots, A_n]$, is possible to execute starting from the initial state by solving the executability problem whether $\mathcal{D} \models executable(do([A_1, A_2, \cdots, A_n], S_0))$. For example, we can check if the composite web service $[admit(P_1), payTuit(P_1, 6000)]$ is possible to be executed from the starting state by solving whether $\mathcal{D} \models executable(S_r)$. Note that both entailment problems can be decided (not only for the query that we consider, but also for any query) because they can be reduced to the satisfiability problem in $C^2$.

### 5.3 Some computational complexity results via regression

Since executability problems can be reduced to projection problems in linear time, below we will focus on the study of the projection problems only. In this section, we consider the computational time of solving the projection problem via regression in $\mathcal{L}_{sc}^{C^2}$. In Section 5.4, we propose a sub-language of $\mathcal{L}_{sc}^{C^2}$, which has better complexity bounds when solving the projection problem via regression. In Section 6, we study the computational complexity of solving the projection problem in general. Also, we discuss why it is still useful (sometimes) to solve the projection problems via regression, even it has a higher computational upper bound.

We first introduce a few new notations for later convenience. For any $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$, let function $size(W)$ be *the size of formula* $W$, which is defined recursively:

1. If $W$ is atomic (including equality), then $size(W) = 1$.
2. If $W$ is of the form $\neg W_1$, $\exists x.W_1$, $\forall x.W_1$, $\exists^{\geq n} x.W_1$, $\exists^{\leq n} x.W_1$, $\exists y.W_1$, $\forall y.W_1$, $\exists^{\geq n} y.W_1$, or $\exists^{\leq n} y.W_1$, then $size(W) = size(W_1) + 1$.
3. If $W$ is of the form $W_1 \wedge W_2$ or $W_1 \vee W_2$, then $size(W) = size(W_1) + size(W_2) + 1$.
4. If $W$ is of the form $W_1 \supset W_2$, then $size(W) = size(\neg W_1 \vee W_2)$.
5. If $W$ is of the form $W_1 \equiv W_2$, then $size(W) = size(W_1 \supset W_2) + size(W_2 \supset W_1) + 1$.[13]

For any situation term $\sigma = do([\alpha_1, \cdots, \alpha_k], S_0)$, let function $sitLength(\sigma) = k$ represent the number of action terms mentioned in $\sigma$. In particular, $sitLength(S_0) = 0$. For any formula $W$, define function

$$maxSit(W) = \max_{\sigma} \{sitLength(\sigma) \mid \sigma \text{ appears in } W\}.$$

Given any BAT $\mathcal{D}$, for any fluent $F$ whose SSA is of the form Eq. 5, define function $numFluent(F)$ as the number of fluents (including repeated ones) appearing in $\Phi_F$ (the RHS of the SSA of fluent $F$), and let

$$numFluentSSA(\mathcal{D}) = \max_{F} \{numFluent(F) \mid \text{any } F \text{ that has an SSA in } \mathcal{D}\}.$$

Besides, let

$$sizeSSA(\mathcal{D}) = \max_{F} \{size(\Phi_F) \mid \text{any SSA } F(\vec{x}, do(a, s)) \equiv \Phi_F \text{ in } \mathcal{D}\}.$$

Note that $numFluentSSA(\mathcal{D})$ and $sizeSSA(\mathcal{D})$ are different: the former one is the maximal number of fluents appearing in the formulas that are on the RHS of the SSAs of a given BAT $\mathcal{D}$, and the latter one is the maximal size of the formulas (including non-fluent atoms and logical connectives) that are on the RHS of the SSAs of a given BAT $\mathcal{D}$. Moreover, notice that once $\mathcal{D}$ is given, $numFluentSSA(\mathcal{D})$ and $sizeSSA(\mathcal{D})$ are fixed. In general, we have the following result.

**Theorem 6** *Consider any $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$ with a given BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$. Then, answering the query whether $\mathcal{D} \models W$ via regression is in the complement of* 2-NExpTime *in the size of $W$.*

*Proof* According to the discussion in the proof of Theorem 5, determining whether $\mathcal{D} \models W$ is equivalent to determining whether $\mathcal{D}_{S_0} \wedge \neg \mathcal{R}[W]$ is unsatisfiable or not, i.e., the complement problem of whether $\mathcal{D}_{S_0} \wedge \neg \mathcal{R}[W]$ is satisfiable or not.

Note that $\mathcal{D}_{S_0} \wedge \neg \mathcal{R}[W]$ is a $C^2$ formula (when the situation argument $S_0$ is suppressed). Moreover, according to [80], the satisfiability problem in language $C^2$ is in NExpTime, that is, NTIME($2^l$) if the input size of the formula is $l$. Since for any given $\mathcal{D}$, the size of $\mathcal{D}_{S_0}$ is fixed, the size of $\neg(\mathcal{D}_{S_0} \wedge \neg \mathcal{R}[W])$ is in $\Theta(n_1)$, where $n_1 = size(\mathcal{R}[W])$. Hence, answering the query whether $\mathcal{D} \models W$ via regression is in *co*-NTIME($2^{n_1}$), i.e., the complement of NExpTime w.r.t. the size of $\mathcal{R}[W]$.

[13] Note that according to this definition, $size(W_1 \equiv W_2) = size(W_1 \wedge W_2 \vee \neg W_1 \wedge \neg W_2)$.

However, in the worst case, computing $\mathcal{R}[W]$ takes ExpTime w.r.t. $n$, where $n = maxSit(W)$, and causes exponential blow-up in the size of formula $W$ w.r.t. $n$. In detail, without loss of generality, we assume that there is no defined concept in $W$. Otherwise, each defined concept will be replaced by its definitions from the TBox axioms within finite number of $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regression steps. This can cause no more than a constant increase to the size of the original formula, because TBox is fixed (once $\mathcal{D}$ is given), TBox is acyclic, there are only finitely many TBox axioms and the size of the formula on the RHS of each TBox axiom is limited by a constant.

Now, let $m = size(W)$, $k = numFluentSSA(\mathcal{D})$, $h = \max(2, sizeSSA(\mathcal{D}))$ and $n = maxSit(W)$. Both $k$ and $h$ are constants for the given BAT $\mathcal{D}$, and $h > k$. Moreover, since all action functions in $\mathcal{L}_{sc}^{\mathcal{C}^2}$ have no more than two arguments, the regression on equalities between action terms (see case **a.**) in Definition 2) results in a formula whose size is no more than three times of the size of the original atomic formulas (including at most one conjunction operator and at most two equality atoms), and such regression applies only once to each equality between action terms. The worst case scenario happens if each SSA mentions all $k$ fluents and the size of the RHS of the SSA is $h$. In this case, each step of regression on a fluent atom creates at most $h$ new branches on the next level of the regression tree ($k$ out of these $h$ branches have atomic fluents as their nodes). The next application of the regression operator replaces fluents in these $k$ nodes by the RHS of the corresponding SSAs, and so on. Since the longest situation term in $W$ before regression is of length $n$, the height of the resulting regression tree is no more than $n + 1$ (including $n$ levels of regressions on fluents and at most 1 level of regression on the equalities between actions), assuming that the regression tree starts at level 0 for root $W$. Finally, we are looking for a total number of leaves in this tree (this number is the size of the regressed formula $\mathcal{R}[W]$), which is

$$
\begin{aligned}
size(\mathcal{R}[W]) &\leq 3m[1 + (h-k)\sum_{i=0}^{n-1} k^i] \\
&= \begin{cases} 3m(h-1)n + 3m & \text{if } k = 1 \\ 3m\left[1 + (h-k)\dfrac{k^n - 1}{k-1}\right] & \text{if } k > 1. \end{cases} \\
&\leq \begin{cases} 3m(h-1)n + 3m & \text{if } k = 1 \\ 3m\big(hk^n\big) & \text{if } k > 1. \end{cases}
\end{aligned}
$$

Clearly, $3m(hk^n)$ is no more than $3h(m2^{n\log_2(k+1)})$. Formally, it is straightforward to prove by induction according to the recursive definition of the regression operator that $size(\mathcal{R}[W]) \in O(mn)$ when $k = 1$, and $size(\mathcal{R}[W]) \in O(mk^n)$ (which is the same as $size(\mathcal{R}[W]) \in O(m2^{n\log_2(k+1)})$) when $k \geq 2$. Overall, in the worst case, answering the query whether $\mathcal{D} \models W$ via regression is in the complement of $\text{NTIME}(2^{3hm2^{n\log_2(k+1)}})$ according to the above discussion, where $h$ and $k$ are constants. That is, answering the query whether $\mathcal{D} \models W$ via regression is in the complement of 2-NExpTime (non-deterministic doubly-exponential time). □

The SSA for a fluent $F$ is called *context-free* if the axiom has the syntactic form

$$F\big(\vec{x}, do(a, s)\big) \equiv \gamma_F^+\big(\vec{x}, a\big) \vee F\big(\vec{x}, s\big) \wedge \neg\gamma_F^-\big(\vec{x}, a\big),$$

that is, both the *positive condition* $\gamma_F^+(\vec{x}, a)$ and the *negative condition* $\gamma_F^-(\vec{x}, a)$ are situation independent (see Chapter 4 in [82]). According to this definition, it is easy to see that all the context conditions $(\psi_i^+(\vec{x}_{(i,+)})[s]$ for all $1 \leq i \leq m_+$ and $\psi_j^-(\vec{x}_{(j,-)})[s]$ for all $1 \leq i \leq m_-)$ in Eq. 5 are situation independent (i.e., there is no $s$ in any of the context conditions). Note that there is a special case if a positive (or negative, respectively) effect only depends on some action term (i.e., there is no context condition, or, the corresponding context condition is always equivalent to *true*). Then, we have the following theorem about the computational complexity for reasoning about projection problems.

**Theorem 7** *Given an $\mathcal{L}_{sc}^{C^2}$-restricted BAT $\mathcal{D}$, suppose that the SSA for a fluent $F$ is context-free. Then, the computational complexity of answering the queries of the form $F(\vec{X}, \sigma)$ via regression is in co-*NExpTime*, where $\vec{X}$ is a vector of object constants and $\sigma$ is a ground situation term.*

*Proof* The result follows from the analysis of the computational complexity of the projection problem in [82] (Chapter 4), which shows that the complexity is at most linear to the complexity of evaluating a sentence in the initial situation under such assumptions. In fact, from the proof of Theorem 6, when a fluent $F$ is context-free, $numFluent(F) = 1$. Let $size(\Phi_F) = h_0$, then the number of leaves in the regression tree of $F(\vec{X}, \sigma)$, i.e., the size of the regressed formula, equals $n(h_0 - 1) + 1$, which is in $O(n)$. Again, using the same reasoning as in the proof of Theorem 6, the problem of answering the queries of the form $F(\vec{X}, \sigma)$ via regression is the complement of the problem whether $\mathcal{D}_{S_0} \wedge \neg F(\vec{X}, \sigma)$ is satisfiable or not. Hence, its computational complexity is in *co*-NTIME$(2^{O(n)})$, i.e, it is in *co*-NExpTime. □

Using the same reasoning as in Theorem 7, in general, we have the following corollary.

**Corollary 1** *Consider an $\mathcal{L}_{sc}^{C^2}$-restricted BAT $\mathcal{D}$, and suppose that every SSA in $\mathcal{D}$ is context-free. Then, for any $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$, answering the query whether $\mathcal{D} \models W$ via regression is in co-*NExpTime *in the size of $W$.*

In Section 5.4, we further consider some special cases of BATs that have better computational complexity results for solving executability and projection problems (via regression), but less expressive power than $\mathcal{L}_{sc}^{C^2}$-restricted BATs.

### 5.4 A description logic based situation calculus

We see from Theorem 6 that the computational complexity of solving the projection problems in $\mathcal{L}_{sc}^{C^2}$ (using the regression) is quite high. On the other hand, with context-free SSAs, although we can gain better complexity (see Theorem 7), the expressive power of context-free SSAs is quite limited. Motivated by the observation that some DL languages have better computational complexity for concept satisfiability problems and/or ABox consistency problems than $C^2$ (see Section 3.2) and the idea of restricting the context conditions in the SSAs similar to context-free SSAs, we now consider another type of restriction on BATs in the language of $\mathcal{L}_{sc}^{C^2}$. We would like to get better complexity results than that of Theorem 6 when solving

the projection problem in general. At the same time, we consider a fragment that is more expressive than context-free SSAs. Moreover, we will see that this language has natural connections with DLs.

We first consider a sub-language of $C^2$, denoted $FO_{DL}$.

**Definition 3** The language of $FO_{DL}$ includes constants, monadic and dyadic predicates. Moreover, it is a union of two sub-languages: $FO_{DL} = FO^x_{DL} \cup FO^y_{DL}$, where the detailed definition of $FO^x_{DL}$ is provided below and $FO^y_{DL}$ is obtained by renaming every $x$ with $y$ and every $y$ with $x$ for every formula in $FO^x_{DL}$. The set $FO^x_{DL}$ is a minimal set of formulas built inductively as follows:

- *true* and *false* are in $FO^x_{DL}$.
- If $AC$ is a monadic predicate name, then $AC(x)$ is in $FO^x_{DL}$.
- If $b$ is a constant, then $x = b$ is in $FO^x_{DL}$.
- If $\phi$ is in $FO^x_{DL}$, then $\neg\phi$ is in $FO^x_{DL}$.
- If $\phi$ and $\psi$ are in $FO^x_{DL}$, then $\phi \wedge \psi$ and $\phi \vee \psi$ are in $FO^x_{DL}$.
- If $\phi(x)$ is in $FO^x_{DL}$, and $\phi(x)$ has at most one free variable $x$, and $R$ is a dyadic predicate name, $\widetilde{\phi}(y)$ is the dual formula of $\phi(x)$, obtained by renaming every $x$ (both free and bound) with $y$ and every $y$ (both free and bound) with $x$ in $\phi$, then $\exists y.R(x, y) \wedge \widetilde{\phi}(y)$ and $\forall y.R(x, y) \supset \widetilde{\phi}(y)$ are in $FO^x_{DL}$.
- If $\phi$ is in $FO^x_{DL}$, $\widetilde{\phi}$ is the dual formula of $\phi$, obtained by renaming every $x$ (both free and bound) with $y$ and every $y$ (both free and bound) with $x$ in $\phi$, then $[\exists y.]\widetilde{\phi}(y)$ and $[\forall y.]\widetilde{\phi}(y)$ are in $FO^x_{DL}$, where $[\exists y.]$ ($[\forall y.]$, respectively) means that if $\widetilde{\phi}$ has a free variable $y$, then it is quantified by $\exists y$ ($\forall y$, respectively); otherwise, there is no need to add the quantifier.

The semantics of $FO_{DL}$ are the same as the usual semantics of $\text{FO}^2$. Notice that for any $\phi \in FO^x_{DL}$ ($\phi \in FO^y_{DL}$, respectively), $\widetilde{\phi}$ is in $FO^y_{DL}$ ($FO^x_{DL}$, respectively). Moreover, it is easy to see that any sentence (i.e., closed formula) in $FO_{DL}$ is in both $FO^x_{DL}$ and $FO^y_{DL}$. We then are able to prove the following lemma (the detailed proof is provided in Appendix B.3).

**Lemma 1** *There are syntactic translations between $FO_{DL}$ and the DL language $\mathcal{ALCO}(U)$, i.e., they are equally expressive. Moreover, such translations lead to no more than a linear increase in the size of the translated formula.*

Recall from the review of DLs in Section 3.1 that the satisfiability problem of a concept and/or the consistency problem of an ABox in the DL language $\mathcal{ALCO}(U)$ can be solved in ExpTime. This is an improvement over $C^2$ and $\text{FO}^2$ (see Section 3.2). For this reason we would like to investigate a new SC based on $FO_{DL}$.

**Definition 4** We say that the SSA for a fluent $F$ is $\mathcal{ALCO}(U)$-*restricted* if the SSA of $F$ has the form of Eq. 5, where each context condition $\psi_i^+$ (or $\psi_i^-$, respectively) is a formula in $FO_{DL}$ when all situation variables are suppressed. Moreover, we say that the set of SSAs $\mathcal{D}_{ss}$ in a BAT $\mathcal{D}$ is $\mathcal{ALCO}(U)$-*restricted* if every axiom of a primitive dynamic concept in $\mathcal{D}_{ss}$ is $\mathcal{ALCO}(U)$-restricted and every axiom of a dynamic role in $\mathcal{D}_{ss}$ is both $\mathcal{ALCO}(U)$-restricted and context-free.

We say that a concept definition of the form Eq. 6 for any defined concept $G$ (including static or dynamic) is $\mathcal{ALCO}(U)$-*restricted* if the formula $\phi_G(x)$ on the RHS

of Eq. 6 is in $FO_{DL}$. Moreover, we say that the acyclic TBox $\mathcal{D}_T$ of a BAT $\mathcal{D}$ is $\mathcal{ALCO}(U)$-*restricted* if every axiom in the set is $\mathcal{ALCO}(U)$-*restricted*.

An initial theory $\mathcal{D}_{S_0}$ is $\mathcal{ALCO}(U)$-*restricted* if every sentence in it is in $FO_{DL}$ with the initial situation $S_0$ suppressed if there is any.

Overall, a BAT in $\mathcal{L}_{sc}^{C^2}$ is $\mathcal{ALCO}(U)$-*restricted* if $\mathcal{D}_{ss}$, $\mathcal{D}_T$ and $\mathcal{D}_{S_0}$ are $\mathcal{ALCO}(U)$-*restricted*.

We can then prove the following lemma (its proof is provided in Appendix B.4).

**Lemma 2** *Consider a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ that is $\mathcal{ALCO}(U)$-restricted. Let $W$ be an $\mathcal{L}_{sc}^{C^2}$ regressable formula that is uniform in a ground situation $S$ and has no appearance of Poss. Let $n = sitLength(S)$ and $m = size(W)$. Then, if $W$ with the situation term $S$ suppressed is in $FO_{DL}$, there is a $\Phi_W$ in $FO_{DL}$ such that $\mathcal{R}[W]$ is equivalent to $\Phi_W[S_0]$. It takes no more than $c \cdot n \cdot size(\Phi_W)$ steps of deduction from $\mathcal{R}[W]$ (with $S_0$ suppressed) to find such $\Phi_W$, where $c$ is a positive integer. Moreover, $size(\Phi_W)$ is in $O(2^{hmn+3h^2n^2})$ for some positive integer $h$. That is, the size of $\Phi_W$ is no more than exponential in the size of $W$.*

Note that it is not obvious at all whether formula resulting from regression of $W \in FO_{DL}$ still can be expressed in the same fragment $FO_{DL}$. Indeed, the regression operator does not preserve in general syntactic restrictions on a given formula. However, the fragment $FO_{DL}$ is designed specifically with an eye towards proving that the formula resulting from regression with respect to an $\mathcal{ALCO}(U)$-restricted BAT can be expressed in a description logic $\mathcal{ALCO}(U)$. The emphasis here on showing that we do not need a logic more expressive than $\mathcal{ALCO}(U)$. The proof of this Lemma 2 is laborious since it requires careful analysis of all possible syntactic forms of the input formula.

Then, we have the following complexity result.

**Theorem 8** *Consider a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_T$ are $\mathcal{ALCO}(U)$-restricted. Let $\mathcal{D}_{S_0}$, with the situation term $S_0$ suppressed, be in $FO_{DL}$. Let $W$ be any $\mathcal{L}_{sc}^{C^2}$ regressable sentence in $\mathcal{D}$ that is uniform in a ground situation $S$ and has no appearance of Poss. If $W$, with the situation term $S$ suppressed, is in $FO_{DL}$, then answering the query whether $\mathcal{D} \models W$ via regression can be solved in* 2-ExpTime.

*Proof* First, $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \models \mathcal{R}[W]$ by Theorem 4. Also, by Lemma 2, we can find a formula $\Phi$ in $FO_{DL}$ in no more than exponential time wrt the size of $W$, such that $\models \Phi[S_0] \equiv \mathcal{R}[W]$, and the size of $\Phi$ is no more than exponential in the size of $W$. Hence, $\mathcal{D}_{S_0} \models \mathcal{R}[W]$ iff $\mathcal{D}_{S_0} \models \Phi[S_0]$. It is the same as answering whether $\mathcal{D}_{S_0} \wedge \neg\Phi[S_0]$ is unsatisfiable or not, which is a complement problem of whether $\mathcal{D}_{S_0} \wedge \neg\Phi[S_0]$ is satisfiable or not. Let $\Psi$ be the formula $\mathcal{D}_{S_0} \wedge \neg\Phi[S_0]$ with situation term $S_0$ suppressed, and it is easy to see that $\Psi$ is in $FO_{DL}$, because $\mathcal{D}_{S_0}$ is in $FO_{DL}$ when the situation term $S_0$ is suppressed and $\Phi$ is in $FO_{DL}$. $\Psi$ has the same size as $\mathcal{D}_{S_0} \wedge \neg\Phi[S_0]$, and it is unsatisfiable iff $\mathcal{D}_{S_0} \wedge \neg\Phi[S_0]$ unsatisfiable. Using the syntactic translation function $\pi$ defined in the proof of Lemma 1 (see Appendix B.3), $\pi(\Psi)$ is a concept in DL language $\mathcal{ALCO}(U)$. To decide whether a concept $\pi(\Psi)$ is satisfiable in $\mathcal{ALCO}(U)$ is in ExpTime with respect to the size of $\pi(\Phi)$, which is linear in the size of $\Phi$. Hence, deciding whether $\mathcal{D}_{S_0} \models \mathcal{R}[W]$ is in *co*-ExpTime wrt the size of

$\mathcal{D}_{S_0} \wedge \neg\Phi[S_0]$. However, when $\mathcal{D}$ is given the size of $\mathcal{D}_{S_0}$ is fixed, hence, deciding whether $\mathcal{D}_{S_0} \models \mathcal{R}[W]$ is in fact in *co*-ExpTime wrt the size of $\Phi[S_0]$ (which is the same as the size of $\Phi$). Again, because the size of $\Phi$ is exponential in the size of $W$, deciding whether $\mathcal{D}_{S_0} \models \mathcal{R}[W]$ is in the complement of 2-ExpTime ( wrt the size of $W$), which is the same as 2-ExpTime. □

Recall from the review of DLs in Section 3.1 that the satisfiability problem of a concept and/or the consistency problem of an ABox in the DL language $\mathcal{ALCQO}(U)$ can also be solved in ExpTime. For this reason, similar to the development above, we can extend $FO_{DL}$ to a sub-language of $C^2$, say $FO_{DL+}$, by adding counting quantifiers to $FO_{DL}$.

**Definition 5** $FO_{DL+} = FO^x_{DL+} \cup FO^y_{DL+}$, where $FO^x_{DL+}$ is a minimal set of formulas built inductively below, and $FO^y_{DL+} = \{\widetilde{\phi} \mid \phi \in FO^x_{DL+}\}$.

- *true* and *false* are in $FO^x_{DL+}$.
- If $AC$ is a monadic predicate name, then $AC(x)$ is in $FO^x_{DL+}$.
- If $b$ is a constant, then $x = b$ is in $FO^x_{DL+}$.
- If $\phi$ is in $FO^x_{DL+}$, then $\neg\phi$ is in $FO^x_{DL+}$.
- If $\phi$ and $\psi$ are in $FO^x_{DL+}$, then $\phi \wedge \psi$ and $\phi \vee \psi$ are in $FO^x_{DL+}$.
- If $\phi(x)$ is in $FO^x_{DL+}$, and $\phi(x)$ has at most one free variable $x$, and $R$ is a dyadic predicate name, $\widetilde{\phi}(y)$ is the dual formula of $\phi(x)$, obtained by renaming every $x$ (both free and bound) with $y$ and every $y$ (both free and bound) with $x$ in $\phi$, then $\exists y.R(x, y) \wedge \widetilde{\phi}(y)$, $\forall y.R(x, y) \supset \widetilde{\phi}(y)$, $\exists^{\geq n} y.R(x, y) \wedge \widetilde{\phi}(y)$ and $\exists^{\leq n} y.R(x, y) \wedge \widetilde{\phi}(y)$ for any $n \in \mathbb{N}$ are in $FO^x_{DL+}$.
- If $\phi$ is in $FO^x_{DL+}$, $\widetilde{\phi}$ i dual formula of $\phi$, obtained by renaming every $x$ (both free and bound) with $y$ and every $y$ (both free and bound) with $x$ in $\phi$, then $[\exists y.]\widetilde{\phi}(y)$, $[\forall y.]\widetilde{\phi}(y)$, $[\exists^{\geq n} y.]\widetilde{\phi}(y)$ and $[\exists^{\leq n} y.]\widetilde{\phi}(y)$ for any $n \in \mathbb{N}$ are in $FO^x_{DL+}$, where $[\exists y.]$ ($[\forall y.]$, $[\exists^{\geq n} y.]$, or $[\exists^{\geq n} y.]$, respectively) means that if $\widetilde{\phi}$ has a free variable $y$, then it is quantified by $\exists y$ ($\forall y$, $\exists^{\geq n} y$, or $\exists^{\geq n} y$, respectively); otherwise, there is no need to add such quantifier.

The semantics of $FO_{DL+}$ is the same as the usual semantics of $C^2$. Similar to Lemma 1, we are able to prove the following lemma.[14]

**Lemma 3** *There are syntactic translations between $FO_{DL+}$ and the DL language $\mathcal{ALCQO}(U)$, i.e., they are equally expressive. Moreover, such translations lead to no more than a linear increase in the size of the translated formula.*

Similarly, we say that the SSA for a fluent $F$ is $\mathcal{ALCQO}(U)$-*restricted* if the SSA of $F$ has the form of Eq. 5, where each context condition $\psi_i^+$ (or $\psi_i^-$, respectively) is a formula in $FO_{DL+}$ when all situation variables are suppressed. Moreover, we say that the set of SSAs $\mathcal{D}_{ss}$ in a BAT is $\mathcal{ALCQO}(U)$-*restricted* if every axiom of a primitive dynamic concept in $\mathcal{D}_{ss}$ is $\mathcal{ALCQO}(U)$-restricted and every axiom of a dynamic role in

[14] The proof of Lemma 3 is exactly the same as the proof of Lemma 1 except that we only need to add translation for counting quantifiers, which is straightforward. Hence, the proof of Lemma 3 is omitted.

$\mathcal{D}_{ss}$ is both $\mathcal{ALCQO}(U)$-restricted and context-free. We say that a concept definition of the form Eq. 6 for any defined concept $G$ (including static or dynamic) is $\mathcal{ALCQO}(U)$-*restricted* if the formula $\phi_G(x)$ on the RHS of Eq. 6 is in $FO_{DL+}$. Moreover, we say that the acyclic TBox $\mathcal{D}_T$ of a BAT $\mathcal{D}$ is $\mathcal{ALCQO}(U)$-restricted.

Similar to Lemma 2, we can also prove a lemma for $\mathcal{ALCQO}(U)$-restricted regressable formulas as follows.

**Lemma 4** *Consider a BAT $\mathcal{D}$ in $\mathcal{L}^{C^2}_{sc}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_T$ are $\mathcal{ALCQO}(U)$-restricted. Let $W$ be an $\mathcal{L}^{C^2}_{sc}$ regressable formula that is uniform in a ground situation $S$ and has no appearance of Poss. Let $n = sitLength(S)$ and $m = size(W)$. Then, if $W$ with the situation term $S$ suppressed is in $FO_{DL+}$, there is a $\Phi_W$ in $FO_{DL+}$ such that $\mathcal{R}[W]$ is equivalent to $\Phi_W[S_0]$. It takes no more than $c \cdot n \cdot size(\Phi_W)$ steps of deduction from $\mathcal{R}[W]$ (with $S_0$ suppressed) to find such $\Phi_W$, where $c$ is a positive integer. Moreover, $size(\Phi_W)$ is in $O(2^{hmn+3h^2n^2})$ for some positive integer $h$. That is, the size of $\Phi_W$ is no more than exponential in the size of $W$.*

*Proof* The proof is exactly the same as the proof of Lemma 2 (see Appendix B.4), where $FO_{DL}$ ($FO^x_{DL}$, or $FO^y_{DL}$, respectively) is replaced by $FO_{DL+}$ ($FO^x_{DL+}$, or $FO^y_{DL+}$, respectively), and we need to consider sub-cases constructed using the counting quantifiers $\exists^{\geq n}$ and $\exists^{\leq n}$ in the proof. But the proof for cases that are constructed using $\exists^{\geq n}$ and $\exists^{\leq n}$ are the same as the proof for the case using the $\exists$-quantifier (see case (d) of the inductive step of the nested induction on the structure of the regressable formula $W$ in the proof of Lemma 2, Appendix B.4). Hence, the detailed proof of Lemma 4 is omitted here. □

As a consequence, similar to the proof of Theorem 8, we also have the following result.

**Theorem 9** *Consider a BAT $\mathcal{D}$ in $\mathcal{L}^{C^2}_{sc}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_T$ are $\mathcal{ALCQO}(U)$-restricted. Let $\mathcal{D}_{S_0}$, with the situation term $S_0$ suppressed, be in $FO_{DL+}$. Let $W$ be any $\mathcal{L}^{C^2}_{sc}$ regressable sentence in $\mathcal{D}$ that is uniform in a ground situation $S$ and has no appearance of Poss. If $W$, with the situation term $S$ suppressed, is in $FO_{DL+}$, then answering the query whether $\mathcal{D} \models W$ via regression can be solved in* 2-ExpTime.

## 6 Reasoning about actions without regression

In this section, we study the computational complexity of projection problems in general without using the regression mechanism. We are able to prove the following theorem.

**Theorem 10** *Consider any regressable sentence $W$ that is uniform in a ground situation $do([\alpha_1, \cdots, \alpha_n], S_0)$ in $\mathcal{L}^{C^2}_{sc}$ with a $\mathcal{L}^{C^2}_{sc}$-restricted BAT. Then, the projection problem of deciding whether $\mathcal{D} \models W$ can be polynomially (w.r.t. the size of a BAT) reduced to a problem deciding whether $\mathcal{D}' \models W'$ for a theory $\mathcal{D}'$ and a sentence $W'$ in $C^2$.*

*Proof* Consider an $\mathcal{L}_{sc}^{C^2}$ regressable sentence $W$ that is uniform in a ground $do([\alpha_1,\cdots,\alpha_n], S_0)$ with a background BAT in $\mathcal{L}_{sc}^{C^2}$. Without loss of generality, we can assume that there no predicate *Poss* appeared in $W$ (Otherwise, we can use one step $\mathcal{L}_{sc}^{C^2}$ regression operator to replace those atoms of the form $Poss(\alpha,\sigma)$ with the right-hand side of the precondition axioms, causing no more than linear increase in the size of the formula).

For every fluent $F(\vec{x},s)$ mentioned in $\mathcal{D}$ and every sub-situation $S_i = do([\alpha_1,\ldots,\alpha_i], S_0)$ of the ground situation $do([\alpha_1,\cdots,\alpha_n], S_0)$ that occurs in the query $W$, we introduce situation-free predicates $F^i(\vec{x})$ into $\mathcal{D}'$ for $i = 0..n$. Intuitively, for any $\vec{x}$ and for every sub-situation $S_i$, $F^i(\vec{x})$ is true iff $F(\vec{x}, S_i)$ is true, i.e., $F^i(\vec{x})$ is an $i$th copy of the fluent $F$ and for each fluent we introduce $n$ copies. For each $i = 0..n$, we also recursively define a translation function $g^i$ for any $\mathcal{L}_{sc}^{C^2}$ formula that has no quantifiers over any variables mentioned in situation terms.

(1) For each atom of the form $a = A(t_1,\ldots,t_k)$, where $A$ is a $k$-ary action function $A$, $g^n(a = A(t_1,\ldots,t_k))$ is not defined. For values $i < n$, the result of translation $g^i(a = A(\vec{t}))$ depends on whether the $(i+1)$st action mentioned in $do([\alpha_1,\cdots,\alpha_n], S_0)$ coincides or is distinct from the action $A$:

$$g^i\big(a = A(\vec{t})\big) \stackrel{def}{=} \begin{cases} false & \text{if } \alpha_{i+1} = B(\vec{t'}) \text{ for some } B \neq A \\ true & \text{if } \alpha_{i+1} = A \text{ and } k = 0 \\ \bigwedge_{j=1}^{k} t'_j = t_j & \text{if } \alpha_{i+1} = A(t'_1,\ldots,t'_k) \end{cases}$$

The intention of this definition is as follows. For each SSA, $F(\vec{x}, do(a,s)) \equiv \phi_F(\vec{x},s)$, an axiom $F^{i+1}(\vec{x}) \equiv g^i(\phi_F(\vec{x}))$ will be added using the translation function $g^i$ so that $g^i(\phi_F(\vec{x}))$ represents the equivalent condition of $F^{i+1}(\vec{x})$ being true, i.e., $F(\vec{x})$ is true after $\alpha_{i+1}$ is executed in the situation $S_i$, but $g^i(\phi_F(\vec{x}))$ should achieve this purpose without mentioning any action or situation terms. Hence, the definition of $g^i(a = A(t_1,\ldots t_k))$ is in fact the result of substituting variable $a$ with action $\alpha_{i+1}$ and then removing action terms using the unique name axioms for actions.

(2) For each situation-independent atom $P(\vec{t})$, $g^i(P(\vec{t})) \stackrel{def}{=} P(\vec{t})$.

(3) For each atomic fluent $F(\vec{t},\sigma)$, $g^i(F(\vec{t},\sigma)) \stackrel{def}{=} F^i(\vec{t})$, where $\sigma$ is a term of sort *Situation*.

(4) For each non-atomic formula, we have

$$g^i(\neg W) \stackrel{def}{=} \neg g^i(W),$$

$$g^i\big(W_1 \circ W_2\big) \stackrel{def}{=} g^i\big(W_1\big) \circ g^i\big(W_2\big) \text{ for any operator } \circ \in \{\wedge,\vee,\supset,\equiv\},$$

$$g^i\big(\mathbb{Q}v.W\big) \stackrel{def}{=} \mathbb{Q}v.g^i(W) \text{ for any quantifier } \mathbb{Q} \in \big\{\exists,\forall,\exists^{\geq m},\exists^{\leq m} \mid m \in \mathbb{N}\big\}.$$

Now, let $W' = g^n(W)$ and $\mathcal{D}' = \mathcal{D}^0_{S_0} \cup \mathcal{D}'_T \cup \mathcal{D}'_{ss}$, where

$$\mathcal{D}^0_{S_0} = \big\{g^0(\phi) \,\big|\, \phi \in \mathcal{D}_{S_0}\big\},$$

$\mathcal{D}'_T = \bigcup_{j=0}^{n} \mathcal{D}^j_T$ such that for each $0 \leq j \leq n$,

$$\mathcal{D}^j_T = \big\{g^j(\phi) \,\big|\, \phi \in \mathcal{D}_T\big\},$$

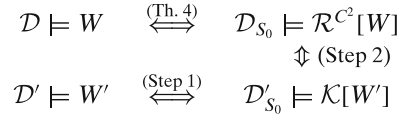


**Fig. 1** Diagram of the outline for proving Theorem 10

and $\mathcal{D}'_{ss} = \bigcup_{i=0}^{n-1} \mathcal{D}^i_{ss}$ such that for each $0 \leq i < n$,

$$\mathcal{D}^i_{ss} = \left\{ F^{i+1}(\vec{x}) \equiv g^i(\Phi_F) \mid \text{Axiom } F(\vec{x}, do(a,s)) \equiv \Phi_F \text{ is in } \mathcal{D}_{ss} \right\}.$$

It is easy to see that all axioms in $\mathcal{D}'$ are $C^2$ formulas and so is $W'$.[15]

Next, we prove that $\mathcal{D} \models W$ iff $\mathcal{D}' \models W'$. The general idea is presented in Fig. 1. Below, let $\mathcal{R}^{C^2}$ be a modified regression operator in $C^2$ defined in Section 5.1. First, we define a reasoning mechanism named $\mathcal{K}$ that is using axioms in $\mathcal{D}'_{ss} \cup \mathcal{D}'_T$ similar to $\mathcal{R}^{C^2}$. Each atom $P(\vec{t})$ that appears on the left-hand side of any axiom in $\mathcal{D}'_{ss} \cup \mathcal{D}'_T$ is to be replaced by the formula on the right-hand side with proper handling of variables. Once this has been done, we can prove that $\mathcal{D}' \models W'$ iff $\mathcal{D}'_{S_0} \models \mathcal{K}[W']$ (Step 1 in Fig. 1). Subsequently, we prove that $\mathcal{D}_{S_0} \models \mathcal{R}^{C^2}[W]$ iff $\mathcal{D}'_{S_0} \models \mathcal{K}[W']$ (Step 2 in Fig. 1).

Step 1. We first define the reasoning mechanism $\mathcal{K}$ recursively for a $C^2$ formula $W'$ as follows.

- If $W'$ is not atomic, i.e., $W'$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W_1$, or $\mathbb{Q}v.W_1$ where $\mathbb{Q}$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol, then

$$\begin{aligned} \mathcal{K}[W_1 \vee W_2] &= \mathcal{K}[W_1] \vee \mathcal{K}[W_2], \\ \mathcal{K}[\neg W_1] &= \neg \mathcal{K}[W_1], \\ \mathcal{K}[W_1 \wedge W_2] &= \mathcal{K}[W_1] \wedge \mathcal{K}[W_2], \\ \mathcal{K}[\mathbb{Q}v.W_1] &= \mathbb{Q}v.\mathcal{K}[W_1]. \end{aligned}$$

- Otherwise, $W'$ is an atom. There are several cases.

  **a.** If $W'$ has the form $G(t)$ for some predicate $G$ and some object term $t$, and there is an axiom $G(x) \equiv \phi_G(x)$ in $\mathcal{D}'_T$ for some $C^2$ formula $\phi_G(x)$. Because of the restrictions of the language $\mathcal{L}^{C^2}_{sc}$, term $t$ can only be a variable $x$, $y$ or a constant. Then, $\mathcal{K}$ is defined as follows:

$$\mathcal{K}[W] = \begin{cases} \mathcal{K}[\phi_G(t)] & \text{if } t \in \{O, x\}, \\ \mathcal{K}[\widetilde{\phi_G}(y)] & \text{otherwise, i.e., if } t = y. \end{cases}$$

  **b.** If $W'$ is of the form $F(t_1)$ (or $F(t_1, t_2)$, respectively) $(1 \leq i \leq n)$ for some predicate $F$ and some terms $t_1$ (and $t_2$) of sort *Object*, and

[15] The translation function $g^i$ defined above creates $n$ copies of each SSA and also $n$ copies of TBox even if not all of new axioms may be required to solve the projection problem. One can define a more economical translation function, but the function $g^i$ turns out to be sufficient for the purposes of proving the Theorem 10.

there is an axiom for $F$ of the form $F(\vec{x}) \equiv \Phi_F(\vec{x})$ in $\mathcal{D}'_{ss}$. Because of the restriction of the language $\mathcal{L}^{C^2}_{sc}$, the terms $t_1$ and $t_2$ can only be a variable $x$, $y$ or some constant $O$. Then, when $W'$ is of the form $F(t_1)$,

$$\mathcal{K}[W'] = \begin{cases} \mathcal{K}[\Phi_F(t_1)] & \text{if } t_1 \in \{O, x\}, \\ \mathcal{K}[\widetilde{\Phi}_F(y)] & \text{otherwise, i.e., if } t_1 = y; \end{cases}$$

and, when $W'$ is of the form $F(t_1, t_2)$,

$$\mathcal{K}[W'] = \begin{cases} \mathcal{K}[(\exists y.x = y \wedge \Phi_F(x, y))] & \text{if } \langle t_1, t_2\rangle = \langle x, x\rangle; \\ \mathcal{K}[(\exists x.y = x \wedge \Phi_F(x, y))] & \text{if } \langle t_1, t_2\rangle = \langle y, y\rangle; \\ \mathcal{K}[\widetilde{\Phi}_F(y, t_2)] & \text{if } \langle t_1, t_2\rangle \in \{\langle y, x\rangle, \langle y, O\rangle, \langle O, x\rangle\}; \\ \mathcal{K}[\Phi_{F^i}(t_1, t_2)] & \text{otherwise, i.e, if } \langle t_1, t_2\rangle \in \\ & \quad \{\langle x, y\rangle, \langle x, O\rangle, \langle O, y\rangle, \langle O, O_1\rangle\}. \end{cases}$$

**c.** If $W'$ is any other atom other than the atom considered in the above cases (including equality between object terms), then

$$\mathcal{K}[W'] = W'.$$

Similar to the proof of Theorem 4, it is easy to prove that $\mathcal{D}' \models W'$ iff $\mathcal{D}'_{S_0} \models \mathcal{K}[W']$ for $W' = g^n(W)$ and the theory $\mathcal{D}'$ defined as above using the translation function $g^i$.

Step 2. Let $\mathbb{F} = \{F(\vec{x}, S_0) \equiv F^0(\vec{x}) \mid \text{for every fluent in } \mathcal{D}\}$. Then, it is obvious to see that $\mathbb{F} \models \Phi_0 \equiv \Phi'_0$ for $\Phi_0$ ($\Phi'_0$, respectively) is the conjunction of all the axioms in $\mathcal{D}_{S_0}$ ($\mathcal{D}'_{S_0}$, respectively) according to how $\mathcal{D}'_{S_0}$ is defined. Moreover, it is easy to prove by structural induction that $\mathbb{F} \models \mathcal{R}^{C^2}[W] \equiv \mathcal{K}[W']$. Hence, $\mathcal{D}_{S_0} \models \mathcal{R}^{C^2}[W]$ iff $\mathcal{D}'_{S_0} \models \mathcal{K}[W']$. □

Then, we have the following corollary.

**Corollary 2** *The executability and projection problems are co-*NExpTime*-complete for $C^2$ regressable sentences with respect to a given $\mathcal{L}^{C^2}_{sc}$-restricted BAT.*

*Proof* For any $C^2$ sentence $\phi$, we can construct a special $\mathcal{L}^{C^2}_{sc}$-restricted BAT $\mathcal{D}$, where $\mathcal{D}$ consists of only one axiom *true* and a $C^2$ regressable sentence $\neg\phi$, then check whether $\phi$ is unsatisfiable iff $\mathcal{D} \models \neg\phi$. Consequently, the unsatisfiability problem in $C^2$ can be reduced to a projection problem in $\mathcal{L}^{C^2}_{sc}$ in linear time. Based on Theorem 10 and the complexity of solving satisfiability problems in $C^2$, it is easy to see that this statement holds. □

**Corollary 3** *The executability and projection problems for $\mathcal{ALCQO}(U)$-restricted ($\mathcal{ALCO}(U)$-restricted, respectively) regressable sentences and BATs are in co-*NExpTime.

*Proof* Based on Theorem 10 and the complexity of solving satisfiability problems in $C^2$, it is easy to see that solving the executability and projection problems for $\mathcal{ALCO}(U)$-restricted ($\mathcal{ALCQO}(U)$-restricted, respectively) regressable sentences and BATs have an upper-bound of *co*-NExpTime. This is because they are specializations of the problems for $\mathcal{L}_{sc}^{C^2}$-restricted BATs and $C^2$ regressable sentences. Recall that in Section 5.4, we showed that the upper-bound on complexity of solving the projection problem in an $\mathcal{ALCO}(U)$-restricted ($\mathcal{ALCQO}(U)$-restricted, respectively) SC is 2-ExpTime; this is achieved with regression (Theorems 8 and 9). However, the relationship between 2-ExpTime and co-NExpTime is still an open problem and for this reason it is not known which of these two bounds is tighter. □

It is tempting to apply the approach of constructing $\mathcal{D}'$ provided in the proof of Theorem 10 to an $\mathcal{ALCO}(U)$-restricted ($\mathcal{ALCQO}(U)$-restricted, respectively) BAT, and see if this approach could deliver better complexity results. (Recall that the satisfiability problem for $\mathcal{ALCO}(U)$ is ExpTime-complete.) However, it is easy to see that the corresponding $\mathcal{D}'$ of an $\mathcal{ALCO}(U)$-restricted ($\mathcal{ALCQO}(U)$-restricted, respectively) BAT can only be considered as a theory in $C^2$, but in a general case, it cannot be a theory in $FO_{DL}$ ($FO_{DL+}$, respectively). In particular, consider the SSAs for roles. For them, the constructed $\mathcal{D}'$ may include axioms of the syntactic form $R^i(x, y) \equiv R^{i-1}(x, y) \wedge \phi(x, y)$ for some binary predicates $R^i$ and $R^{i-1}$ and some $C^2$ context formula $\phi(x, y)$ (in the case of context-free SSAs for roles, as in our paper, $\phi(x, y)$ is *true*, therefore it can be omitted). Since, informally speaking, these axioms are not about concepts, axioms of this syntactic form cannot be equivalent to any formula in $FO_{DL}$ ($FO_{DL+}$). For this reason, the technique based on the translation function $g^i$ is not directly transferable to fragments of $\mathcal{L}_{sc}^{C^2}$. However, for a special case of an $\mathcal{ALCO}(U)$-restricted ($\mathcal{ALCQO}(U)$-restricted, respectively) BAT that has no SSAs for roles (i.e., roles are static and never change from one situation to another), there is no problem of dealing with axioms that involve roles which are not bundled into concepts. For this special case, one can easily repeat the same proofs as above for Theorem 10 and for Corollary 2 and show that both the projection and the executability problems for $\mathcal{ALCQO}(U)$-restricted ($\mathcal{ALCO}(U)$-restricted, respectively) regressable sentences and BATs are ExpTime-complete.

Our approach to proving Theorem 10 is inspired by the proofs in [2], an extended version of [3], where a different way of integrating description logics and actions is proposed. In [2, 3], all actions and their effects are ground, and the preconditions and effects of actions are specified for each action individually using DL ABox axioms in the sub-languages of $\mathcal{ALCQIO}$ (including $\mathcal{ALCQIO}$ itself). Thanks to these restrictions, the technique for proving complexity of the projection problem works for several DLs considered in [2, 3]. We will provide further comparison between our work and [2, 3] in Section 7.

Thus, we see that solutions of the executability and projection problems in $\mathcal{L}_{sc}^{C^2}$ can be reduced to the (un)satisfiability problems in $C^2$ directly (without using regression). Although the complexity upper-bound of solving executability and projection problems for $C^2$-restricted BATs and queries via regression is higher than

in Theorem 10, it is still necessary to study regression in $\mathcal{L}_{sc}^{C^2}$ for several reasons. First, regression is a natural well-known reasoning mechanism in the situation calculus, and it would be arrogant not to explore how regression works in $\mathcal{L}_{sc}^{C^2}$, in its fragments and the complexities of some essential reasoning problems about actions and change (such as the projection and executability problems) in $\mathcal{L}_{sc}^{C^2}$ and in its fragments via regression. In the previous sections, we show that the regression technique works well for reasoning about projection queries in $\mathcal{L}_{sc}^{C^2}$, when all situation terms are ground in the queries (see Section 5.1 and examples). Second, reasoning via regression for solving executability and projection problems is not always as bad as its complexity upper-bound and we envisage much better computational performance on realistic application domains in comparison to the worse-case scenario considered in the proof of complexity. It is important to implement carefully both regression in $\mathcal{L}_{sc}^{C^2}$ and the approach based on the translation function $g^i$ and conduct extensive testing and comparison of these implementations on several realistic domains (such as logistics, the blocks world, assembly and other benchmarks well-known in the planning community). Finally, and most importantly, the approach presented in the proof of Theorem 10 works only for regressable queries with ground situation terms, while the regression technique in Reiter's situation calculus works for more general queries where situation terms are not necessarily to be ground. We believe that it is possible to extend our regression operator for more general reasoning problems at least for some restricted BATs in $\mathcal{L}_{sc}^{C^2}$. For example, in the example of school enrolment, it is possible to use our modified regression to reason about some queries with non-ground situation terms, such as $\exists x.incoming(x, do(admit(x), S_0))$ and $\exists x.\forall y.incoming(x, do(drop(x, y), S_0))$, although currently our regression operator in $\mathcal{L}_{sc}^{C^2}$ is only defined for regressable formulas with ground situation terms. It is one of our future research topics to figure out for what kind of BATs in $\mathcal{L}_{sc}^{C^2}$, it is possible to reason about certain queries with non-ground situation terms (generalized queries in $\mathcal{L}_{sc}^{C^2}$) using regression in $\mathcal{L}_{sc}^{C^2}$ and still keep the reasoning problem to be decidable, and whether or not there are weaker restrictions on the format of the generalized queries.

## 7 Discussion and future work

The major consequence of the results proved above for the problem of service composition is the following. If both atomic services and properties of the world that can be affected by these services have no more than two parameters (other than the situation argument), then we are guaranteed that even in the state of incomplete information about the world, one can always determine whether a sequentially composed service is executable and whether this composite service will achieve a desired effect. The previously proposed approaches made different assumptions: [69] assumes that the complete information is available about the world when effects of a composite service are computed, and [23, 30] consider the propositional fragment of the situation calculus.

In [69, 73], it was proposed to use Golog for composition of Semantic Web services. Because our primitive actions correspond to elementary services, it is desirable to define Golog in our modified situation calculus too. It is surprisingly straightforward to define almost all Golog operators starting from our $C^2$-based

situation calculus. The only restriction in comparison with the original Golog [51, 82] is that we cannot define the operator $(\pi x)\delta(x)$, non-deterministic choice of an action argument, because $\mathcal{L}^{C^2}_{sc}$ regressable formulas cannot have occurrences of non-ground action terms in situation terms. In the original Golog this is allowed, because the regression operator is defined for a larger class of regressable formulas. However, everything else from the original Golog specifications remain in force, no modifications are required. In addition to providing a well-defined semantics for Web services, our approach also guarantees that the evaluation of tests in Golog programs is decidable (w.r.t. an arbitrary initial theory $\mathcal{D}_{S_0}$), which is missing in other approaches (unless one can make the closed world assumption or impose another restriction to regain decidability).

In [2, 3] ([2] is the extended version of [3]), an integration of the description logic $\mathcal{ALCQIO}$ (and its sub-languages) with an action formalism for reasoning about Web services is proposed. Their paper starts with a description logic $\mathcal{ALCQIO}$ and then defines services (actions) meta-theoretically: an atomic service is defined as the triple of sets of description logic formulas. The actions described in [3] are all ground, the preconditions and effects of the actions are described using ABox axioms with certain syntactic restrictions. To solve the executability and projection problems the paper reduces these problems to description logic reasoning.[16] The main aim is to show how the executability of sequences of actions and a solution to the projection problem can be computed, and how the complexity of solving these problems depends on the chosen description logic.

Despite that our work and [3] have common goals, our developments start differently and proceed in different directions. We start from the syntactically restricted first-order language $C^2$ (that is significantly more expressive than $\mathcal{ALCQIO}$), use it to construct the modified situation calculus (where actions are functional terms rather than ground terms only), and describe the preconditions and effects of actions using $C^2$-restricted BATs. In particular, our approach to representing the effects of actions inherits Reiter's solution to the frame problem. Moreover, it is possible to represent actions with global effects in our framework.[17] We study different approaches to solving the executability and projection problems (with and without regression), and show that the computational complexity for the executability and projection problems in $\mathcal{L}^{C^2}_{sc}$ is co-NExpTime-complete, which is same as that of in the framework of [3] based on $\mathcal{ALCQIO}$, yet $\mathcal{L}^{C^2}_{sc}$ is significantly more expressive both for representing actions and their effects. We also considered sub-languages of $\mathcal{L}^{C^2}_{sc}$, which are related to $\mathcal{ALCO}(U)$ and $\mathcal{ALCQO}(U)$ to gain better complexity bounds on reasoning about actions via regression. The restricted sub-languages (based on $\mathcal{ALCO}(U)$ or $\mathcal{ALCQO}(U)$) are incomparable with those studied in [3], since they have different expressive powers. For instance, it is not possible to quantify over

[16] Our translation function $g^i$ defined in the previous section is constructed by analogy with the reduction technique introduced in [2, 3].

[17] As an example of an action with global effects, consider an action *moveContainer*$(x)$. This action has an effect not only on the container represented by the argument $x$, but also has effects on locations of *all* objects inside this container. More formally speaking, for actions with global effects, a fluent can have universally quantified arguments that are not mentioned as arguments of actions that occur on the right hand side of the corresponding SSA.

arguments of an action or to describe actions that have global effects in [3], while there is restriction on the context-conditions in the SSAs of roles for $\mathcal{ALCO}(U)$-restricted and $\mathcal{ALCQO}(U)$-restricted sub-languages in our work.

Thanks to using ABox axioms to describe atomic services (ground actions), the advantage of [3] is that all reasoning is reduced to reasoning in description logics (and, consequently, can be efficiently implemented especially for less expressive fragments of $\mathcal{ALCQIO}$). Our advantages are: the convenience of representing actions as functional terms and the expressive power of $\mathcal{L}_{sc}^{C^2}$, including the compact representation of BATs. Besides, since $C^2$ and $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ are equally expressive, there are some (situation suppressed) formulas in our modified situation calculus that cannot be expressed in $\mathcal{ALCQIO}$ (that does not allow complex roles). In particular, [93] shows that $\mathcal{ALCQI}$ has the same complexity as $C^2$, but $\mathcal{ALCQI}$ is strictly less expressive than $C^2$: reflexive binary relations cannot be expressed in $\mathcal{ALCQI}$. Moreover, since our actions are represented as functional terms, in our future research, we can consider reasoning problems other than the executability and projection problems, and take advantage of the fact that actions are no longer ground.

The more recent papers of the same research group continue to explore the research direction initiated in [3]: [71] investigate complexity of planning in a description logic based action formalism, [56] attempts to solve the ramification problem when a TBox consists of general concept inclusion axioms (GCIs), and it is no longer an acyclic TBox as in [3]. It will also be one of our future work to consider a general TBox in the framework of $\mathcal{L}_{sc}^{C^2}$.

Propositional dynamic logic (**PDL**) was derived from dynamic logic and has several nice properties: **PDL** has the finite model property and is decidable [38]. Its satisfiability problem is ExpTime-complete [24, 79]. It turned out to be popular not only for reasoning about regular programs, but also as a logic of action [29, 81]. It is well known that dynamic logic extends modal logic by associating to every

action $a$, basic or complex, the modal operators $[a]$ and $\langle a \rangle$, thereby making it a multi-modal logic. But, in **PDL** quantification over actions is not allowed. More recently, [16, 19, 20] adapt **PDL** to reasoning about actions by quantifying over actions and allowing for equality between actions. They use regression and formulate the successor state axioms to solve the frame problem similar to [82]. However, in their framework, action terms can be constants or variables only (the domain closure axiom for actions or another similar assumption is required) and all fluents are propositional only. In our work, actions in BATs can be first-order terms, and the arity of each action function is no greater than two. Moreover, in our language, fluents can be dynamic concepts or dynamic roles, not just propositional statements. Also, as we mentioned above, it is possible to define complex Golog programs in our language.

In [99], the combined dynamic description language $\mathcal{PDLC}$ has been proposed as an attempt to reason about dynamics in description logics. From the perspective of modal logic, [99] combines polymodal **K** with **PDL** and proves the decidability of the resulting hybrid logic. $\mathcal{PDLC}$ is somewhat related to the products of modal logics (see [26–28] for the definition and survey of results). The issues related to combining modal logics in a more general context are reviewed in [50]. The proposed dynamic description logic is intended to define and classify concepts referring to actions and to describe dynamically changing domains by means of varying extensions of

concepts. A careful examination of the syntax of $\mathcal{PDLC}$ shows that actions can only be terms built from atomic actions (i.e., action variables) using standard dynamic logic constructors (composition, alternation, iteration) and from formulas using tests. Another restriction is that only concepts can change after executing an action: $[\alpha]C$ is a concept, where $\alpha$ is an action term and $C$ is a concept, but there is no similar constructor for roles. However, for any atomic formula $\phi$ which is either an ABox statement ($a : C$, or $aRb$, where $a, b$ are object names) or a boolean combination of ABox statements, and for any action term $\alpha$, $[\alpha]\phi$ is also a formula. The main contribution of [99] is the proof of the theorem that the satisfiability problem for $\mathcal{PDLC}$-formulas is decidable, but the complexity of the decision problem and the design of efficient decision algorithms are not explored. Our modified situation calculus can use action functions with arity no greater than two, and our dynamic roles can change after executing a sequence of actions too. However, we do not prove the decidability of the satisfiability problem for arbitrary formulas in our language. Moreover, we conjecture that this problem is undecidable in our language. From the positive side, we demonstrate that the executability and the projection problem are decidable for a wide class of queries and because these problems are the most essential in applications, the ability to solve these problems is sufficient for our present purposes. In [17, 18], the authors propose a logic that is similar to [3, 4, 99]. In the proposed logic, one not only can reason about complex actions similar to [99], but also can characterize actions by preconditions and conditional effects as in [3]. Also, a tableau algorithm for deciding satisfiability proposed in [17, 18] is based on an elaborated combination of previously known tableau algorithms.

In our paper [37], we investigated not only regression, but also progression as an alternative approach to solving the projection problem. We considered a modified progression that is weaker than the classical progression [55] for an incomplete knowledge base with *local-effect* SSAs, defined in [59]. We proved that the modified progression is sound wrt the classical progression, and we also provided an algorithm to compute our progression for the case when the initial theory is a CNF-based knowledge base (a set of disjunctions of equality-based formulas). Recently, [97] considers a notion of *strong progression*, a slight variant of the classical progression. In [97], it is shown that the strong progression is first-order definable for a BAT $\mathcal{D}$ with local-effect SSAs and the algorithm for computing progression was proposed for a special case of a BAT $\mathcal{D}$ with the so-called *strong local-effect* SSAs. Most recently, [58] presents a result stronger than that of [97] that for local-effect actions, progression is always first-order definable and computable. They give a very simple proof for this via the concept of forgetting. They also obtain results about first-order definability and computability for a class of knowledge bases and actions with non-local effects. Moreover, for a certain class of local-effect actions and knowledge bases for representing disjunctive information, they show that progression is not only first-order definable but also efficiently computable. The results in [58] can be adapted to our $\mathcal{L}_{sc}^{C^2}$ situation calculus straightforwardly, but nothing more than that. We currently cannot prove any stronger results by using $\mathcal{L}_{sc}^{C^2}$. Therefore, in this paper we do not include a separate section on progression just to repeat what they have recently accomplished. The further study of progression in $\mathcal{L}_{sc}^{C^2}$ is another future research direction.

There is research on updating in the description logic community that is somewhat related to computing progression in the situation calculus. Liu et al. [57] considers

update of an ABox in a DL with an acyclic TBox following [98] and also mentions that update can be applied to a boolean ABox formulated in $C^2$, but their update is defined in terms of a conjunction of primitive fluent literals, i.e., it is different from classical progression (the exact relations remain unexplored). de Giacomo et al. [31] uses a less expressive DL language *DL-Lite*, but defines update for the case when TBox consists of GCIs in comparison to acyclic TBox that is required in [57]. It is shown that the result of an update is always expressible by a *DL-Lite* ABox and a polynomial-time algorithm is provided that computes the update over a *DL-Lite* knowledge base. The more recent paper [15] from the same research group proposes to use Golog-like programs to efficiently reason about actions over ontologies based on a functional view of ontology with cyclic TBox in the case when the ontology is expressed in *DL-Lite*.

There are several other proposals to capture the dynamics of the world in the framework of description logics and/or its slight extensions. Similar to our paper [37], Drescher and Thielscher [21] explored reasoning about actions based on a description logic, but they concentrate on the *fluent calculus* [85] instead of the situation calculus. Instead of dealing with actions and the changes caused by actions, some of the approaches turned to extensions of description logics with temporal logics to capture the changes of the world over time [1, 5], and some others combined planning techniques with description logics to reason about tasks, plans and goals and exploit descriptions of actions, plans, and goals during plan generation, plan recognition, or plan evaluation [32]. Both [1] and [32] review several other related papers. Researchers also proposed to describe actions and the changes in terminological knowledge bases, closely related to description logics. For example, C. Kemke [49] describes action concepts by a set of parameters or object variables which refer to concepts in the object taxonomy, and precondition formulas as well as effect formulas describing how the world changes through actions (similar to STRIPS planning systems). In [9], all the actions of *e*-services are specified as constants, all the fluents of the system have only situation arguments, and BATs are translated under such assumptions into the description logic framework. It has a limited expressive power without using arguments of objects for actions and/or fluents: this may cause a blow-up of the knowledge base.

Since our paper concentrates on the situation calculus, we restrain here from reviewing an extensive literature on reasoning about actions, and numerous scenarios addressed in this literature, but the issue of decidability is important as well for action theories formulated in other frameworks.

In the future, we plan to extend this work along several directions. It would be interesting to see how our modified situation calculus can be used in real applications along the lines of SNAP, an e-commerce ontology developed at IBM for an automated system for recommending products and services in the domains of banking, insurance and telephony [72].

The most important direction for future research is an efficient implementation of practical scenarios of reasoning in $\mathcal{L}_{sc}^{C^2}$ and in its fragments: an efficient implementation of a decision procedure for solving the executability and projection problems. This procedure should handle the modified $\mathcal{L}_{sc}^{C^2}$ regression and perform efficient reasoning in $\mathcal{D}_{S_0}$. It should be straightforward to modify existing implementations of the regression operator for our purposes, but it is less obvious which reasoner will work most efficiently on practical problems. There are several different directions

that can be explored. First, according to [12] and Theorem 1, there exists an efficient algorithm for translating $C^2$ formulas to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ formulas. Also, if we consider fragments of $\mathcal{L}_{sc}^{C^2}$ introduced in Section 5.4 that guarantee a better complexity of solving the projection problem (see Theorem 8), more specifically, a BAT $\mathcal{D}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_T$ are $\mathcal{ALCO}(U)$-restricted, then a reasoning procedure working with $\mathcal{D}_{S_0}$ should be able to handle the description logic $\mathcal{ALCO}(U)$. Consequently, one can try to adapt tableaux-based decision procedures, such as those proposed in [87, 88], for (un)satisfiability checking in $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ and in $\mathcal{ALCO}(U)$. Since $\mathcal{ALCO}(U)$ is a fragment of OWL2, the recently developed extension of OWL, we can use directly existing reasoners that can solve the satisfiability problem in OWL2 [45]. The computational complexity of the satisfiability problem in $\mathcal{SROIQ}$, a DL underlying OWL2, is high, however, the currently available tableaux algorithms demonstrate excellent performance on realistic formulas. Second, one can try to avoid any translation from $C^2$ to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ and adapt resolution based automated theorem provers for the purposes of reasoning in $\mathcal{D}_{S_0}$ [46, 74]. Although in general, the worst-case computational complexity for the reasoning problems in $\mathcal{L}_{sc}^{C^2}$ or in its fragments is high, some practical scenarios may facilitate empirically efficient solutions to the projection and executability problems.

There are several other important research directions that can be taken. One of them is exploring meta-theoretical properties of our action theories. In the paper [41], several postulates are formulated, such that a well designed action theory should satisfy these postulates to avoid unexpected conclusions. For example, an action theory should not have implicit static laws (also known as domain constraints), and should not have implicit effect laws (that cannot be derived from static and effect axioms alone). The action theories conforming to these postulates are not only consistent, but also they are modular. Adapting these postulates to our action theories and providing algorithms that check if these postulates hold are topics for further investigation. Modular action theories are more elaboration tolerant in J.McCarthy's sense. There is also another issue related to elaboration tolerance. It has been argued in [22, 42, 96] that action theories might have to be revised and updated. We are going to investigate this issue in future too.

Finally, we would like to explore how our version of the situation calculus can accommodate events considered by John McCarthy in [67].

**Acknowledgements** Thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC) and to the Department of Computer Science of the University of Toronto for providing partial financial support. We are grateful to anonymous reviewers, Meghyn Bienvenu and Wael Yehia for useful comments on an earlier version of this paper.

## Appendix A: Semantics of description logics

In this appendix, we list the semantics of description logic syntax appearing in this paper. More details can be found in [5].

**Table 1** The semantics of some common description logic concept constructors

| Name | Syntax | Interpretation |
|---|---|---|
| Top | $\top$ | $\Delta^{\mathcal{I}}$ |
| Bottom | $\bot$ | $\emptyset$ |
| Nominal | $\{b\}$ | $b^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ |
| Negation | $\neg C$ | $\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$ |
| Intersection | $C_1 \sqcap C_2$ | $C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$ |
| Union | $C_1 \sqcup C_2$ | $C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$ |
| Qualified at-least restriction | $\geqslant nR.C$ | $\{\delta \in \Delta^{\mathcal{I}} \mid \lvert\{\delta_1 \in \Delta^{\mathcal{I}} \mid (\delta, \delta_1) \in R^{\mathcal{I}} \wedge \delta_1 \in C^{\mathcal{I}}\}\rvert \geq n\}$ |
| Qualified at-most restriction | $\leqslant nR.C$ | $\{\delta \in \Delta^{\mathcal{I}} \mid \lvert\{\delta_1 \in \Delta^{\mathcal{I}} \mid (\delta, \delta_1) \in R^{\mathcal{I}} \wedge \delta_1 \in C^{\mathcal{I}}\}\rvert \leq n\}$ |
| Existential quantification | $\exists R.C$ | $\{\delta \in \Delta^{\mathcal{I}} \mid \exists \delta_1.(\delta, \delta_1) \in R^{\mathcal{I}} \wedge \delta_1 \in C^{\mathcal{I}}\}$ |
| Value restriction | $\forall R.C$ | $\{\delta \in \Delta^{\mathcal{I}} \mid \forall \delta_1.(\delta, \delta_1) \in R^{\mathcal{I}} \supset \delta_1 \in C^{\mathcal{I}}\}$ |

**Table 2** The semantics of some common description logic role constructors

| Name | Syntax | Interpretation |
|---|---|---|
| Universal role | $U$ | $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ |
| Inverse | $R^{-}$ | $\{(\delta_1, \delta) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (\delta, \delta_1) \in R^{\mathcal{I}}\}$ |
| Complement | $\neg R$ | $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \backslash R^{\mathcal{I}}$ |
| Intersection | $R_1 \sqcap R_2$ | $R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}}$ |
| Union | $R_1 \sqcup R_2$ | $R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}}$ |
| Role restriction | $R\lvert_C$ | $R^{\mathcal{I}} \cap \Delta^{\mathcal{I}} \times C^{\mathcal{I}}$ |
| Identity | $id(C)$ | $\{(\delta, \delta) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \delta \in C^{\mathcal{I}}\}$ |
| Reflexive-transitive closure | $R^*$ | $\bigcup_{n \geq 0} (R^{\mathcal{I}})^n$ |
| Composition | $R_1 \circ R_2$ | $\{(\delta_1, \delta_2) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \exists \delta \in \Delta^{\mathcal{I}}. (\delta_1, \delta) \in R_1^{\mathcal{I}} \wedge (\delta, \delta_2) \in R_2^{\mathcal{I}}\}$ |

**Table 3** The semantics of terminological and assertional axioms

| Name | Syntax | Interpretation |
|---|---|---|
| Concept inclusion | $C_1 \sqsubseteq C_2$ | $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ |
| Role inclusion | $R_1 \sqsubseteq R_2$ | $R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$ |
| Concept equality | $C_1 \equiv C_2$ | $C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$ |
| Role equality | $R_1 \equiv R_2$ | $R_1^{\mathcal{I}} = R_2^{\mathcal{I}}$ |
| Concept assertion | $C(b)$ | $b^{\mathcal{I}} \in C^{\mathcal{I}}$ |
| Role assertion | $R(b_1, b_2)$ | $(b_1^{\mathcal{I}}, b_2^{\mathcal{I}}) \in R^{\mathcal{I}}$ |

## Appendix B: Proofs of lemmas and theorems

### B.1 $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ and $C^2$ are equally expressive

In this subsection, we will provide detailed proof for Theorem 1. First, we prove the following two lemmas.

**Lemma 5** *$C^2$ is as expressive as the language of $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$. In addition, the translation leads to no more than a linear increase in the size of the translated formula.*

*Proof* Similar to the proof in [12], we present the translation function from $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ to $C^2$ in several variants that behave as follows: $\tau^x\langle\rangle$ makes $x$ be the free variable of the monadic predicate, which is produced for its argument concept, while $\tau^y\langle\rangle$ makes the free variable be $y$. So, for an atomic concept $AC \in C_N$, $\tau^x\langle C\rangle = C(x)$, while $\tau^y\langle C\rangle = C(y)$. For an atomic role $R \in C_N$, $\tau^{x,y}\langle R\rangle$ produces a dyadic predicate $R(x, y)$, while $\tau^{y,x}\langle R\rangle$ produces a dyadic predicate $R(y, x)$. The translation functions $\tau^x\langle\rangle$, $\tau^y\langle\rangle$, and $\tau^{x,y}\langle\rangle$ are presented in the following two tables (Table 4 and Table 5). $\tau^{y,x}\langle\rangle$ is obtained from $\tau^{x,y}\langle\rangle$ by simultaneously exchanging all occurrences of $x$ and $y$ (whether free or bound).

The translation function $\tau\langle\rangle$ can now be defined simply as $\tau\langle C\rangle \stackrel{def}{=} \tau^x\langle C\rangle$ for any concept $C$, $\tau\langle R\rangle \stackrel{def}{=} \tau^{x,y}\langle R\rangle$ for any role $R$.

Then, the translation of terminological and assertional axioms can be defined as:

$$\tau\langle C(b)\rangle \stackrel{def}{=} \exists x.\tau^x\langle C\rangle \wedge x = b \text{ for any concept assertion } C(b);$$

$$\tau\langle R(b, b')\rangle \stackrel{def}{=} \exists x.\exists y.\tau^{x,y}\langle R\rangle \wedge x = b \wedge y = b' \text{ for any role assertion } R(b, b');$$

$$\tau\langle C_1 \sqsubseteq C_2\rangle \stackrel{def}{=} \forall x.\tau^x(C_1) \supset \tau^x(C_2) \text{ for any concept inclusion } C_1 \sqsubseteq C_2 \text{ if any};$$

$$\tau\langle C_1 \equiv C_2\rangle \stackrel{def}{=} \forall x.\tau^x(C_1) \equiv \tau^x(C_2) \text{ for any concept equality } C_1 \equiv C_2 \text{ if any};$$

$$\tau\langle R_1 \sqsubseteq R_2\rangle \stackrel{def}{=} \forall x.\forall y.\tau^{x,y}(R_1) \supset \tau^{x,y}(R_2) \text{ for any role inclusion } R_1 \sqsubseteq R_2 \text{ if any}.$$

**Table 4** A translation from $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ to $C^2$ for concept constructors

| Term $C$ | $\tau^x\langle C\rangle$ | $\tau^y\langle C\rangle$ |
|---|---|---|
| $AC, AC \in N_C$ | $AC(x)$ | $AC(y)$ |
| $\top$ | $x = x$ | $y = y$ |
| $\bot$ | $\neg(x = x)$ | $\neg(y = y)$ |
| $\{b\}$ | $x = b$ | $y = b$ |
| $\neg C$ | $\neg\tau^x\langle C\rangle$ | $\neg\tau^y\langle C\rangle$ |
| $C_1 \sqcap C_2$ | $\tau^x\langle C_1\rangle \wedge \tau^x\langle C_2\rangle$ | $\tau^y\langle C_1\rangle \wedge \tau^y\langle C_2\rangle$ |
| $C_1 \sqcup C_2$ | $\tau^x\langle C_1\rangle \vee \tau^x\langle C_2\rangle$ | $\tau^y\langle C_1\rangle \vee \tau^y\langle C_2\rangle$ |
| $\geqslant nR.C$ | $\exists^{\geq n} y.\, \tau^{x,y}\langle R\rangle \wedge \tau^y\langle C\rangle$ | $\exists^{\geq n} x.\, \tau^{y,x}\langle R\rangle \wedge \tau^x\langle C\rangle$ |
| $\leqslant nR.C$ | $\exists^{\leq n} y.\, \tau^{x,y}\langle R\rangle \wedge \tau^y\langle C\rangle$ | $\exists^{\leq n} x.\, \tau^{y,x}\langle R\rangle \wedge \tau^x\langle C\rangle$ |
| $\forall R.C$ | $\forall y.\, \tau^{x,y}\langle R\rangle \supset \tau^y\langle C\rangle$ | $\forall x.\, \tau^{y,x}\langle R\rangle \supset \tau^x\langle C\rangle$ |
| $\exists R.C$ | $\exists y.\, \tau^{x,y}\langle R\rangle \wedge \tau^y\langle C\rangle$ | $\exists x.\, \tau^{y,x}\langle R\rangle \wedge \tau^x\langle C\rangle$ |

**Table 5** A translation from $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ to $C^2$ for role constructors

| Term $R$ | $\tau^{x,y}\langle R\rangle$ |
|---|---|
| $R, R \in N_R$ | $R(x, y)$ |
| $U$ (universal role) | $x = x \wedge y = y$ |
| $id(C)$ | $x = y \wedge \tau^x\langle C\rangle$ |
| $\neg R$ | $\neg\tau^{x,y}\langle R\rangle$ |
| $R\|_C$ | $\tau^{x,y}\langle R\rangle \wedge \tau^y\langle C\rangle$ |
| $R^-$ | $\tau^{y,x}\langle R\rangle$ |
| $R_1 \sqcap R_2$ | $\tau^{x,y}\langle R_1\rangle \wedge \tau^{x,y}\langle R_2\rangle$ |
| $R_1 \sqcup R_2$ | $\tau^{x,y}\langle R_1\rangle \vee \tau^{x,y}\langle R_2\rangle$ |

For any DL interpretation $\mathcal{I}$, and the conventional FO interpretation $\mathcal{I}_1$ such that $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}}$ and $AC^{\mathcal{I}_1} = AC^{\mathcal{I}}$ ($R^{\mathcal{I}_1} = R^{\mathcal{I}}$, respectively ) for each atomic concept $AC$ (atomic role $R$, respectively), it is straightforward to prove by induction that $(\phi)^{\mathcal{I}} = (\tau(\phi))^{\mathcal{I}_1}$ for any formula $\phi$ in $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$.

In addition, it is obvious that the translation from $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ to $C^2$ can be done in linear time and causes no more than a linear increase in the size of the translated formula according to the translation function $\tau$ defined above. □

**Lemma 6** *The language of $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ is as expressive as $C^2$. In addition, the translation leads to no more than a linear increase in the size of the translated formula.*

*Proof* We proceed by structural induction on the syntax of formulas in $C^2$ with up to two free variables $x$ and $y$. Table 6 lists all possible kinds of formulas $\Gamma(x)$ that have a single free variable $x$, and shows how each kind is translated into a concept $C_\Gamma$. Let $N_C = \{AC \mid AC(x)$ or $AC(y)$ is a monadic predicate in language $C^2\}$, and $N_R = \{R \mid R(x, y)$ or $R(y, x)$ is a dyadic predicate in language $C^2\}$.

The translation of formulas with a single free variable $y$ is identical, except for the case when $\Gamma(y)$ is of the form $\exists x.\Psi(x, y)$, ($\exists^{\geq n}x.\Psi(x, y)$, and $\exists^{\leq n}x.\Psi(x, y)$, respectively), when we need to invert the relationship represented by $\Psi$. So, it is translated as $\exists(R_\Psi)^-.\top$ ($\geqslant n(R_\Psi)^-.\top$, and $\leqslant n(R_\Psi)^-.\top$, respectively).

**Table 6** A translation from $C^2$ to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ for formulas with a single free variable $x$

| $\Gamma(x)$ | $C_\Gamma$ | $\Gamma(x)$ | $C_\Gamma$ |
|---|---|---|---|
| $AC(x), AC \in N_C$ | $AC$ | $\Psi() \wedge \Phi(x)$ | $C_{\Psi()} \sqcap C_\Phi$ |
| $R(x, b), R \in N_R$ | $\exists R.\{b\}$ | $\Psi(x) \wedge \Phi(x)$ | $C_\Psi \sqcap C_\Phi$ |
| $R(b, x), R \in N_R$ | $\exists R^-.\{b\}$ | $\exists y.\Psi(x, y)$ | $\exists R_\Psi.\top$ |
| $R(x, x), R \in N_R$ | $\exists (R \sqcap id(\top)).\top$ | $\exists^{\geq n} y.\Psi(x, y)$ | $\geqslant n R_\Psi.\top$ |
| $x = b$ | $\{b\}$ | $\exists^{\leq n} y.\Psi(x, y)$ | $\leqslant n R_\Psi.\top$ |
| $x = x$ | $\top$ | $\exists y.\Psi(x)$ | $C_\Psi$ |
| $\neg\Psi(x)$ | $\neg C_\Psi$ | $\exists^{\geq n} y.\Psi(x)$ | $C_\Psi$ |
| | | $\exists^{\leq n} y.\Psi(x)$ | $C_\Psi$ |

**Table 7** A translation from $C^2$ to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ for formulas with two free variables

| $\Gamma(x, y)$ | $R_\Gamma$ |
|---|---|
| $R(x, y), R \in N_R$ | $R$ |
| $R(y, x), R \in N_R$ | $R^-$ |
| $x = y$ | $id(\top)$ |
| $\neg\Psi(x, y)$ | $\neg R_\Psi$ |
| $\Psi(x) \wedge \Phi(y)$ | $C_\Psi \times C_\Phi$ |
| $\Psi(x, y) \wedge \Phi()$ | $R_\Psi \sqcap R_{\Phi()}$ |
| $\Psi(x, y) \wedge \Phi(x)$ | $R_\Psi \sqcap (C_\Phi \times \top)$ |
| $\Psi(x, y) \wedge \Phi(y)$ | $R_\Psi \sqcap (\top \times C_\Phi)$ |
| $\Psi(x, y) \wedge \Phi(x, y)$ | $R_\Psi \sqcap R_\Phi$ |

Formulae of the form $\Gamma(x, y)$ with two free variables are translated to roles $R_\Gamma$ relating $x$ to $y$ according to Table 7. In Table 7, notice that the role constructor $C_1 \times C_2$ for any two concepts $C_1$ and $C_2$ is introduced in [12], whose semantics is defined as $C_1^{\mathcal{I}} \times C_2^{\mathcal{I}}$ given any interpretation $\mathcal{I}$. It is easy to see that $\times$ can be replaced using the standard role constructors in $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$, that is,

$$C_1 \times C_2 \stackrel{def}{=} \left((R \sqcup \neg R)\big|_{C_1}\right)^- \sqcap (R \sqcup \neg R)\big|_{C_2}$$

for any atomic role $R \in N_R$.

When a formula $\Gamma()$ without free variables occurs as a conjunct, then the number of free variables (1 or 2) in its context determines its translation: a concept or a role. For the case when a concept is desired, we need a translated concept $C_{\Gamma()}$ with the property that for any conventional $C^2$ interpretation $\mathcal{I}_1$ and a DL interpretation $\mathcal{I}$ such that $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}}$ and $AC^{\mathcal{I}_1} = AC^{\mathcal{I}}$ ($R^{\mathcal{I}_1} = R^{\mathcal{I}}$, respectively) for each atomic concept $AC$ (atomic role $R$, respectively), $\mathcal{I}_1 \models \Gamma() \equiv \mathit{true}$ iff $(C_\Gamma)^{\mathcal{I}} = \Delta^{\mathcal{I}}$, and $\mathcal{I}_1 \models \Gamma() \equiv \mathit{false}$ iff $(C_\Gamma)^{\mathcal{I}} = \emptyset$. Table 8 provides such translations. In contexts where we require roles, the translation is just $R_{\Gamma()} = C_{\Gamma()} \times C_{\Gamma()}$.

**Table 8** A translation from $C^2$ to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ for formulas without free variables

| $\Gamma()$ | $C_{\Gamma()}$ | $\Gamma()$ | $C_{\Gamma()}$ |
|---|---|---|---|
| $\mathit{true}$ | $\top$ | $\exists x.\Psi()$ | $C_{\Psi()}$ |
| $\mathit{false}$ | $\bot$ | $\exists y.\Psi()$ | $C_{\Psi()}$ |
| $C(b)$ | $\forall(\top\times\{b\}).C$ | $\exists^{\geq n} x.\Psi(x)$ | $\geqslant nU.C_\Psi$ |
| $R(b, b)$ | $\forall(\top\times\{b\}).(\exists R.\{b\})$ | $\exists^{\geq n} y.\Psi(y)$ | $\geqslant nU.C_\Psi$ |
| $R(b', b)$ | $\forall(\top\times\{b'\}).(\exists R.\{b\})$ | $\exists^{\geq n} x.\Psi()$ | $C_{\Psi()}$ |
| $b = b$ | $\top$ | $\exists^{\geq n} y.\Psi()$ | $C_{\Psi()}$ |
| $b' = b$ | $\bot$ | $\exists^{\leq n} x.\Psi(x)$ | $\leqslant nU.C_\Psi$ |
| $\neg\Psi()$ | $\neg C_{\Psi()}$ | $\exists^{\leq n} y.\Psi(y)$ | $\leqslant nU.C_\Psi$ |
| $\Psi() \wedge \Phi()$ | $R_{\Psi()} \sqcap R_{\Phi()}$ | $\exists^{\leq n} x.\Psi()$ | $C_{\Psi()}$ |
| $\exists x.\Psi(x)$ | $\exists U.C_\Psi$ | $\exists^{\leq n} y.\Psi()$ | $C_{\Psi()}$ |
| $\exists y.\Psi(y)$ | $\exists U.C_\Psi$ | | |

For any formula $\phi$ in $C^2$, the translation function *transl* can now be defined as: $transl(\phi) = C_\phi$ if $\phi$ has no free variables, or has only one free variable $x$ or $y$; and $transl(\phi) = R_\phi$ if $\phi$ has exactly two free variables.

We can prove by induction that for any conventional $C^2$ interpretation $\mathcal{I}_1$ and a DL interpretation $\mathcal{I}$ such that $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}}$ and $AC^{\mathcal{I}_1} = AC^{\mathcal{I}}$ ($R^{\mathcal{I}_1} = R^{\mathcal{I}}$, respectively) for each atomic concept $AC$ (atomic role $R$, respectively), we have $\mathcal{I}_1 \models \phi \equiv true$ iff $(transl(\phi))^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\mathcal{I}_1 \models \phi \equiv false$ iff $(transl(\phi))^{\mathcal{I}} = \emptyset$ for any (closed) sentence $\phi$.

It is obvious that the translation from $C^2$ to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ can be done in linear time and causes no more than a linear increase in the size of the translated formula according to the translation function $\tau$ defined above. □

**Theorem 1** (Section 3.2) *The description logic* $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ *and* $C^2$ *are equally expressive. In addition, translation in both directions leads to no more than a linear increase in the size of the translated formula.*

*Proof* It is a direct consequence of combining Lemmas 5 and 6. □

B.2 The correctness of the modified regression operator

In this subsection we provide a detailed proof for Theorem 3 in Section 5.1.

**Theorem 3** (Section 5.1) *Suppose $W$ is an $\mathcal{L}_{sc}^{C^2}$ regressable sentence with the background BAT $\mathcal{D}$ in language $\mathcal{L}_{sc}^{C^2}$. Then, $\mathcal{R}[W]$ is an $\mathcal{L}_{sc}^{C^2}$ sentence uniform in $S_0$ and it is a $C^2$ sentence when the situation argument $S_0$ is suppressed. Moreover, $\mathcal{D} \models W \equiv \mathcal{R}[W]$.*

*Proof* This theorem can be proved by induction on the number of regression steps.

Base case: It takes one step to terminate the regression.
If $W$ is of the form $A_1(\vec{t}) = A_2(\vec{t'})$ for some action function symbols $A_1$ and $A_2$, then there are three sub-cases:

(1) If $A_1 \neq A_2$, $\mathcal{R}[W] = false$ (by definition), which is uniform in $S_0$ and is a $C^2$ sentence. Note that $\mathcal{D} \models W \equiv false$ by the unique name axioms for actions in $\mathcal{D}$. Hence, $\mathcal{D} \models \mathcal{R}[W] \equiv W$.
(2) If $A_1 = A_2$ and $A_1$, $A_2$ are constant action functions, $\mathcal{R}[W] = true$ (by definition), which is uniform in $S_0$ and is a $C^2$ sentence. Note that $\mathcal{D} \models W \equiv true$ by the unique name axioms for actions in $\mathcal{D}$. Hence, $\mathcal{D} \models \mathcal{R}[W] \equiv W$.
(3) Otherwise, i.e., $A_1 = A_2$ and $A_1$, $A_2$ are not constant action functions, then $\mathcal{R}[W] = \bigwedge_{i=1}^{|\vec{t}|} t_i = t'_i$ (by definition), which is uniform in

$S_0$ and is a $C^2$ sentence. Note that $\mathcal{D} \models W \equiv \bigwedge_{i=1}^{|\vec{t}|} t_i = t_i'$ by the unique name axioms for actions in $\mathcal{D}$. Hence, $\mathcal{D} \models \mathcal{R}[W] \equiv W$.

Otherwise, $W$ is any other situation independent atom (including equality between object terms) or $W$ is a concept or role uniform in $S_0$, so $\mathcal{R}[W] = W$ (by definition), and it is obvious that $\mathcal{R}[W]$ is uniform in $S_0$ and is a $C^2$ formula when $S_0$ is suppressed. Moreover, $\mathcal{D} \models \mathcal{R}[W] \equiv W$.

Inductive step: Assume that our theorem is true for any regression that takes no more than $n$ steps ($n \geq 1$), now we prove it is true for any regression that takes $n+1$ steps. There are several cases as follows.

**a.** $W$ is of the form $Poss(A(\vec{t}), \sigma)$, for terms of sort *action* and *situation*, respectively, in $\mathcal{L}_{sc}^{C^2}$. Assume that the precondition axiom for action function $A(\vec{x})$ is of the form $Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x}, s)$, where $\vec{x}$ is either empty, $x$, or $\langle x, y\rangle$. There are four sub-cases:

(a.1) If $\vec{t} = \langle x, x\rangle$, then

$$\begin{aligned}
\mathcal{D} &\models \mathcal{R}[W] \\
&\equiv \mathcal{R}\big[\exists y.x = y \wedge Poss(A(x, y), \sigma)\big] \\
&= \exists y.x = y \wedge \mathcal{R}\big[\Pi_A(x, y, \sigma)\big] \quad \text{(by the definition of } \mathcal{R}) \\
&\equiv \exists y.x = y \wedge \Pi_A(x, y, \sigma) \quad \text{(by the induction hypothesis)} \\
&\equiv \exists y.x = y \wedge Poss(A(x, y), \sigma) \quad \big(\text{by } \mathcal{D}_{ap}\big) \\
&\equiv Poss(A(x, x), \sigma) \\
&= W
\end{aligned}$$

Moreover, by the induction hypothesis that $\mathcal{R}[\exists y.x = y \wedge \Pi_A(x, y, \sigma)]$ is uniform in $S_0$ and is a $C^2$ formula (when $S_0$ is suppressed), and so is $\mathcal{R}[W]$.

(a.2) Similarly to case (a.1) above, we can prove that the theorem is true if $\vec{t} = \langle y, y\rangle$.

(a.3) If $\vec{t} \in \{y, \langle y, O\rangle, \langle O, x\rangle, \langle y, x\rangle\}$, we need to ensure the result of substituting $\vec{t}$ into the precondition axiom is still logically equivalent to the original one. It can be proved case by case. We will just show one case as an example, and the rest of the cases can be proved similarly. For example, when $\vec{t}$ is $\langle y, O\rangle$ $\vec{x}$ can only be $\langle x, y\rangle$ in the precondition axiom. It is obvious that $Poss(A(y, x), s) \equiv \widetilde{\Pi_A}(y, x, s)$ is logically equivalent to $Poss(A(x, y), s) \equiv \Pi_A(x, y, s)$ by renaming all $x$ with $y$ and all $y$ with $x$ (free or bound). Hence, we are able

to substitute $\vec{t}$ into the precondition of $Poss(A(y, x), s)$ without introducing new variables. Then,

$$\begin{aligned}
\mathcal{D} &\models \mathcal{R}[W] \\
&= \mathcal{R}\big[\widetilde{\Pi}_A(y, O, \sigma)\big] \qquad \text{(by the definition of } \mathcal{R}) \\
&\equiv \widetilde{\Pi}_A(y, O, \sigma) \qquad \text{(by the induction hypothesis)} \\
&\equiv Poss(A(y, O), \sigma) \qquad \text{(by the renamed precondition axiom)} \\
&= W
\end{aligned}$$

Moreover, by the induction hypothesis that $\mathcal{R}[\widetilde{\Pi}_A(y, O, \sigma)]$ is uniform in $S_0$ and is a $C^2$ formula (when $S_0$ is suppressed), and so is $\mathcal{R}[W]$.

(a.4) Otherwise, i.e., if $\vec{t}$ either is empty or $\vec{t} \in \{O, x, \langle x, y\rangle, \langle x, O\rangle, \langle O, y\rangle, \langle O, O_1\rangle\}$, it is obvious that we can substitute $\vec{t}$ directly into the precondition axiom without causing any problem. That is,

$$\begin{aligned}
\mathcal{D} &\models \mathcal{R}[W] \\
&= \mathcal{R}\big[\Pi_A(\vec{t}, \sigma)\big] \qquad \text{(by the definition of } \mathcal{R}) \\
&\equiv \Pi_A(\vec{t}, \sigma) \qquad \text{(by the induction hypothesis)} \\
&\equiv Poss\big(A(\vec{t}), \sigma\big) \quad \big(\text{by } \mathcal{D}_{ap}\big) \\
&= W
\end{aligned}$$

Again, by using the induction hypothesis, $\mathcal{R}[\Pi_A(\vec{t}, \sigma)]$ is uniform in $S_0$ and is a $C^2$ formula (when $S_0$ is suppressed), and so is $\mathcal{R}[W]$.

**b.** $W$ is a defined dynamic concept of the form $G(t, \sigma)$ for some object term $t$ and ground situation term $\sigma$, and there must be a TBox axiom for $G$ of the form $G(x, s) \equiv \phi_G(x, s)$. Because of the restrictions of the language $\mathcal{L}_{sc}^{C^2}$, term $t$ can only be a variable $x$, $y$ or a constant. There are two sub-cases.

(b.1) When $t \in \{O, x\}$, it is obvious see that

$$\begin{aligned}
\mathcal{D} &\models \mathcal{R}[W] \\
&= \mathcal{R}[\phi_G(t, \sigma)] \quad \text{(by the definition of } \mathcal{R}) \\
&\equiv \phi_G(t, \sigma) \quad \text{(by the induction hypothesis)} \\
&\equiv G(t, \sigma) \quad \text{(by the TBox axiom)} \\
&= W
\end{aligned}$$

Again, by using the induction hypothesis, $\mathcal{R}[\phi_G(t, \sigma)]$ is uniform in $S_0$ and is a $C^2$ formula (when $S_0$ is suppressed), and so is $\mathcal{R}[W]$.

(b.2) When $t$ is variable $y$, then we can rename all $x$ ($y$, respectively) in the TBox axiom with $y$ ($x$, respectively), and still get an equivalent TBox axiom: $G(y,s) \equiv \widetilde{\phi_G}(y,s)$. Then,

$$\begin{aligned}\mathcal{D} &\models \mathcal{R}[W]\\ &= \mathcal{R}[\widetilde{\phi_G}(y,\sigma)] \quad \text{(by the definition of } \mathcal{R})\\ &\equiv \widetilde{\phi_G}(y,\sigma) \quad \text{(by the induction hypothesis)}\\ &\equiv G(y,\sigma) \quad \text{(by the renamed TBox axiom)}\\ &= W\end{aligned}$$

Again, by using the induction hypothesis, $\mathcal{R}[\widetilde{\phi_G}(y,\sigma)]$ is uniform in $S_0$ and is a $C^2$ formula (when $S_0$ is suppressed), and so is $\mathcal{R}[W]$.

**c.** $W$ is a primitive dynamic concept (a dynamic role) of the form $F(t_1, do(\alpha,\sigma))$ (or $F(t_1,t_2,do(\alpha,\sigma))$, respectively) for some terms $t_1$ (and $t_2$) of sort *object*, ground term $\alpha$ of sort *action* and ground term $\sigma$ of sort *situation*. There must be an SSA for fluent $F$ of the form $F(\vec{x}, do(a,s)) \equiv \Phi_F(\vec{x},a,s)$, whose detailed syntax is Eq. 5. Because of the restriction of the language $\mathcal{L}^{C^2}_{sc}$, the terms $t_1$ and $t_2$ can only be a variable $x$, $y$ or some constant $O$. In fact, the discussion of sub-cases for a primitive dynamic concept $F(t_1, do(\alpha,\sigma))$ is very similar to the proof for defined concepts except that instead of using a TBox axiom, we will use the SSA of $F$. The discussion of sub-cases for a dynamic role $F(t_1,t_2,do(\alpha,\sigma))$ is very similar to the proof for an atom of the form $Poss(A(t_1,t_2),\sigma)$ except that instead of using precondition axioms, we use the SSA of $F$. Since it is straightforward, details are omitted here.

**d.** $W$ is not atomic, i.e., $W$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W'$, or $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol. This is the last case we need to consider for the inductive step. Therefore, it is obvious that there are four sub-cases depending on the different forms of $W$. Because the discussions for all sub-cases are very similar except that they use different logical constructors, we will provide details for one of the sub-cases, and omit the rest. As an example, we consider the sub-case that $W$ is of the form $W_1 \vee W_2$. Then,

$$\begin{aligned}\mathcal{D} \models \mathcal{R}[W] &= \mathcal{R}\big[W_1\big] \vee \mathcal{R}\big[W_2\big] \text{ (by the definition of } \mathcal{R})\\ &\equiv W_1 \vee W_2 \qquad \text{(by the induction hypothesis)}\\ &= W\end{aligned}$$

Again, by using the induction hypothesis, $\mathcal{R}[W_1]$ and $\mathcal{R}[W_2]$ are uniform in $S_0$ and are both $C^2$ formulas (when $S_0$ is suppressed),

hence $\mathcal{R}[W]$ is still uniform in $S_0$ and is a $C^2$ formula (when $S_0$ is suppressed).

Overall, we proved that for any $\mathcal{L}_{sc}^{C^2}$ regressable sentence $W$ with the background BAT $\mathcal{D}$ in language $\mathcal{L}_{sc}^{C^2}$, $\mathcal{R}[W]$ is an $\mathcal{L}_{sc}^{C^2}$ sentence uniform in $S_0$ and it is a $C^2$ sentence when the situation argument $S_0$ is suppressed. Moreover, $\mathcal{D} \models W \equiv \mathcal{R}[W]$. □

### B.3 $\mathcal{ALCO}(U)$ and $FO_{DL}$ are equally expressive

In this section, we prove Lemma 1 presented in Section 5.4. Notice that in the proof of this Lemma, we provide purely syntactic translation functions between $\mathcal{ALCO}(U)$ and $FO_{DL}$.

**Lemma 1** (Section 5.4) *There are syntactic translations between $FO_{DL}$ and the DL language $\mathcal{ALCO}(U)$, i.e., they are equally expressive. Moreover, such translations lead to no more than a linear increase in the size of the translated formula.*

*Proof* The proof is similar to the proof of Theorem 1. We first prove that there is a syntactic translation function from $\mathcal{ALCO}(U)$ to $FO_{DL}$.

A syntactic translation $\tau$ from $\mathcal{ALCO}(U)$ to $FO_{DL}$ for any concept $C$ is defined as: $\tau(C) \stackrel{def}{=} \tau^x(C)$ for any concept $C$. $\tau^x()$ makes $x$ be the free variable of the monadic predicate, which is produced for its argument concept (see Table 9). During translation we also need a variant of $\tau$, $\tau^y()$, to make $y$ be the free variable of the monadic predicate (see Table 9). Then, the translation of terminological and assertional axioms can be defined as:

$$\tau(C(b)) \stackrel{def}{=} \exists x.\tau^x(C) \wedge x = b \text{ for any concept assertion } C(b);$$

$$\tau\big(R(b, b')\big) \stackrel{def}{=} \exists x.x = b \wedge \exists y.\tau^{x,y}(R) \wedge y = b';$$

**Table 9** A syntactic translation from $\mathcal{ALCO}(U)$ to $FO_{DL}$

| Term $C$ | $\tau^x(C)$ | $\tau^y(C)$ |
|---|---|---|
| $AC, AC \in N_C$ | $AC(x)$ | $AC(y)$ |
| $\top$ | *True* | *True* |
| $\bot$ | *False* | *False* |
| $\{b\}$ | $x{=}b$ | $y{=}b$ |
| $\neg C_1$ | $\neg\tau^x(C_1)$ | $\neg\tau^y(C_1)$ |
| $C_1 \sqcap C_2$ | $\tau^x(C_1) \wedge \tau^x(C_2)$ | $\tau^y(C_1) \wedge \tau^y(C_2)$ |
| $C_1 \sqcup C_2$ | $\tau^x(C_1) \vee \tau^x(C_2)$ | $\tau^y(C_1) \vee \tau^y(C_2)$ |
| $\exists R.C_1, R \in N_R$ | $\exists y.\ R(x, y) \wedge \tau^y(C_1)$ | $\exists x.\ R(x, y) \wedge \tau^x(C_1)$ |
| $\forall R.C_1, R \in N_R$ | $\forall y.\ R(x, y) \supset \tau^y(C_1)$ | $\forall x.\ R(x, y) \supset \tau^x(C_1)$ |
| $\exists U.C_1$ | $\exists y.\tau^y(C_1)$ | $\exists x.\tau^x(C_1)$ |
| $\forall U.C_1$ | $\forall y.\tau^y(C_1)$ | $\forall x.\tau^x(C_1)$ |

**Table 10** A syntactic translation from $FO_{DL}$ to $\mathcal{ALCO}(U)$

| $\Phi$ | $\pi(\Phi)$ |
|---|---|
| $AC(x)$, $AC(x)$ is atomic | $AC$ |
| $AC(y)$, $AC(y)$ is atomic | $AC$ |
| $True$ | $\top$ |
| $False$ | $\bot$ |
| $x=b$, $b$ is a constant | $\{b\}$ |
| $y=b$, $b$ is a constant | $\{b\}$ |
| $\neg\Psi$, | $\neg\pi(\Psi)$ |
| $\Psi_1 \vee \Psi_2$ | $\pi(\Psi_1) \sqcup \pi(\Psi_2)$ |
| $\Psi_1 \wedge \Psi_2$ | $\pi(\Psi_1) \sqcap \pi(\Psi_2)$ |
| $\exists y.\, R(x, y) \wedge \Psi(y)$, $R \in N_R$ | $\exists R.\pi(\Psi(y))$ |
| $\exists x.\, R(y, x) \wedge \Psi(x)$, $R \in N_R$ | $\exists R.\pi(\Psi(x))$ |
| $\forall y.\, R(x, y) \supset \Psi(y)$, $R \in N_R$ | $\forall R.\pi(\Psi(y))$ |
| $\forall x.\, R(y, x) \supset \Psi(x)$, $R \in N_R$ | $\forall R.\pi(\Psi(x))$ |
| $\exists y.\Psi(y)$, $\Psi(y)$ has only one free variable $y$ | $\exists U.\pi(\Psi(y))$ |
| $\exists x.\Psi(x)$, $\Psi(x)$ has only one free variable $x$ | $\exists U.\pi(\Psi(x))$ |
| $\forall y.\Psi(y)$, $\Psi(y)$ has only one free variable $y$ | $\forall U.\pi(\Psi(y))$ |
| $\forall x.\Psi(x)$, $\Psi(x)$ has only one free variable $x$ | $\forall U.\pi(\Psi(x))$ |

for any role assertion $R(b, b')$;

$$\tau(C_1 \sqsubseteq C_2) \stackrel{def}{=} \forall x.\neg\tau^x(C_1) \vee \tau^x(C_2) \text{ for any concept inclusion axiom } C_1 \sqsubseteq C_2 \text{if there is any;}$$

$$\tau(C_1 \equiv C_2) \stackrel{def}{=} (\forall x.\neg\tau^x(C_1) \vee \tau^x(C_2)) \wedge (\forall x.\neg\tau^x(C_2) \vee \tau^x(C_1)) \text{ for any concept equality axiom } C_1 \equiv C_2 \text{ if there is any.}$$

In addition, according to the definition of $\tau$ in Table 9 and the fact that there are no nested appearances of $\sqsubseteq$ and $\equiv$ in DL KBs, it is obvious that the translation from $\mathcal{ALCO}(U)$ to $FO_{DL}$ can be done in linear time and causes no more than a linear increase in the size of the translated formula.

Now, we prove that there is a syntactic translation function from $FO_{DL}$ to $\mathcal{ALCO}(U)$.

A syntactic translation $\pi$ from $FO_{DL}$ to $\mathcal{ALCO}(U)$ for any formula $\Phi \in FO_{DL}$ is defined in Table 10. In addition, it is obvious that the translation from $FO_{DL}$ to $\mathcal{ALCO}(U)$ can be done in linear time and causes no more than a linear increase in the size of the translated formula according to the translation function $\pi$ defined above. □

## B.4 Restricting syntax of BATs to gain computational advantages

In this section, we will prove Lemma 1 in Section 5.4. But first, we define an operator $\epsilon$ on any $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$, such that it will replace all atomic formula of the

form $A_1(\vec{t}\,) = A_2(\vec{t'})$ for some action terms $A_1(\vec{t}\,)$ and $A_2(\vec{t'})$ using the unique name axioms for actions in $\mathcal{D}_{una}$ for any given BAT $\mathcal{D}$.

**Definition 6** For any given BAT $\mathcal{D}$ and an $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$ in it, we define $\epsilon$ recursively as follows:

– If $W$ is of the form $A_1(\vec{t}\,) = A_2(\vec{t'})$ for some action terms $A_1(\vec{t}\,)$ and $A_2(\vec{t'})$ (i.e., *equality between action terms*), then

$$\epsilon[W] = \begin{cases} false & \text{if } A_1 \neq A_2, \\ true & \text{if } A_1 = A_2 \text{ and } A_1, A_2 \text{ are constant action functions,} \\ \bigwedge_{i=1}^{|\vec{t}|} t_i = t'_i & \text{otherwise.} \end{cases}$$

Otherwise, if $W$ is any other situation independent atom, then

$$\epsilon[W] = W.$$

– Otherwise, if $W$ is not atomic, i.e., $W$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W'$, or $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol, then

$$\epsilon[W_1 \vee W_2] = \epsilon[W_1] \vee \epsilon[W_2], \;\; \epsilon[\neg W'] = \neg \epsilon[W'],$$
$$\epsilon[W_1 \wedge W_2] = \epsilon[W_1] \wedge \epsilon[W_2], \;\; \epsilon[Qv.W'] = Qv.\epsilon[W'].$$

Note that $\epsilon$ can be considered as performing one step of $\mathcal{L}_{sc}^{C^2}$ regression on equalities between action terms in the given $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$, and it is easy to see that if $W$ is uniform in situation $S$, then $\epsilon[W]$ is still uniform in situation $S$. Moreover, we can prove the following property for $\epsilon$.

*Property 1* For any given BAT $\mathcal{D}$ and an $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$ in $\mathcal{D}$, we have that $\epsilon[W]$ is still $\mathcal{L}_{sc}^{C^2}$ regressable and $\mathcal{D} \models W \equiv \epsilon[W]$.

*Proof* It is easy to prove by induction on the structure of $W$.

Base case: If $W$ is atomic, there are two sub-cases.

(1) $W$ is of the form $A_1(\vec{t}\,) = A_2(\vec{t'})$ for some action terms $A_1(\vec{t}\,)$ and $A_2(\vec{t'})$. If $A_1 \neq A_2$, we have $\mathcal{D} \models W \equiv false$ by axioms in $\mathcal{D}_{una}$, which is $\mathcal{D} \models W \equiv \epsilon[W]$, since $\epsilon[W] = false$ by the definition of $\epsilon$; else, if $A_1 = A_2$ and $A_1, A_2$ are constant action functions, then by axioms in $\mathcal{D}_{una}$, $\mathcal{D} \models W \equiv true$, therefore $\mathcal{D} \models W \equiv \epsilon[W]$ according to the definition of $\epsilon$; otherwise, $\mathcal{D} \models W \equiv \bigwedge_{i=1}^{|\vec{t}|} t_i = t'_i$ by axioms in $\mathcal{D}_{una}$, which is $\mathcal{D} \models W \equiv \epsilon[W]$, since $\epsilon[W] = \bigwedge_{i=1}^{|\vec{t}|} t_i = t'_i$ by the definition of $\epsilon$. Moreover, it is obvious that $\epsilon[W]$ is still $\mathcal{L}_{sc}^{C^2}$ regressable.

(2) Otherwise, $W$ is atomic and not of the above form. By the definition of $\epsilon$, we have $\epsilon[W]=W$, hence $\mathcal{D} \models W \equiv \epsilon[W]$. Moreover, it is obvious that $\epsilon[W]$ is still $\mathcal{L}_{sc}^{C^2}$ regressable.

Inductive step: $W$ is not atomic and $W$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W'$, or $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol. Then for each sub-case, it is easy to prove that $\mathcal{D} \models W \equiv \epsilon[W]$ by the induction hypothesis. For instance, if $W$ is of the form $W_1 \vee W_2$, then

$$\begin{aligned} \mathcal{D} \models W &= W_1 \vee W_2 \\ &\equiv \epsilon\big[W_1\big] \vee \epsilon\big[W_2\big] \text{ (by the induction hypothesis)} \\ &= \epsilon\big[W_1 \vee W_2\big] \quad \text{(by the definition of } \epsilon) \\ &= \epsilon[W]. \end{aligned}$$

Moreover, it is obvious that $\epsilon[W]$ is still $\mathcal{L}_{sc}^{C^2}$ regressable by the induction hypothesis that $\epsilon[W_1]$ and $\epsilon[W_2]$ are both $\mathcal{L}_{sc}^{C^2}$ regressable. It is easy to see that for other sub-cases, such as $W_1 \wedge W_2$, $\neg W'$, and $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers), the proof is very similar to the sub-case of $W_1 \vee W_2$ and therefore details are omitted here.
Overall, $\mathcal{D} \models W \equiv \epsilon[W]$ for any $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$ in $\mathcal{D}$ and $\epsilon[W]$ is still $\mathcal{L}_{sc}^{C^2}$ regressable. □

We prove the following lemma that will be useful when proving Lemma 1 in Section 5.4. Notice that the lemma says that the regression of $W$ is *the same as* (not just equivalent to) the regression of $\epsilon[W]$.

**Lemma 7** *Consider any given BAT $\mathcal{D}$, the $\mathcal{L}_{sc}^{C^2}$ regression operator $\mathcal{R}$ defined in Section 5.1, and any $\mathcal{L}_{sc}^{C^2}$ regressable formula $W$ in $\mathcal{D}$. Then, $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]$.*

*Proof* It is easy to prove by induction on the structure of $W$.

Base case: If $W$ is atomic, there are two sub-cases.

(1) $W$ is of the form $A_1(\vec{t}\,)=A_2(\vec{t'})$ for some action terms $A_1(\vec{t}\,)$ and $A_2(\vec{t'})$. If $A_1 \neq A_2$, we have $\mathcal{R}[W]=\mathit{false}$ by the definition of $\mathcal{R}$ in Section 5.1, and $\mathcal{R}[\epsilon[W]]=\mathcal{R}[\mathit{false}]=\mathit{false}$ by the definitions of $\epsilon$ and $\mathcal{R}$, therefore, $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]$;
else, if $A_1=A_2$ and $A_1, A_2$ are constant action functions, by the definition of $\epsilon$ and $\mathcal{R}$, it is easy to see that $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]=\mathit{true}$;
otherwise, we have $\mathcal{R}[W]=\mathcal{R}[\bigwedge_{i=1}^{|\vec{t}|} t_i=t'_i]$ by the definition of $\mathcal{R}$,
and since $\mathcal{R}[\epsilon[W]]=\mathcal{R}[\epsilon[\bigwedge_{i=1}^{|\vec{t}|} t_i=t'_i]]$ by the definition of $\epsilon$ and $\mathcal{R}$, it is easy to see that $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]$.

(2) Otherwise, $W$ is atomic and not of the above form, by the definition of $\epsilon$, we have $\epsilon[W]=W$, hence $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]$.

Inductive step: $W$ is not atomic and $W$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W'$, or $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol. Then for each sub-case, it is easy to prove that $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]$ by the induction hypothesis. For instance, if $W$ is of the form $W_1 \vee W_2$, then

$$\begin{aligned}
\mathcal{R}[W] &= \mathcal{R}\big[W_1\big] \vee \mathcal{R}\big[W_2\big] && \text{(by the definition of } \mathcal{R}) \\
&= \mathcal{R}\big[\epsilon\big[W_1\big]\big] \vee \mathcal{R}\big[\epsilon\big[W_2\big]\big] && \text{(by the induction hypothesis)} \\
&= \mathcal{R}\big[\epsilon\big[W_1\big] \vee \epsilon\big[W_2\big]\big] && \text{(by the definition of} \mathcal{R}) \\
&= \mathcal{R}\big[\epsilon\big[W_1 \vee W_2\big]\big] && \text{(by the definition of } \epsilon) \\
&= \mathcal{R}[\epsilon[W]].
\end{aligned}$$

It is easy to see that for other sub-cases, such as $W_1 \wedge W_2$, $\neg W'$, and $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers), the proof is very similar to the sub-case of $W_1 \vee W_2$ and therefore details are omitted here.
Overall, $\mathcal{R}[W]=\mathcal{R}[\epsilon[W]]$ for any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ in $\mathcal{D}$. □

Moreover, according to the definition of $\epsilon$, it is straightforward to prove the following property of $\epsilon$. Because the proof is rather obvious, it is omitted here.

*Property 2* Given any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ whose size is $m$, i.e., $m=size(W)$, it takes no more than $m$ steps to obtain $\epsilon[W]$, and the size of $\epsilon[W]$ is no more than $3m$.

We also recursively define a *one-step* regression operator $\rho$ for any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$, which has no appearances of *Poss*, such that it performs one step of $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regression on each fluent in $W$. This operator $\rho$ will also be useful in the proof of Lemma 1. The formal definition of $\rho$ is as follows, where $\sigma$ denotes the term of sort *situation*, and $\alpha$ denotes the term of sort *action*.

**Definition 7**

– If $W$ is not atomic, i.e., $W$ is of the form $W_1 \vee W_2$, $W_1 \wedge W_2$, $\neg W'$, or $Qv.W'$ where $Q$ represents a quantifier (including counting quantifiers) and $v$ represents a variable symbol, then

$$\rho\big[W_1 \vee W_2\big] = \rho\big[W_1\big] \vee \rho\big[W_2\big],\ \ \rho\big[\neg W'\big] = \neg\rho\big[W'\big],$$
$$\rho\big[W_1 \wedge W_2\big] = \rho\big[W_1\big] \wedge \rho\big[W_2\big],\ \ \rho\big[Qv.W'\big] = Qv.\rho\big[W'\big].$$

– Otherwise, $W$ is an atom. There are several cases.

  **a.** If $W$ is a situation independent atom, or $W$ is a concept or role uniform in $S_0$, then

$$\rho[W] = W.$$

**b.** If $W$ is a defined dynamic concept, so it has the form $G(t, \sigma)$ for some object term $t$ and situation term $\sigma$, then there must be a TBox axiom for $G$ of the form $G(x, s) \equiv \phi_G(x, s)$. Because of the restrictions of the language $\mathcal{L}_{sc}^{\mathcal{C}^2}$, term $t$ can only be a variable $x$, $y$ or a constant. Then, we use the lazy unfolding technique as follows:

$$\rho[W] = \begin{cases} \rho\big[\phi_G(t, \sigma)\big] & \text{if } t \text{ is not variable } y, \\ \rho\big[\widetilde{\phi_G}(y, \sigma)\big] & \text{otherwise.} \end{cases}$$

**c.** If $W$ is a primitive dynamic concept (a dynamic role, respectively), it has the form $F(t_1, do(\alpha, \sigma))$ or $F(t_1, t_2, do(\alpha, \sigma))$ for some terms $t_1$ (and $t_2$) of sort *object*, term $\alpha$ of sort *action* and term $\sigma$ of sort *situation*. Then there must be an SSA for fluent $F$, whose detailed syntax is Eq. 5. Because of the restriction of the language $\mathcal{L}_{sc}^{\mathcal{C}^2}$, the terms $t_1$ and $t_2$ can only be a variable $x$, $y$ or a constant $O$ and $\alpha$ can only be an action function with no more than two arguments of sort *object*. Then, when $W$ is a concept,

$$\rho[W] = \begin{cases} \Phi_F\big(t_1, \alpha, \sigma\big) & \text{if } t_1 \text{ is not variable } y, \\ \widetilde{\Phi}_F(y, \alpha, \sigma) & \text{otherwise, i.e., if } t_1 = y; \end{cases}$$

and, when $W$ is a role,

$$\rho[W] = \begin{cases} \big(\exists y\big)\big(x{=}y \wedge \Phi_F(x, y, \alpha, \sigma)\big) & \text{if } t_1{=}x, t_2{=}x, \\ \big(\exists x\big)\big(y{=}x \wedge \Phi_F(x, y, \alpha, \sigma)\big) & \text{if } t_1{=}y, t_2{=}y, \\ \widetilde{\Phi}_F\big(y, t_2, \alpha, \sigma\big) & \text{if } t_1{=}y, t_2 \in \{x, O\} \text{ or } t_1{=}O, t_2{=}x, \\ \Phi_F\big(t_1, t_2, \alpha, \sigma\big) & \text{otherwise.} \end{cases}$$

Note that the operator $\rho$ ($\mathcal{R}$, respectively) is generally defined for any $C^2$ regressable formula. For some restricted BATs, when applying $\mathcal{R}$ to any regressable formulas that is in $FO_{DL}$ or $FO_{DL+}$ respectively (with any situation term suppressed), below we will show that the resulting formula is still in $FO_{DL}$ or $FO_{DL+}$ respectively (with any situation term suppressed) via $\epsilon$ and $\rho$. In particular, notice that any formula in $FO_{DL}$ or $FO_{DL+}$ respectively does not have any predicate of the form $x = x$, $y = y$, $R(x, x)$ or $R(y, y)$. Hence, we do not need to worry about the first two sub-cases of the one-step regression $\rho$ for roles, where object terms can be $(x, x)$ or $(y, y)$.

Similar to the proof of Property 1, we can prove the following property for $\rho$ by using induction on the structure of formulas.

*Property 3* For any given BAT $\mathcal{D}$ and an $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ in $\mathcal{D}$, we have that $\rho[W]$ is still $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable and $\mathcal{D} \models W \equiv \rho[W]$.

In addition, also using induction on the structure of the formulas, it is straightforward to prove the following property, which is useful in the proof of Lemma 1. Because the proof is rather obvious, it is omitted here.

*Property 4* Consider a BAT $\mathcal{D}$ in the language of $\mathcal{L}_{sc}^{\mathcal{C}^2}$, if a given $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ is uniform in $do(\alpha, S)$ for some ground action $\alpha$ and ground situation $S$,

and predicate *Poss* does not appear in $W$, then $\rho[W]$ is uniform in $S$ and there is still no appearance of *Poss*.

Again, similar to the proof of Lemma 7, we can prove the following lemma.

**Lemma 8** *Consider any given BAT $\mathcal{D}$, the $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regression operator $\mathcal{R}$ defined in Section 5.1, and any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ in $\mathcal{D}$. Then, $\mathcal{R}[W] = \mathcal{R}[\rho[W]]$.*

Moreover, according to the definition of $\rho$, it is straightforward to prove the following property of $\rho$.

*Property 5* Consider any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ with a background BAT $\mathcal{D}$, including a finite set $\mathcal{D}_{TBox}$ of acyclic TBox axioms. Assume that there is no appearance of *Poss* in $W$. Let $m = size(W)$, $h = \max(2, sizeSSA(\mathcal{D}))$, $h_0 = |\mathcal{D}_{TBox}|$ (i.e., the size of $\mathcal{D}_{TBox}$) and $h_1 = \max_G\{size(\Phi_G) \mid G(x) \equiv \Phi_G(x)$ is a TBox axiom in $\mathcal{D}_{TBox}\}$ if $h_0 \neq 0$, or $h_1 = 0$ otherwise. Notice that $h$, $h_0$ and $h_1$ are fixed when $\mathcal{D}$ is given. Since TBox is acyclic, and in the worst case, $\rho$ may unfold TBox completely before it terminates, it takes no more than $m(h_0 + 1)$ steps to obtain $\rho[W]$, whose size is no more than $c \cdot m$, where $c$ is a constant that equals to $h_1^{h_0} h$.

We also have the following corollary of Lemmas 7 and 8.

**Corollary 4** *Consider any given BAT $\mathcal{D}$, the $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regression operator $\mathcal{R}$ defined in Section 5.1, and any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula $W$ in $\mathcal{D}$. Then, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]]$.*

*Proof* By Lemma 7, $\mathcal{R}[\epsilon[\rho[W]]] = \mathcal{R}[\rho[W]]$, and by Lemma 8, $\mathcal{R}[\rho[W]] = \mathcal{R}[W]$. Therefore, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]]$. □

Now, we provide a detailed proof of Lemma 1 in Section 5.4.

**Lemma 1** (Section 5.4) *Consider a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{\mathcal{C}^2}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_{\mathcal{T}}$ are $\mathcal{ALCO}(U)$-restricted. Let $W$ be any $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable formula in $\mathcal{D}$ that is uniform in a ground situation $S$ and has no appearance of Poss. Let $n = sitLength(S)$ and $m = size(W)$. Then if $W$ with the situation term $S$ suppressed is in $FO_{DL}$, there is a formula $\Phi_W$ in $FO_{DL}$ such that $\mathcal{R}[W]$ is equivalent to $\Phi_W[S_0]$. It takes no more than $c \cdot n \cdot size(\Phi_W)$ steps of deduction from $\mathcal{R}[W]$ (with $S_0$ suppressed) to find such $\Phi_W$ for some constant number $c$. Moreover, $size(\Phi_W)$ is in $O(2^{hmn+3h^2n^2})$ for some positive integer $h$. That is, the size of $\Phi_W$ is no more than exponential in the size of $W$.*

*Proof* Without loss of generality, we assume that there is no defined concept in $W \in FO_{DL}$. Otherwise, each defined concept will be replaced by its definitions from the TBox axioms with finite steps of $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regression, which causes no more than a constant increase in the size of the original formula, because TBox is fixed (once $\mathcal{D}$ is given), TBox is acyclic, there are only finitely many TBox axioms and the size of the formula on the RHS of each TBox axiom is limited from above by a constant.

We will first prove such formula $\phi_W \in FO_{DL}$ always exists, and then estimate the size of the formula. Note that the proof of Lemma 1 is not obvious, given that the

regression operator itself does not generally preserve the syntactic form of a formula on its input.

Now, we define a notation for later convenience. If $W$ is a formula uniform in any situation $s$, we denote the formula with all situation terms suppressed (if any) in $W$ simply as $W[-s]$ . Moreover, to simplify the presentation of the proof, below we write $W_1 \equiv W_2$ whenever $\models W_1 \equiv W_2$ for any formulas $W_1$ and $W_2$. First, we are going to prove the following more specific statement below wrt the given BAT $\mathcal{D}$:

(**Statement 1**) *Consider any ground situation $S$ and a $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$ with the background BAT $\mathcal{D}$, where $W$ is uniform in $S$ and has no occurrences of Poss. If $W[-S]$ is in $FO^x_{DL}$ ($FO^y_{DL}$, respectively), there is a formula $\varphi$ in $FO^x_{DL}$ ($FO^y_{DL}$, respectively) such that $\mathcal{R}[W]$ is equivalent to $\varphi[S_0]$.*

The structure of our proof will consist of two nested proofs by induction, where the internal proof by induction will include an analysis of many sub-cases. The main proof will proceed by induction on the length of $S$, i.e., the number of actions involved in $S$. Inside the inductive step of this proof, we will prove the statement by induction on the structure of a $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$. In the latter, the most time consuming parts will be two cases: when $W$ is a primitive dynamic concept (a fluent with one object argument and one situation argument); or, when $W$ is of the form $\exists y.R(x, y, S) \wedge W_1(y)[S]$ for some dynamic role $R$ (a fluent with two object arguments and one situation argument) and formula $W_1(y) \in FO^y_{DL}$. These two cases are laborious and require an analysis of numerous sub-cases depending on the structure of logical formulas in SSAs.

*Base case of the induction on the length of $S$*: If $S = S_0$, then let $\varphi = W[-S_0]$, and it is trivial to see that Statement (1) is true.

*Inductive step of the induction on the length of $S$*: Now, without loss of generality, we assume that $S = do(\alpha, S_1)$ and Statement (1) is true for any $\mathcal{L}^{C^2}_{sc}$ regressable formula $W'$ that is uniform in $S_1$ and has no appearance of *Poss*. We prove Statement (1) for any $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$ that is uniform in $S$ and has no appearance of *Poss* by induction on the structure of $W[-S]$.

Since every formula in $FO^y_{DL}$ is a dual formula to a formula in $FO^x_{DL}$, the proof for Statement (1) where $W[-S]$ is in $FO^y_{DL}$ is "*dual*" to the proof for Statement (1) where $W[-S]$ is in $FO^x_{DL}$, in the sense that we only need to replace every appearance of $x$ with $y$ and $y$ with $x$. Hence, below we will only provide a detailed proof for Statement (1) where $W[-S] \in FO^x_{DL}$, and omit details of the proof of Statement (1) where $W[-S] \in FO^y_{DL}$.

In order to prove Statement (1) for a ground situation $S$, we will prove both Statement (1) and the following statement (**Statement (2)**) for $S$ together by induction on the structure of $W$:

(**Statement 2**) *For any $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$ that is uniform in $S$ (where $S = do(\alpha, S_1)$) and has no appearance of Poss, if $W[-S]$ is in $FO^x_{DL}$ ($FO^y_{DL}$, respectively), then there is a formula $\varphi$ in $FO^x_{DL}$ ($FO^y_{DL}$, respectively), which can be found in no more than $c \cdot size(\varphi)$ number of steps for some constant positive integer $c$. Moreover, $\varphi[S_1]$ is equivalent to $\epsilon[\rho[W]]$, and $\varphi[S_1]$ is $\mathcal{L}^{C^2}_{sc}$ regressable with no appearance of Poss.*

*Base case of the induction on the structure of* $W[-S]$: First, we consider when $W[-S]$ is in $FO^x_{DL}$ and is atomic. There are in total three cases (a–c) below.

a. $W[-S]$ is either *true* or *false*. Then, $\epsilon[\rho[W]][-S_1]$ is still *true* or *false*, which is in $FO^x_{DL}$; and, $(\mathcal{R}[W])[-S_0]$ is still *true* or *false*, which is in $FO^x_{DL}$. Hence, it is trivial to see that Statement (1) and Statement (2) hold.

b. $W[-S]$ is of the form $x=b$ for some constant $b$. Then, $\epsilon[\rho[W]][-S_1]$ is still $x=b$, which is in $FO^x_{DL}$; and, $(\mathcal{R}[W])[-S_0]$ is still $x=b$, which is in $FO^x_{DL}$. Again, it is trivial to see that Statement (1) and Statement (2) hold.

c. $W[-S]$ is a monadic predicate. Then, there are two sub-cases: If $W$ is situation-independent, then $\epsilon[\rho[W]][-S_1]=W=W[-S]$, which is in $FO^x_{DL}$; and, $(\mathcal{R}[W])[-S_0]=W=W[-S]$, which is in $FO^x_{DL}$. Again, it is trivial to see that Statement (1) and Statement (2) hold.
Otherwise, $W=F(x, S)$ for some fluent $F$. Assume that fluent $F(x, s)$ has an SSA of the form Eq. 5, whose context conditions (with situation terms suppressed) are all in $FO_{DL}$. Depending on whether the context conditions are in $FO^x_{DL}$ (e.g., cases (1–12) in Table 11) or in $FO^y_{DL}$ (e.g., cases (1'–12') in Table 11), what variables appear in action functions and/or in the conditions (none, $x$ only, $y$ only, $x$ and $y$), and whether or not the variables are quantified, the SSA of $F$ is

$$F(x, do(a, s)) \equiv \bigvee_{i=1}^{m_+} \phi_i^+(x, a, s) \vee F(x, s) \wedge \neg \left( \bigvee_{j=1}^{m_-} \phi_j^-(x, a, s) \right), \tag{12}$$

where each $\phi_i^+(x, a, s)$ ($\phi_j^-(x, a, s)$, respectively) is a formula that has the syntactic form of one of the following cases listed in Table 11 and all the cases we described in Note 1 below. Recall that we prove this lemma for those SSAs which have $\mathcal{ALCO}(U)$-restricted context formulas only. Notice that in Table 11, $\psi(x)$ ($\psi(y)$, respectively) is a formula in $FO^x_{DL}$ ($FO^y_{DL}$, respectively) with *at most* one free variable $x$ ($y$, respectively). In cases (1) and (1'), $A$ represents some constant action function. In cases (2)–(6) and (2')–(6'), $A$ represents some unary action function name. And, in cases (7)–(12) and (7')–(12'), $A$ represents some

**Table 11** All possible syntactic forms for $\phi_i^+(x, a, s)$ or $\phi_j^-(x, a, s)$ in Eq. 12

| | | | |
|---|---|---|---|
| 1 | $a=A \wedge \psi(x)[s]$ | 1' | $a=A \wedge [\exists y.]\psi(y)[s]$ |
| 2 | $a=A(x) \wedge \psi(x)[s]$ | 2' | $a=A(x) \wedge [\exists y.]\psi(y)[s]$ |
| 3 | $\exists x.a=A(x) \wedge \psi(x)[s]$ | 3' | $\exists x.a=A(x) \wedge [\exists y.]\psi(y)[s]$ |
| 4 | $\exists x(a=A(x)) \wedge \psi(x)[s]$ | 4' | $\exists x(a=A(x)) \wedge [\exists y.]\psi(y)[s]$ |
| 5 | $\exists y.a=A(y) \wedge \psi(x)[s]$ | 5' | $\exists y.a=A(y) \wedge \psi(y)[s]$ |
| 6 | $\exists y(a=A(y)) \wedge \psi(x)[s]$ | 6' | $\exists y(a=A(y)) \wedge [\exists y.]\psi(y)[s]$ |
| 7 | $\exists y.a=A(x, y) \wedge \psi(x)[s]$ | 7' | $\exists y.a=A(x, y) \wedge \psi(y)[s]$ |
| 8 | $\exists y(a=A(x, y)) \wedge \psi(x)[s]$ | 8' | $\exists y(a=A(x, y)) \wedge [\exists y.]\psi(y)[s]$ |
| 9 | $\exists x.\exists y.a=A(x, y) \wedge \psi(x)[s]$ | 9' | $\exists x.\exists y.a=A(x, y) \wedge \psi(y)[s]$ |
| 10 | $\exists x.\exists y(a=A(x, y)) \wedge \psi(x)[s]$ | 10' | $\exists x.\exists y(a=A(x, y)) \wedge [\exists y.]\psi(y)[s]$ |
| 11 | $\exists y.\exists x(a=A(x, y)) \wedge \psi(x)[s]$ | 11' | $\exists y.\exists x(a=A(x, y)) \wedge \psi(y)[s]$ |
| 12 | $\exists y(\exists x(a=A(x, y))) \wedge \psi(x)[s]$ | 12' | $\exists y(\exists x(a=A(x, y))) \wedge [\exists y.]\psi(y)[s]$ |

binary action function name. Moreover, in Table 11, $[\exists y.]$ represents that $\exists y.$ only appears when $\psi(y)$ has a free variable $y$. Note that we need $\exists y$ to bound the variable $y$ in $\psi(y)$, because otherwise $W$ would not be in $FO^x_{DL}$, but would have both $x$ and $y$ as free variables. To make sure that we have exhausted all the possibilities, the cases we listed could include some duplications. For example, case (6) in fact is the same as case (4) by renaming; case (1') is in fact the same as case (1), because $[\exists y.]\psi(y)$ is a formula in $FO^x_{DL}$ (according to the definition of $FO^x_{DL}$). We require these restrictions to the format of the SSAs of dynamic roles to make sure that we are able to get a formula that can be still expressed (as an equivalent formula) in $FO_{DL}$ after regression.

*Note 1* Let $O$, $O_1$ and $O_2$ be some constant objects. There are also cases for $a = A(O)$ (or $a = A(O_1, O_2)$, respectively) in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $F$, which can be proved similarly to the cases in Table 11 where $a = A$. There are also cases for $a = A(O_1, x)$ (or $a = (x, O_1)$, respectively) in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $F$, which can be proved similarly to the cases in Table 11 where $a = A(x)$. There are also cases for $a = A(O_1, y)$ (or $a = (y, O_1)$, respectively) in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $F$, which can be proved similarly to the cases in Table 11 where $a = A(y)$. There are also cases for $a = A(y, x)$ in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $F$, which can be proved similarly to the cases in Table 11 where $a = A(x, y)$. Notice that using the unique name axioms for object constants, we can replace all (in)equalities between object constants with either *true* or *false* in the resulting formula that is equivalent to $\epsilon[\rho[W]]$ for any $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$. Moreover, such deduction takes at most a constant number of steps wrt the size of the resulting formula.

We first prove case by case for all possible syntactic forms of $\phi_i^+(x, \alpha, S_1)$ ($\phi_j^-(x, \alpha, S_1)$, respectively), that there exist some formulas in $FO^x_{DL}$ for any $i$ ($j$, respectively) such that they are logically equivalent to $\epsilon[\phi_i^+(x, \alpha, S_1)]$ ($\epsilon[\phi_j^-(x, \alpha, S_1)]$, respectively) when the situation term $S_1$ is restored. Later, it will be convenient to call these formulas as $\nu_i^+$ (or $\nu_j^-$, respectively). Moreover, finding the equivalent formulas takes a constant number of steps of deduction wrt to the size of $\epsilon[\phi_i^+(x, \alpha, S_1)]$ ($\epsilon[\phi_j^-(x, \alpha, S_1)]$, respectively).

Here is one trivial sub-case: if the function name of $\alpha$ is not $A$, then in each of the aforementioned cases (1)–(12), (1')–(12') and Note 1, $\epsilon$ of each formula ($\phi_i^+$ or $\phi_j^-$) equals *false*, which is still in $FO^x_{DL}$. Hence, below we only discuss the case when $\alpha$ has the same function name (with the same number of arguments) as the given action function name in each case, and we let $O$, $O_1$ and $O_2$ be some constants. Notice that in case (1), since the context condition $\psi(x)$ is in $FO^x_{DL}$, $\psi(x)[S_1]$ does not contain any equality between action terms, hence $\epsilon[\psi(x)[S_1]] = \psi(x)[S_1]$. The same reasoning will be used in other cases, and detailed explanations are omitted to avoid repetition.

(1) $a = A \wedge \psi(x)[s]$
Assume that $\alpha = A$, then

$$\begin{aligned} & \epsilon[\alpha = A \wedge \psi(x)[S_1]] \\ = \; & \epsilon[\alpha = A] \wedge \epsilon[\psi(x)[S_1]]] \\ = \; & \mathit{true} \wedge \psi(x)[S_1] \;\equiv\; \psi(x)[S_1]. \end{aligned}$$

Clearly, $\psi(x)$ is in $FO^x_{DL}$, and it takes no steps of logical deductions to find the equivalent formula.

(1') $a = A \wedge [\exists y.]\psi(y)[s]$

Because $[\exists y.]\psi(y)$ is in fact in $FO^x_{DL}$, the proof for (1') is the same as for (1).

(2) $a = A(x) \wedge \psi(x)[s]$

Assume that $\alpha = A(O)$, then

$$\begin{aligned} & \epsilon[\alpha = A(x) \wedge \psi(x)[S_1]] \\ = \; & \epsilon[A(O) = A(x)] \wedge \psi(x)[S_1] \\ = \; & (x = O) \wedge \psi(x)[S_1] \\ = \; & (x = O \wedge \psi(x))[S_1]. \end{aligned}$$

Clearly, given that $\psi(x)$ is in $FO^x_{DL}$, we have that $x = O \wedge \psi(x)$ is in $FO^x_{DL}$, and it takes no steps of logical deductions.

(2') $a = A(x) \wedge [\exists y.]\psi(y)[s]$

Because $[\exists y.]\psi(y)$ is in fact in $FO^x_{DL}$, the proof for (2') is the same as for (2).

(3) $\exists x.a = A(x) \wedge \psi(x)[s]$

Assume that $\alpha = A(O)$, then

$$\begin{aligned} & \epsilon[\exists x.\alpha = A(x) \wedge \psi(x)[S_1]] \\ = \; & \exists x.\epsilon[A(O) = A(x) \wedge \psi(x)[S_1]] \\ = \; & (\exists x.x = O \wedge \psi(x))[S_1]. \end{aligned}$$

Clearly, the resulting formula $(\exists x.x = O \wedge \psi(x))$ is in $FO^x_{DL}$ since any closed sentence is in both $FO^x_{DL}$ and $FO^y_{DL}$. It takes no steps of logical deductions.

(3') $\exists x.a = A(x) \wedge [\exists y.]\psi(y)[s]$

Note that $[\exists y.]\psi(y)$ has no free variable $x$, hence $x$ is in fact only quantified over $a = A(x)$, hence case (3') is equivalent to case (4') below.

(4) $\exists x(a = A(x)) \wedge \psi(x)[s]$

Assume that $\alpha = A(O)$, then

$$\begin{aligned} & \epsilon[\exists x(\alpha = A(x)) \wedge \psi(x)[S_1]] \\ = \; & \epsilon[\exists x(A(O) = A(x)) \wedge \psi(x)[S_1]] \\ = \; & (\exists x(x = O) \wedge \psi(x))[S_1]. \end{aligned}$$

Because $\exists x(x = O)$ and $\psi(x)$ are in $FO^x_{DL}$, $\exists x(x = O) \wedge \psi(x)$ is in $FO^x_{DL}$. It takes no steps of logical deductions.

(4') $\exists x(a = A(x)) \wedge [\exists y.]\psi(y)[s]$

Because $[\exists y.]\psi(y)$ is in fact in $FO^x_{DL}$, case (4') is a special case of case (2).

(5) $\exists y.a = A(y) \wedge \psi(x)[s]$

Assume that $\alpha = A(O)$, then

$$\begin{aligned} & \epsilon[\exists y.\alpha = A(y) \wedge \psi(x)[S_1]] \\ \equiv \; & \epsilon[\exists y(A(O) = A(y)) \wedge \psi(x)[S_1]] \\ = \; & \exists y(y = O) \wedge \psi(x)[S_1] \\ = \; & (\exists y(y = O) \wedge \psi(x))[S_1]. \end{aligned}$$

Clearly, the closed formula $\exists y(y = O) \wedge \psi(x)$ is in $FO^x_{DL}$. It takes one step of logical deductions to minimize the quantification scope of $\exists y$. Note that although the resulting formula can be simplified to $\psi(O)[S_1]$, but our regression operator and the operator $\epsilon$ does not do any simplification.

(5') $\exists y.a = A(y) \wedge \psi(y)[s]$

Assume that $\alpha = A(O)$, then

$$\begin{aligned} & \epsilon[\exists y.\alpha = A(y) \wedge \psi(y)[S_1]] \\ = \; & \epsilon[\exists y.A(O) = A(y) \wedge \psi(y)[S_1]] \\ = \; & \exists y.\epsilon[A(O) = A(y) \wedge \psi(y)[S_1]] \\ = \; & \exists y.y = O \wedge \psi(y)[S_1] \\ = \; & (\exists y.y = O \wedge \psi(y))[S_1]. \end{aligned}$$

$\psi(y)$ is in $FO^{y}_{DL}$, hence $y = O \wedge \psi(y)$ is in $FO^{y}_{DL}$ and $\exists y.y = O \wedge \psi(y)$ is in $FO^{x}_{DL}$. It takes no steps of logical deductions.

(6) $\exists y(a = A(y)) \wedge \psi(x)[s]$
Case (6) is equivalent to case (5), because in case (5) the quantification range of $y$ is in fact only over $a = A(y)$. Hence, the statement is true for case (6) by the definition of $FO^{x}_{DL}$.

(6') $\exists y(a = A(y)) \wedge [\exists y.]\psi(y)[s]$ Because $[\exists y.]\psi(y)$ is in $FO^{x}_{DL}$, case (6') is a special case of (6).

(7) $\exists y.a = A(x, y) \wedge \psi(x)[s]$
Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y.\alpha = A(x, y) \wedge \psi(x)[S_1]] \\ = \ & \epsilon[\exists y.A(O_1, O_2) = A(x, y) \wedge \psi(x)[S_1]] \\ = \ & (\exists y.x = O_1 \wedge y = O_2 \wedge \psi(x))[S_1] \\ \equiv \ & (x = O_1 \wedge \exists y(y = O_2) \wedge \psi(x))[S_1]. \end{aligned}$$

Clearly, given that $\psi(x)$ is in $FO^{x}_{DL}$, we have that $x = O_1 \wedge \exists y(y = O_2) \wedge \psi(x)$ is in $FO^{x}_{DL}$ by the definition of $FO^{x}_{DL}$. It takes one step of logical deductions to minimize the quantification scope of $\exists y$.

(7') $\exists y.a = A(x, y) \wedge \psi(y)[s]$
Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y.\alpha = A(x, y) \wedge \psi(y)[S_1]] \\ = \ & \epsilon[\exists y.A(O_1, O_2) = A(x, y) \wedge \psi(y)[S_1]] \\ = \ & (\exists y.x = O_1 \wedge y = O_2 \wedge \psi(y))[S_1] \\ \equiv \ & (x = O_1 \wedge (\exists y.y = O_2 \wedge \psi(y)))[S_1]. \end{aligned}$$

Clearly, given that $\psi(y)$ is in $FO^{y}_{DL}$, we have that $x = O_1 \wedge (\exists y.y = O_2 \wedge \psi(y))$ is in $FO^{x}_{DL}$ by the definition of $FO^{x}_{DL}$. It takes one step of logical deductions to minimize the quantification scope of $\exists y$.

(8) $\exists y(a = A(x, y)) \wedge \psi(x)[s]$
Case (8) is equivalent to case (7), because in case (7) the quantification range of $y$ is in fact only over $a = A(x, y)$.

(8') $\exists y(a = A(x, y)) \wedge [\exists y.]\psi(y)[s]$
Because $[\exists y.]\psi(y)$ is in $FO^{x}_{DL}$, case (8') is a special case of case (8).

(9) $\exists x.\exists y.a = A(x, y) \wedge \psi(x)[s]$
Case (9) is equivalent to case (10), because in case (9) the quantification range of $y$ is in fact only over $a = A(x, y)$.

(9') $\exists x.\exists y.a = A(x, y) \wedge \psi(y)[s]$
Case (9') is equivalent to case (11'), because in case (9') the quantification range of $x$ is in fact only over $a = A(x, y)$.

(10) $\exists x.\exists y(a = A(x, y)) \wedge \psi(x)[s]$
Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists x.\exists y(\alpha = A(x, y)) \wedge \psi(x)[S_1]] \\ = \ & \epsilon[\exists x.\exists y(\alpha = A(x, y)) \wedge \psi(x)[S_1]] \\ = \ & \epsilon[\exists x.\exists y(A(O_1, O_2) = A(x, y))\psi(x)[S_1]] \\ = \ & (\exists x.\exists y(x = O_1 \wedge y = O_2) \wedge \psi(x))[S_1] \\ \equiv \ & (\exists x.x = O_1 \wedge \exists y(y = O_2) \wedge \psi(x))[S_1]. \end{aligned}$$

It is easy to see that the resulting formula is in $FO^{x}_{DL}$, and it takes no more than two steps of logical deductions to do the transformation.

(10') $\exists x.\exists y(a = A(x, y)) \wedge [\exists y.]\psi(y)[s]$
Because $[\exists y.]\psi(y)$ is in $FO^{x}_{DL}$, case (10') is a special case of case (10).

(11) $\exists y.\exists x(a = A(x, y)) \wedge \psi(x)[s]$

Case (11) is equivalent to case (12), because in case (11) the quantification range of $y$ is in fact only over $a = A(x, y)$.

(11') $\exists y.\exists x(a = A(x, y)) \wedge \psi(y)[s]$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned}
&\epsilon[\exists y.\exists x(\alpha = A(x, y)) \wedge \psi(y)[S_1]] \\
&= \epsilon[\exists y.\exists x(A(O_1, O_2) = A(x, y)) \wedge \psi(y)[S_1]] \\
&= \exists y.\exists x(x = O_1 \wedge y = O_2) \wedge \psi(y)[S_1] \\
&\equiv (\exists y.\exists x(x = O_1) \wedge y = O_2 \wedge \psi(y))[S_1] \\
&\equiv \exists x(x = O_1) \wedge\wedge y(y = O_2 \wedge \psi(y))[S_1].
\end{aligned}$$

It is easy to see that $\exists y.\exists x(x = O_1) \wedge y = O_2 \wedge \psi(y)$ is in $FO^x_{DL}$, and it takes one step to minimize the scope of $\exists x$ and another one step to minimize the scope of $\exists y$.

(12) $\exists y(\exists x(a = A(x, y))) \wedge \psi(x)[s]$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned}
&\epsilon[\exists y(\exists x(\alpha = A(x, y))) \wedge \psi(x)[S_1]] \\
&= \epsilon[\exists y(\exists x(A(O_1, O_2) = A(x, y)) \wedge \psi(x)[S_1]] \\
&= \exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(x)[S_1] \\
&\equiv (\exists y(\exists x(x = O_1) \wedge y = O_2) \wedge \psi(x))[S_1] \\
&\equiv (\exists x(x = O_1) \wedge \exists y(y = O_2) \wedge \psi(x))[S_1].
\end{aligned}$$

It is easy to see that $\exists y(\exists x(x = O_1) \wedge y = O_2) \wedge \psi(x)$ is in $FO^x_{DL}$, and it takes one step of logical deductions to minimize the scope of $\exists x$ and another one step to minimize the scope of $\exists y$.

(12') $\exists y(\exists x(a = A(x, y))) \wedge [\exists y.]\psi(y)[s]$

Because $[\exists y.]\psi(y)$ is in $FO^x_{DL}$, case (12') is a special case of case (12).

When $a$ is substituted by a ground action $\alpha$ and $s$ is substituted by a ground situation $S_1$, by the definition of regression using the SSA of the form (12), $\epsilon$ and $\rho$,

$$\begin{aligned}
\epsilon[\rho[F(x, S)]] &= \epsilon[\rho[F(x, do(\alpha, S_1))]] \\
&= \epsilon\left[\bigvee_{i=1}^{m_+} \phi_i^+(x, \alpha, S_1) \vee F(x, S_1) \wedge \neg\left(\bigvee_{j=1}^{m_-} \phi_j^-(x, \alpha, S_1)\right)\right] \\
&\qquad \text{(by the definition of } \rho \text{ and (12)),} \\
&= \bigvee_{i=1}^{m_+} \epsilon[\phi_i^+(x, \alpha, S_1)] \vee F(x, S_1) \wedge \neg\left(\bigvee_{j=1}^{m_-} \epsilon\left[\phi_j^-(x, \alpha, S_1)\right)\right] \\
&\qquad \text{(by the definition of } \epsilon\text{),} \\
&\equiv \bigvee_{i=1}^{m_+} \nu_i^+(x)[S_1] \vee F(x, S_1) \wedge \neg\left(\bigvee_{j=1}^{m_-} \nu_j^-(x)[S_1]\right. \\
&\qquad \text{(according to the proof of cases (1)–(12),(1')–(12')),} \\
&\equiv \left(\bigvee_{i=1}^{m_+} \nu_i^+(x) \vee F(x) \wedge \neg\left(\bigvee_{j=1}^{m_-} \nu_j^-(x)\right)\right)[S_1], \qquad (13)
\end{aligned}$$

where each $\nu_i^+(x)$ ($\nu_j^-(x)$, respectively) is a formula in $FO^x_{DL}$ that has at most one free variable $x$. Each $\nu_i^+(x)$ ($\nu_j^-(x)$, respectively) is logically equivalent to $\epsilon[\phi_i^+(x,\alpha,S_1)][-S_1]$ ($\epsilon[\phi_j^-(x,\alpha,S_1)][-S_1]$, respectively). Clearly, the formula on the RHS of Eq. 13 is regressable, uniform in $S_1$, and is in $FO^x_{DL}$ (with $S_1$ suppressed) according to the definition of the set $FO^x_{DL}$. Moreover, it takes only a constant number of steps wrt the size of the resulting formula to find the equivalent formula. Then, using the induction hypothesis for situation $S_1$ and Corollary 4 (i.e., $\mathcal{R}[F(x,S)]=\mathcal{R}[\epsilon[\rho[F(x,S)]]]$), we have that $(\mathcal{R}[F(x,do(\alpha,S_1))])[-S_0]$ is still be equivalent to some formula in $FO^x_{DL}$.

Similarly, we can prove Statements (1) and (2) for the case when $W[-S]$ is in $FO^y_{DL}$ and is atomic.

*Inductive step of the induction on the structure of* $W[-S]$: Now, we complete our remaining cases when $W$ (hence, $W[-S]$ ) is not atomic. Recall that $S=do(\alpha,S_1)$. In total, there are four cases (a–d), as follows.

a. $W[-S]$ is of the form $\neg W_1$, where $W_1 \in FO^x_{DL}$.
It is obvious that $W=\neg W_1[S]$, and $\epsilon[\rho[W]]=\neg\epsilon[\rho[W_1[S]]]$. Moreover, by the induction hypothesis on the structure of $W$, there is a formula $\phi_1 \in FO^x_{DL}$ such that $\epsilon[\rho[W_1[S]]] \equiv \phi_1[S_1]$, which is regressable, uniform in $S_1$, has no appearance of *Poss* and can be found $c\cdot size(\phi_1)$ for some integer $c$. Hence, $\epsilon[\rho[W]] \equiv \neg\phi_1[S_1]$, and Statement (2) is true for $W$. Then, according to Corollary 4, $\mathcal{R}[W]=\mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[\neg\phi_1[S_1]]$. Next, by the induction hypothesis on $S_1$, $\mathcal{R}[W][-S_0]$ is equivalent to some formula in $FO^x_{DL}$, and it is easy to see that Statement (1) is true for $W$ that is uniform in situation $S$.

b. $W[-S]$ is of the form $W_1\wedge W_2$ or of the form $W_1\vee W_2$, where $W_1, W_2 \in FO^x_{DL}$.
Then, if $W[-S]$ is of the form $W_1\wedge W_2$, it is obvious that $W=(W_1\wedge W_2)[S]$, so $\epsilon[\rho[W]]=\epsilon[\rho[W_1[S]]]\wedge\epsilon[\rho[W_2[S]]]$. By the induction hypothesis on the structure of $W$, there are formulas $\phi_1,\phi_2 \in FO^x_{DL}$ such that $\epsilon[\rho[W_1[S]]] \equiv \phi_1[S_1]$ and $\epsilon[\rho[W_2[S]]] \equiv \phi_2[S_1]$, which are regressable, uniform in $S_1$, has no appearance of *Poss* and can be found in $c_1\cdot(size(\phi_1)$ and $c_2\cdot size(\phi_2)$ steps for some positive constants $c_1$ and $c_2$ respectively. Hence, $\epsilon[\rho[W]] \equiv (\phi_1\wedge\phi_2)[S_1]$, which can be found in $c\cdot(size(\phi_1)+size(\phi_2)+1)$ steps for some integer $c=max(c_1,c_2)$, and Statement (2) is true for $W$. Then, according to Corollary 4, $\mathcal{R}[W]=\mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[(\phi_1\wedge\phi_2)[S_1]]$. Next, by the induction hypothesis on $S_1$, $\mathcal{R}[W][-S_0]$ is equivalent to some formula in $FO^x_{DL}$, and it is easy to see that Statement (1) is true for $W$ that is uniform in $S$.
It is very similar to prove that Statements (1) and (2) are true when $W[-S]$ is of the form $W_1\vee W_2$, and details are omitted here.

c. $W[-S]$ is of the form $[\exists y.]W_1(y)$ or $[\forall y.]W_1(y)$, where $W_1(y)$ is in $FO^y_{DL}$.
Then, if $W[-S]$ is of the form $[\exists y.]W_1(y)$, $\epsilon[\rho[W]]=[\exists y.]\epsilon[\rho[W_1(y)[S]]]$. Since $S=do(\alpha,S_1)$, by the induction hypothesis on the structure of $W$, there is a formula $\phi_1(y) \in FO^y_{DL}$ such that $\epsilon[\rho[W_1(y)[S]]] \equiv \phi_1(y)[S_1]$, which is regressable, uniform in $S_1$, has no appearance of *Poss* and can be found in $c\cdot size(\phi_1(y))$ steps for some integer $c>0$. Hence, $\epsilon[\rho[W]] \equiv ([\exists y.]\phi_1(y))[S_1]$, and Statement (2) is true for $W$. Then, according to Corollary 4, $\mathcal{R}[W]=\mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[([\exists y.]\phi_1(y))[S_1]]$. Next, by the induction hypothesis on $S_1$, $\mathcal{R}[W][-S_0]$ is equivalent to some formula in $FO^x_{DL}$, and it is easy to see that Statement (1) is true for $W$ that is uniform in situation $S$.

It is very similar to prove that Statements (1) and (2) are true when $W[-S]$ is of the form $[\forall y.]W_1(y)$, and details are omitted here.

d. $W[-S]$ is of the form $\exists y.R(x, y) \wedge W_1(y)$ or $\forall y.R(x, y) \supset W_1(y)$, where $R(x, y)$ is a dynamic predicate and $W_1(y)$ is in $FO^y_{DL}$.
We first consider the case when $W[-S]$ is of the form $\exists y.R(x, y) \wedge W_1(y)$. Then, $W = \exists y.R(x, y)[S] \wedge W_1(y)[S]$, and there are two sub-cases.

(d.1) If $R$ is a situation-independent predicate, then $\epsilon[\rho[W]] = [\exists y.]R(x, y) \wedge \epsilon[\rho[W_1(y)[S]]]$. By the induction hypothesis on the structure of $W_1$, there is a formula $\phi_1(y) \in FO^y_{DL}$ such that $\epsilon[\rho[W_1(y)[S]]] \equiv \phi_1(y)[S_1]$, which is regressable, uniform in $S_1$, has no appearance of *Poss* and can be found in $c \cdot size(\phi_1(y))$ steps for some integer $c$. Hence, $\epsilon[\rho[W]] \equiv ([\exists y.]R(x, y) \wedge \phi_1(y))[S_1]$, and Statement (2) is true for $W$. Then, according to Corollary 4, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[([\exists y.]R(x, y) \wedge \phi_1(y))[S_1]]$. Next, by the induction hypothesis on $S_1$, $\mathcal{R}[W][-S_0]$ is equivalent to some formula in $FO^x_{DL}$, and it is easy to see that Statement (1) is true for $W$ that is uniform in situation $S$.

(d.2) Otherwise, if $R$ is a fluent, then $\epsilon[\rho[W]] = [\exists y.]\epsilon[\rho[R(x, y, s)]] \wedge \epsilon[\rho[W_1(y)[S]]]$. Moreover, we need to consider different sub-cases for the SSA of $R$. Notice that according to the definition of a $\mathcal{D}_{ss}$ that is $\mathcal{ALCO}(U)$-restricted, all dynamic roles are both $\mathcal{ALCO}(U)$-restricted and context-free. So, depending on whether the context conditions are in $FO^x_{DL}$, the cases (1)–(16) in Table 12, or $FO^y_{DL}$, the cases (1')–(16') in Table 12, what variables appear in action functions and/or in the conditions (none, $x$ only, $y$ only, $x$ and $y$), and whether or not the variables are quantified, the SSA of $R$ is

$$R(x, y, do(a, s)) \equiv \bigvee_{i=1}^{m_+} \phi_i^+(x, y, a) \vee R(x, y, s) \wedge \neg \left( \bigvee_{j=1}^{m_-} \phi_j^-(x, y, a) \right), \quad (14)$$

**Table 12** All possible syntactic forms for $\phi_i^+(x, y, a)$ or $\phi_j^-(x, y, a)$ in Eq. 14

| | | | |
|---|---|---|---|
| 1 | $a = A \wedge \psi(x)$ | 1' | $a = A \wedge \psi(y)$ |
| 2 | $a = A(x) \wedge \psi(x)$ | 2' | $a = A(x) \wedge \psi(y)$ |
| 3 | $\exists x(a = A(x)) \wedge \psi(x)$ | 3' | $\exists x(a = A(x)) \wedge \psi(y)$ |
| 4 | $\exists x.a = A(x) \wedge \psi(x)$ | 4' | $\exists x.a = A(x) \wedge \psi(y)$ |
| 5 | $a = A(y) \wedge \psi(x)$ | 5' | $a = A(y) \wedge \psi(y)$ |
| 6 | $\exists y(a = A(y)) \wedge \psi(x)$ | 6' | $\exists y(a = A(y)) \wedge \psi(y)$ |
| 7 | $\exists y.a = A(y) \wedge \psi(x)$ | 7' | $\exists y.a = A(y) \wedge \psi(y)$ |
| 8 | $a = A(x, y) \wedge \psi(x)$ | 8' | $a = A(x, y) \wedge \psi(y)$ |
| 9 | $\exists x(a = A(x, y)) \wedge \psi(x)$ | 9' | $\exists x(a = A(x, y)) \wedge \psi(y)$ |
| 10 | $\exists x.a = A(x, y) \wedge \psi(x)$ | 10' | $\exists x.a = A(x, y) \wedge \psi(y)$ |
| 11 | $\exists y(a = A(x, y)) \wedge \psi(x)$ | 11' | $\exists y(a = A(x, y)) \wedge \psi(y)$ |
| 12 | $\exists y.a = A(x, y) \wedge \psi(x)$ | 12' | $\exists y.a = A(x, y) \wedge \psi(y)$ |
| 13 | $\exists y(\exists x(a = A(x, y))) \wedge \psi(x)$ | 13' | $\exists y(\exists x(a = A(x, y))) \wedge \psi(y)$ |
| 14 | $\exists y.\exists x(a = A(x, y)) \wedge \psi(x)$ | 14' | $\exists y.\exists x(a = A(x, y)) \wedge \psi(y)$ |
| 15 | $\exists x.\exists y(a = A(x, y)) \wedge \psi(x)$ | 15' | $\exists x.\exists y(a = A(x, y)) \wedge \psi(y)$ |
| 16 | $\exists x.\exists y.a = A(x, y) \wedge \psi(x)$ | 16' | $\exists x.\exists y.a = A(x, y) \wedge \psi(y)$ |

where each $\phi_i^+(x, y, a)$ ($\phi_j^-(x, y, a)$, respectively) is a situation-independent formula whose syntactic form is one of the following cases listed in Table 12 and the cases we described in Note 2. Recall that we prove the lemma for those SSAs which have $\mathcal{ALCO}(U)$-restricted context formulas only. Notice that in Table 12, $\psi(x)$ ($\psi(y)$, respectively) is a formula in $FO^x_{DL}$ ($FO^y_{DL}$, respectively) with *at most* one free variable $x$ ($y$, respectively). In the cases (1) and (1'), $A$ represents some constant action function. In the cases (2)–(7) and (2')–(7') , $A$ represents some unary action function name. And, in the cases (8)–(16) and (8')–(16'), $A$ represents some binary action function name. Again, to assure that we have exhausted all the possibilities, the cases we listed in Table 12 exhaust all possible combinations of actions, quantifiers and the format of the additional condition $\psi$, and may include some other duplications. For example, in fact, the case (6) is the same as the case (3) by renaming.

*Note 2* For any formula $\psi_1(x)$ ($\psi_1(y)$, respectively) in $FO^x_{DL}$ ($FO^y_{DL}$, respectively), there are also cases where the context conditions are either of the form $[\exists x.]\psi_1(x)$ or of the form $[\exists y.]\psi_1(y)$. However, $[\exists x.]\psi_1(x)$ (or $[\exists y.]\psi_1(y)$, respectively) is a formula in both $FO^x_{DL}$ and $FO^y_{DL}$, which can be considered as a special case of the form $\psi(x) \in FO^x_{DL}$ or $\psi(y) \in FO^y_{DL}$. Hence the proof of such cases where the context conditions are either of the form $[\exists x.]\psi_1(x)$ or $[\exists y.]\psi_1(y)$ are the same as the cases where the context conditions are of the form $\psi(x)$ (or $\psi(y)$, either one is fine) in Table 12. Moreover, let $O$, $O_1$ and $O_2$ be some constants. There are also cases for $a = A(O)$ (or $a = A(O_1, O_2)$, respectively) in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $R$, which can be proved similarly to the cases in Table 12 where $a = A$. There are also cases for $a = A(O_1, x)$ (or $a = (x, O_1)$, respectively) in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $R$, which can be proved similarly to the cases in Table 12 where $a = A(x)$. There are also cases for $a = A(O_1, y)$ (or $a = (y, O_1)$, respectively) in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $R$, which can be proved similarly to the cases in Table 12 where $a = A(y)$. There are also cases for $a = A(y, x)$ in $\phi_i^+$ and/or $\phi_j^-$ in the SSA of $R$, which can be proved similarly to the cases in Table 12 where $a = A(x, y)$. Notice that using unique name axioms for object constants, we can replace all (in)equalities between object constants with either *true* or *false* in the resulting formula that is equivalent to $\epsilon[\rho[W]]$ for any $\mathcal{L}^{C^2}_{sc}$ regressable formula $W$. Moreover, such deduction takes at most linear number of steps wrt the size of the resulting formula.

We first show that for any case in Table 12, $\epsilon[\phi_i^+(x, y, \alpha)]$ ($\epsilon[\phi_j^-(x, y, \alpha)]$, respectively) results in a formula that is equivalent to a conjunction of one formula in $FO^x_{DL}$ and another formula in $FO^y_{DL}$, i.e., $(\nu(x) \wedge \eta(y))$ for some $\nu(x) \in FO^x_{DL}$ and $\eta(y)) \in FO^y_{DL}$. Furthermore, we will see in the proof below for all cases in Table 12, the resulting formulas are in one of the four specific forms of $\nu(x) \wedge \eta(y)$: $\nu(x)$ (when $\eta(y)$ is *true*), $\eta(y)$ (when $\nu(x)$) is *true*), $\nu(x) \wedge y = O$ for some constant $O$, or $x = O \wedge \eta(y)$ for some constant $O$. We introduce these new notations $\nu(x)$ and $\eta(y)$ to show clearly that for each context condition, the regression of it is expressible as a conjunction of formulas with free variables $x$ and $y$ separated if there are any of them: $\nu(x)$ has no free occurrences of $y$ and $\eta(y)$ has no free occurrences of $x$. Subsequently, it is important that only these four syntactic forms actually occur.

From now on, without particular emphasis, all the cases we discuss below are the cases in Table 12. Moreover, for similar proofs, some details are skipped. Again, note that the operators $\rho$ and $\epsilon$ do not perform logical simplifications.

Here is one trivial sub-case: if the function name of $\alpha$ is not $A$, then in each of the aforementioned cases (1)–(16), (1')–(16') and in Note 2, $\epsilon$ of each formula ($\phi_i^+$ or $\phi_j^-$) equals *false*, which is still in $FO^x_{DL}$. Hence, below we only discuss the condition that $\alpha$ has the same function name (with the same number of arguments) as the given action function name in each case, and we let $O$, $O_1$ and $O_2$ be some constants.

(1) $a = A \wedge \psi(x)$

Assume that $\alpha = A$, then
$$\begin{aligned} & \epsilon[\alpha = A \wedge \psi(x)] \\ = \ & true \wedge \psi(x). \end{aligned}$$
Clearly, $true \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $\psi(y)$ and let $\eta(y)$ be *true*.

(1') $a = A \wedge \psi(y)$

Assume that $\alpha = A$, then
$$\begin{aligned} & \epsilon[\alpha = A \wedge \psi(y)] \\ = \ & true \wedge \psi(y). \end{aligned}$$
Clearly, $true \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be *true* and let $\eta(y)$ be $\psi(y)$.

(2) $a = A(x) \wedge \psi(x)$

Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\alpha = A(x) \wedge \psi(x)] \\ = \ & \epsilon[A(O) = A(x)] \wedge \psi(x) \\ = \ & (x = O) \wedge \psi(x). \end{aligned}$$
Clearly, $x = O \wedge \psi(x)$ is in $FO^x_{DL}$ and is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $x = O \wedge \psi(x)$ and let $\eta(y)$ be *true*.

(2') $a = A(x) \wedge \psi(y)$

Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\alpha = A(x) \wedge \psi(y)] \\ = \ & \epsilon[A(O) = A(x)] \wedge \psi(y) \\ = \ & (x = O) \wedge \psi(y). \end{aligned}$$
Clearly, $x = O \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $x = O$ and let $\eta(y)$ be $\psi(y)$.

(3) $\exists x(a = A(x)) \wedge \psi(x)$

Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\exists x(\alpha = A(x)) \wedge \psi(x)] \\ = \ & \epsilon[\exists x(A(O) = A(x))] \wedge \epsilon[\psi(x)] \\ = \ & \exists x(x = O) \wedge \psi(x). \end{aligned}$$
Clearly, $\exists x(x = O) \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $\psi(x)$ and let $\eta(y)$ be $\exists x(x = O)$.

(3') $\exists x(a = A(x)) \wedge \psi(y)$

Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\exists x(\alpha = A(x)) \wedge \psi(y)] \\ = \ & \epsilon[\exists x(A(O) = A(x))] \wedge \epsilon[\psi(y)] \\ = \ & \exists x(x = O) \wedge \psi(y). \end{aligned}$$
Clearly, $\exists x(x = O) \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be *true* and let $\eta(y)$ be $\exists x(x = O) \wedge \psi(y)$.

(4) $\exists x.a = A(x) \wedge \psi(x)$

Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\exists x.\alpha = A(x) \wedge \psi(x)] \\ = \ & \exists x.\epsilon[A(O) = A(x)] \wedge \epsilon[\psi(x)] \\ = \ & \exists x.x = O \wedge \psi(x). \end{aligned}$$
Clearly, $\exists x.x = O \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be *true* and let $\eta(y)$ be $\exists x.x = O \wedge \psi(x)$.

(4') $\exists x.a = A(x) \wedge \psi(y)$
Case (4') is equivalent to case (3'), because in case (4') the quantification range of $x$ is in fact only over $a = A(x)$.

(5) $a = A(y) \wedge \psi(x)$ (The proof is similar to that of case (2') above.)
Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\alpha = A(y) \wedge \psi(x)] \\ = \quad & y = O \wedge \psi(x). \end{aligned}$$
Clearly, $y = O \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $\psi(x)$ and let $\eta(y)$ be $y = O$.

(5') $a = A(y) \wedge \psi(y)$ (The proof is similar to that of case (2) above.)
Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\alpha = A(y) \wedge \psi(y)] \\ = \quad & y = O \wedge \psi(y). \end{aligned}$$
Clearly, $y = O \wedge \psi(y)$ is in $FO^{y}_{DL}$ and is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $y = O \wedge \psi(y)$ and let $\nu(x)$ be *true*.

(6) $\exists y(a = A(y)) \wedge \psi(x)$ (The proof is similar to that of case (3') above.)
Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\exists y(\alpha = A(y)) \wedge \psi(x)] \\ = \quad & \exists y(y = O) \wedge \psi(x). \end{aligned}$$
Clearly, $\exists y(y = O) \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be *true* and let $\nu(x)$ be $\exists y(y = O) \wedge \psi(x)$.

(6') $\exists y(a = A(y)) \wedge \psi(y)$ (The proof is similar to that of case (3) above.)
Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\exists y(\alpha = A(y)) \wedge \psi(y)] \\ = \quad & \exists y(y = O) \wedge \psi(y). \end{aligned}$$
Clearly, $\exists y(y = O) \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $\psi(y)$ and let $\nu(x)$ be $\exists y(y = O)$.

(7) $\exists y.a = A(y) \wedge \psi(x)$ (The proof is similar to that of case (3) above.)
Case (7) is equivalent to case (6), because in case (7) the quantification range of $y$ is in fact only over $a = A(y)$.

(7') $\exists y.a = A(y) \wedge \psi(y)$ (The proof is similar to that of case (4) above.)
Assume that $\alpha = A(O)$, then
$$\begin{aligned} & \epsilon[\exists y.\alpha = A(y) \wedge \psi(y)] \\ = \quad & \exists y.y = O \wedge \psi(y). \end{aligned}$$
Clearly, $\exists y.y = O \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be *true* and let $\nu(x)$ be $\exists y.y = O \wedge \psi(y)$.

(8) $a = A(x, y) \wedge \psi(x)$
Assume that $\alpha = A(O_1, O_2)$, then
$$\begin{aligned} & \epsilon[\alpha = A(x, y) \wedge \psi(x)] \\ = \quad & \epsilon[A(O_1, O_2) = A(x, y) \wedge \psi(x)] \\ = \quad & y = O_2 \wedge x = O_1 \wedge \psi(x). \end{aligned}$$
Clearly, $y = O_2 \wedge x = O_1 \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $y = O_2$ and let $\nu(x)$ be $x = O_1 \wedge \psi(x)$.

(8') $a = A(x, y) \wedge \psi(y)$
Assume that $\alpha = A(O_1, O_2)$, then
$$\begin{aligned} & \epsilon[\alpha = A(x, y) \wedge \psi(y)] \\ = \quad & \epsilon[A(O_1, O_2) = A(x, y) \wedge \psi(y)] \\ = \quad & x = O_1 \wedge y = O_2 \wedge \psi(y). \end{aligned}$$
Clearly, $x = O_1 \wedge y = O_2 \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $y = O_2 \wedge \psi(y)$ and let $\nu(x)$ be $x = O_1$.

(9) $\exists x(a = A(x, y)) \wedge \psi(x)$
Assume that $\alpha = A(O_1, O_2)$, then
$$\begin{aligned} & \epsilon[\exists x(\alpha = A(x, y)) \wedge \psi(x)] \\ = \quad & \epsilon[\exists x(A(O_1, O_2) = A(x, y)) \wedge \psi(x)] \\ = \quad & \exists x(x = O_1 \wedge y = O_2) \wedge \psi(x) \\ \equiv \quad & y = O_2 \wedge \exists x(x = O_1) \wedge \psi(x). \end{aligned}$$

Clearly, $y = O_2 \wedge \exists x(x = O_1) \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $y = O_2$ and let $\nu(x)$ be $\exists x(x = O_1) \wedge \psi(x)$.

(9') $\exists x(a = A(x, y)) \wedge \psi(y)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists x(\alpha = A(x, y)) \wedge \psi(y)] \\ = \; & (\exists x(x = O_1 \wedge y = O_2) \wedge \psi(y)) \\ \equiv \; & \exists x(x = O_1) \wedge y = O_2 \wedge \psi(y). \end{aligned}$$

Clearly, $\exists x(x = O_1) \wedge y = O_2 \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $y = O_2 \wedge \psi(y)$ and let $\nu(x)$ be $\exists x(x = O_1)$.

(10) $\exists x.a = A(x, y) \wedge \psi(x)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists x.\alpha = A(x, y) \wedge \psi(x)] \\ = \; & (\exists x.x = O_1 \wedge y = O_2 \wedge \psi(x)) \\ \equiv \; & y = O_2 \wedge \exists x.x = O_1 \wedge \psi(x). \end{aligned}$$

Clearly, $y = O_2 \wedge \exists x.x = O_1 \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be *true* and let $\eta(y)$ be $y = O_2 \wedge \exists x.x = O_1 \wedge \psi(x)$.

(10') $\exists x.a = A(x, y) \wedge \psi(y)$

Case (10') is equivalent to case (9'), because in case (10') the quantification range of $x$ is in fact only over $a = A(x, y)$.

(11) $\exists y(a = A(x, y)) \wedge \psi(x)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y(\alpha = A(x, y)) \wedge \psi(x)] \\ = \; & (\exists y(x = O_1 \wedge y = O_2) \wedge \psi(x)) \\ \equiv \; & \exists y(y = O_2) \wedge x = O_1 \wedge \psi(x). \end{aligned}$$

Clearly, $\exists y(y = O_2) \wedge x = O_1 \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be *true* and let $\nu(x)$ be $\exists y(y = O_2) \wedge x = O_1 \wedge \psi(x)$.

(11') $\exists y(a = A(x, y)) \wedge \psi(y)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y(\alpha = A(x, y)) \wedge \psi(y)] \\ = \; & (\exists y(x = O_1 \wedge y = O_2) \wedge \psi(y)) \\ \equiv \; & \exists y(y = O_2) \wedge x = O_1 \wedge \psi(y). \end{aligned}$$

Clearly, $\exists y(y = O_2) \wedge x = O_1 \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\eta(y)$ be $\psi(y)$ and let $\nu(x)$ be $\exists y(y = O_2) \wedge x = O_1$.

(12) $\exists y.a = A(x, y) \wedge \psi(x)$

Case (12) is equivalent to case (11), because in case (12) the quantification range of $y$ is in fact only over $a = A(x, y)$.

(12') $\exists y.a = A(x, y) \wedge \psi(y)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y.\alpha = A(x, y) \wedge \psi(y)] \\ = \; & (\exists y.x = O_1 \wedge y = O_2 \wedge \psi(y)) \\ \equiv \; & x = O_1 \wedge \exists y.y = O_2 \wedge \psi(y). \end{aligned}$$

Clearly, $x = O_1 \wedge \exists y.y = O_2 \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $x = O_1$ and let $\eta(y)$ be $\exists y.y = O_2 \wedge \wedge \psi(y)$.

(13) $\exists y(\exists x(a = A(x, y))) \wedge \psi(x)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y(\exists x(\alpha = A(x, y))) \wedge \psi(x)] \\ = \; & \exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(x) \\ = \; & \exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(x) \\ \equiv \; & \exists x(x = O_1) \wedge \exists y(y = O_2) \wedge \psi(x). \end{aligned}$$

Clearly, $\exists x(x = O_1) \wedge \exists y(y = O_2) \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $\exists y(y = O_2) \wedge \psi(x)$ and let $\eta(y)$ be $\exists x(x = O_1)$.

(13') $\exists y(\exists x(a = A(x, y))) \wedge \psi(y)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y(\exists x(\alpha = A(x, y))) \wedge \psi(y)] \\ = \; & \exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(y) \\ = \; & \exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(y). \end{aligned}$$

Clearly, $\exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be *true* and let $\eta(y)$ be $\exists y(\exists x(x = O_1 \wedge y = O_2)) \wedge \psi(y)$.

(14) $\exists y.\exists x(a = A(x, y)) \wedge \psi(x)$

Case (14) is equivalent to case (13), because in case (14) the quantification range of $y$ is in fact only over $a = A(x, y)$.

(14') $\exists y.\exists x(a = A(x, y)) \wedge \psi(y)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists y.\exists x(\alpha = A(x, y)) \wedge \psi(y)] \\ = \ & \exists y.\exists x(x = O_1 \wedge y = O_2) \wedge \psi(y) \\ \equiv \ & \exists x(x = O_1) \wedge \exists y.y = O_2 \wedge \psi(y). \end{aligned}$$

Clearly, $\exists x(x = O_1) \wedge \exists y.y = O_2 \wedge \psi(y)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $\exists y.y = O_2 \wedge \psi(y)$ and let $\eta(y)$ be $\exists x(x = O_1)$.

(15) $\exists x.\exists y(a = A(x, y)) \wedge \psi(x)$

Assume that $\alpha = A(O_1, O_2)$, then

$$\begin{aligned} & \epsilon[\exists x.\exists y(\alpha = A(x, y)) \wedge \psi(x)] \\ = \ & \exists x.\exists y(x = O_1 \wedge y = O_2) \wedge \psi(x) \\ \equiv \ & \exists y(y = O_2) \wedge \exists x.x = O_1 \wedge \psi(x). \end{aligned}$$

Clearly, $\exists y(y = O_2) \wedge \exists x.x = O_1 \wedge \psi(x)$ is of the form $\nu(x) \wedge \eta(y)$: let $\nu(x)$ be $\exists y(y = O_2)$ and let $\eta(y)$ be $\exists x.x = O_1 \wedge \psi(x)$.

(15') $\exists x.\exists y(a = A(x, y)) \wedge \psi(y)$

Case (15') is equivalent to case (13), because in case (15') the quantification range of $x$ is in fact only over $a = A(x, y)$.

(16) $\exists x.\exists y.a = A(x, y) \wedge \psi(x)$

Case (16) is equivalent to case (14'), because in case (16) the quantification range of $y$ is in fact only over $a = A(x, y)$.

(16') $\exists x.\exists y.a = A(x, y) \wedge \psi(y)$

Case (16') is equivalent to case (13), because in case (16') the quantification range of $x$ is in fact only over $a = A(x, y)$.

Notice that for each of the cases above, it takes no more than one step of logical deductions to find the equivalent formula of the form $\nu(x) \wedge \eta(y)$ such that $\nu(x) \in FO^x_{DL}$ and $\eta(y) \in FO^y_{DL}$, which takes a constant number of steps wrt the size of the resulting formula. Now, we complete the proof of Statement (2) for case (d.2). Recall that we consider the last remaining case when $W = \exists y.R(x, y, S) \wedge W_1(y)[S]$.

$$\begin{aligned} \epsilon[\rho[W]] &= \epsilon[\rho[\exists y.R(x, y, S) \wedge W_1(y)[S]]] \\ &= \exists y.\epsilon[\rho[R(x, y, S)]] \wedge \epsilon[\rho[W_1(y)[S]]] \\ &= \exists y.\epsilon[\rho[R(x, y, S)]] \wedge W'_1(y)[S_1] \end{aligned}$$

(by the induction hypothesis on $W_1(y)[S_1]$, let $\epsilon[\rho[W_1(y)[S]]] \equiv W'_1(y)[S_1]$ be a formula uniform in $S_1$ and $W'_1(y)$ is in $FO^y_{DL}$; moreover, $W'_1(y)$ can be found in no more than $c \cdot size(W'_1(y))$ steps for some constant $c$)

$$= \exists y.\epsilon \left[ \left( \bigvee_{i=1}^{m_+} \phi_i^+(x, y, \alpha) \vee R(x, y, S_1) \wedge \neg \left( \bigvee_{j=1}^{m_-} \phi_j^-(x, y, \alpha) \right) \right] \wedge W'_1(y)[S_1]$$

$$= \exists y.\Big(\bigvee_{i=1}^{m_+} \epsilon[\phi_i^+(x,y,\alpha)] \vee R(x,y,S_1) \wedge \neg\Big(\bigvee_{j=1}^{m_-} \epsilon[\phi_j^-(x,y,\alpha)]\Big)\Big) \wedge W_1'(y)[S_1]$$

$$\equiv \exists y.\Big(\bigvee_{i=1}^{m_+} (\nu_i^+(x) \wedge \eta_i^+(y)) \vee R(x,y,S_1) \wedge \bigwedge_{j=1}^{m_-} \neg(\nu_j^-(x) \wedge \eta_j^-(y))\Big) \wedge W_1'(y)[S_1]$$

(according the proof above, for each $i$, let $\epsilon[\phi_i^+(x,y,\alpha)] \equiv (\nu_i^+(x) \wedge \eta_i^+(y))$ be a formula uniform in $S_1$ for some $\nu_i^+(x) \in FO_{DL}^x$ and $\eta_i^+(y) \in FO_{DL}^y$; for each $j$, let $\epsilon[\phi_j^-(x,y,\alpha)] \equiv (\nu_j^-(x) \wedge \eta_j^-(y))$ be a formula uniform in $S_1$ for some $\nu_j^-(x) \in FO_{DL}^x$ and $\eta_j^-(y) \in FO_{DL}^y$)

$$\equiv \exists y.\Big(\bigvee_{i=1}^{m_+} (\nu_i^+(x) \wedge \eta_i^+(y) \wedge W_1'(y)) \vee R(x,y) \wedge W_1'(y) \wedge \bigwedge_{j=1}^{m_-} \neg(\nu_j^-(x) \wedge \eta_j^-(y))\Big)[S_1]$$

$$\equiv \Big\{\bigvee_{i=1}^{m_+} \nu_i^+(x) \wedge \exists y(\eta_i^+(y) \wedge W_1'(y))) \vee \exists y.$$
$$\bigwedge_{j=1}^{m_-} (\neg\nu_j^-(x) \vee \neg\eta_j^-(y)) \wedge R(x,y) \wedge W_1'(y)\Big\}[S_1]$$

$$\equiv \Big\{\bigvee_{i=1}^{m_+} \nu_i^+(x) \wedge \exists y(\eta_i^+(y) \wedge W_1'(y)) \vee$$
$$\bigvee_{k=1}^{2^{m_-}} \bigwedge_{j \in N_k} \neg\nu_j^-(x) \wedge \exists y(R(x,y) \wedge W_1'(y) \bigwedge_{l \in \overline{N_k}} \neg\eta_l^-(y))\Big\}[S_1]$$

(by applying distributive law over conjunction of $j$ and then minimizing the scope of $\exists y.$ for the negative conditions using the laws:
$\exists y.P_1(y) \vee P_2(y) \equiv \exists y(P_1(y)) \vee \exists y(P_2(y))$, and
$\exists y.P_1 \wedge P_2(y) \equiv P_1 \wedge \exists y(P_2(y))$ when $y$ is not free in $P_1$)

$$= W'[S_1], \tag{15}$$

where each $N_k$ ($k = 1..2^{m_-}$) enumerates a subset of indices $\{1, \cdots, m_-\}$, and $\overline{N_k} = \{1, \cdots, m_-\} - N_k$, i.e., $\overline{N_k}$ is the complement set of $N_k$. It is clear the formula on the RHS of Eq. 15 (denoted as $W'[S_1]$ above) is equivalent to $\epsilon[\rho[W]]$, is $\mathcal{L}_{sc}^{\mathcal{C}^2}$ regressable, and is in $FO_{DL}^x$ when $S_1$ is suppressed, and has no appearance of *Poss*. It is easy to see that to find it, it takes no more than $c \cdot size(W')$ for some integer $c$. Hence, Statement (2) is true for $W$. Moreover, according to Corollary 4, we have that $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[W'[S_1]]$. Then, by the induction hypothesis on formulas uniform in $S_1$, we have $\mathcal{R}[W][-S_0]$ will be equivalent to some formula in $FO_{DL}^x$.

It is very similar to prove that Statements (1) and (2) are true when $W[-S]$ is of the form $\forall y.R(x,y) \supset W_1(y)$, and details are omitted here.

Similarly, we can show that Statement (1) and Statement (2) are true when $W[-S]$ is in $FO^y_{DL}$ and is not atomic. Overall, we have proved the Statement (1).

Now, consider any $\mathcal{L}^{C^2}_{sc}$ regressable $W$ that is uniform in a ground situation $S$. When $W[-S]$ is in $FO^x_{DL}$, assume that we have found some $\Phi_W$ that is in $FO^x_{DL}$, such that $\mathcal{R}[W] \equiv \Phi_W[S_0]$. Below we will estimate an upper bound on the size of $\Phi_W$. Let $n = sitLength(S)$ ($n \in \mathbb{N}$ and $n \geq 0$), i.e., the number of action terms involved in $S$. Let $m = size(W)$ ($m \in \mathbb{N}$ and $m \geq 1$). Let function $f(m, n)$ be the size of $\Phi_W$, which is a non-decreasing function.

Firstly, it is straightforward that $f(1, 0) = 1$.

Secondly, when $m = 1$, $W$ is atomic, which is either *true*, or *false*, or $x = b$ for some constant $b$, or a situation-independent predicate, or a primitive dynamic concept. We now consider $n \geq 1$. Assume that $S = do(\alpha_n, S_1)$, and $sitLength(S_1) = n - 1$. According to the discussion above of "the base case of the induction on the structure of $W[-S]$" (i.e., cases (a–c)), for any $n \in \mathbb{N}$ and $n \geq 1$, $f(1, n) \leq f(3h, n - 1)$, where $h = \max(2, sizeSSA(\mathcal{D}))$ ($h$ is a constant number for the given $\mathcal{D}$). By Corollary 4, we have $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]]$, where $\epsilon[\rho[W]]$ is uniform in $S_1$ (no matter whether it is situation-independent or not), and is equivalent to some $\Phi_1 \in FO^x_{DL}$, whose size is no more than $3h$ (including all cases when $W$ is atomic). Moreover, the equivalent formula $\Phi_W[S_0]$ that we are looking for can be obtained by looking for the equivalent formula of $\mathcal{R}[\Phi_1[S_1]]$, whose size is no more than $f(3h, n - 1)$.

Thirdly, we consider any $m \geq 2$ and $n \in \mathbb{N}$. In fact, when $m \geq 2$, $W$ is not atomic. According to the definition of $FO^x_{DL}$, there are three sub-cases.

1. $W[-S]$ is of the form $\neg W_1$, or $\exists y.W_1(y)$, or $\forall y.W_1(y)$ where $W_1$ is in $FO^x_{DL}$, or $W_1(y)$ is in $FO^y_{DL}$. It is easy to see that $f(m, n) = f(m - 1, n) + 1$ according to the definition of the regression operator $\mathcal{R}$ and the way $FO^x_{DL}$ constructed. For example, $\mathcal{R}[\neg W_1[S]] = \neg\mathcal{R}[W_1[S]]$, if we find a formula $\Phi_1$ in $FO^x_{DL}$ such that $\Phi_1[S_0]$ is equivalent to $\mathcal{R}[W_1[S]]$, then $\Phi_W = \neg\Phi_1$ is the formula that we are looking for.
2. $W[-S]$ is of the form $W_1 \vee W_2$, or $W_1 \wedge W_2$ where $W_1$ and $W_2$ are in $FO^x_{DL}$. It is easy to see that $f(m, n) = f(size(W_1), n) + f(size(W_2), n) + 1$, which is no more than $2f(m - 1, n) + 1$, according to the definition of the regression operator $\mathcal{R}$ and the way $FO^x_{DL}$ constructed.
3. $W[-S]$ is of the form $\exists y.R(x, y) \wedge W_1(y)$, or $\forall y.R(x, y) \supset W_1(y)$, where $W_1(y)$ is in $FO^x_{DL}$. According to the definition of *size* in Section 5.3, we have $size(W_1(y)) = m - 3$. For instance, we consider the case when $W[-S]$ is of the form $\exists y.R(x, y) \wedge W_1(y)$, and it is similar for the case when $W[-S]$ is of the form $\forall y.R(x, y) \supset W_1(y)$. According to the definition of $\mathcal{R}$, we have

$$\begin{aligned}
\mathcal{R}[W] &= \mathcal{R}[\exists y.R(x, y)[S] \wedge W_1(y)[S]] \\
&= \exists y.\mathcal{R}[R(x, y)[S]] \wedge \mathcal{R}[W_1(y)[S]] \\
&= \begin{cases} \exists y.\mathcal{R}[R(x, y, S)] \wedge \mathcal{R}[W_1(y)[S]] & \text{if } R \text{ is a fluent,} \\ \exists y.R(x, y) \wedge \mathcal{R}[W_1(y)[S]] & \text{otherwise.} \end{cases}
\end{aligned} \tag{16}$$

Assume that $\mathcal{R}[W_1(y)[S]]$ with the initial situation $S_0$ suppressed is equivalent to some $\Phi_1(y) \in FO^y_{DL}$. When $R$ is situation-independent, $\exists y.R(x, y) \wedge \Phi_1(y)$

is the formula that we are looking for, whose size is $f(m-3,n)+3$. When $R$ is a dynamic role, we assume that its SSA is of the form Eq. 14, $S_i = do([\alpha_1, \cdots, \alpha_{n-i}], S_0)$ for any $i = 1..n-1$, and $S = do(\alpha_n, S_1)$. Since it is difficult to estimate the upper of the size of $R[\exists y.R(x,y,S) \wedge W_1(y)[S]]$ in this case recursively using Eq. 15, we will compute its regression result and then estimate its size. Note that we give the computation of the regression first and then provide the explanations for some major steps in the computation right after the equations.

$$
\begin{aligned}
\mathcal{R}[W] &= \exists y.\mathcal{R}[R(x,y,S)] \wedge \mathcal{R}[W_1(y)[S]] \\
&\equiv \exists y.\mathcal{R}[R(x,y,S)] \wedge \Phi_1(y)[S_0] \\
&\equiv \exists y.\left\{\bigvee_{i=1}^{m_+} \mathcal{R}[\phi_i^+(x,y,\alpha_n)] \vee \mathcal{R}[R(x,y,S_1)] \wedge \neg\left(\bigvee_{j=1}^{m_-} \mathcal{R}[\phi_j^-(x,y,\alpha_n)]\right)\right\} \\
&\quad \wedge \Phi_1(y)[S_0] \\
&\equiv \exists y.\left\{\bigvee_{i=1}^{m_+} \mathcal{R}[\epsilon[\rho[\phi_i^+(x,y,\alpha_n)]]] \vee \mathcal{R}[R(x,y,S_1)]\right. \\
&\qquad \left. \wedge \neg\left(\bigvee_{j=1}^{m_-} \mathcal{R}[\epsilon[\rho[\phi_j^-(x,y,\alpha_n)]]]\right)\right\} \wedge \Phi_1(y)[S_0] \\
&\quad \text{(use Corollary 4)} \\
&\equiv \exists y.\left\{\bigvee_{i=1}^{m_+} (\nu_{i,n}^+(x) \wedge \eta_{i,n}^+(y)) \vee \mathcal{R}[R(x,y,S_1)] \wedge \left(\bigwedge_{j=1}^{m_-} (\neg\nu_{j,n}^-(x) \vee \neg\eta_{j,n}^-(y))\right)\right\} \\
&\quad \wedge \Phi_1(y)[S_0] \\
&\quad \text{(use the result in Statement (2) for each } \phi_i^+(x,y,\alpha_n) \text{ and } \phi_j^-(x,y,\alpha_n), \\
&\quad \text{i.e., for each } i, \mathcal{R}[\epsilon[\rho[\phi_i^+(x,y,\alpha_n)]]] \equiv \nu_{i,n}^+(x) \wedge \eta_{i,n}^+(y) \\
&\quad \text{for some } \nu_{i,n}^+(x) \in FO_{DL}^x \text{ and some } \eta_{i,n}^+(y) \in FO_{DL}^y \\
&\quad \text{and for each } j, \mathcal{R}[\epsilon[\rho[\phi_j^-(x,y,\alpha_n)]]] \equiv \nu_{j,n}^-(x) \wedge \eta_{j,n}^-(y) \\
&\quad \text{for some } \nu_{j,n}^-(x) \in FO_{DL}^x \text{ and some } \eta_{j,n}^-(y) \in FO_{DL}^y) \\
&\equiv \exists y.\left\{\bigvee_{i=1}^{m_+} (\nu_{i,n}^+(x) \wedge \eta_{i,n}^+(y)) \vee \left(\bigvee_{i=1}^{m_+} \mathcal{R}[\phi_i^+(x,y,\alpha_{n-1})] \vee \mathcal{R}[R(x,y,S_2)] \wedge\right.\right. \\
&\qquad \left.\left. \neg\left(\bigvee_{j=1}^{m_-} \mathcal{R}[\phi_j^-(x,y,\alpha_{n-1})]\right)\right) \wedge \left(\bigwedge_{j=1}^{m_-} (\neg\nu_{j,n}^-(x) \vee \neg\eta_{j,n}^-(y))\right)\right\} \wedge \Phi_1(y)[S_0] \\
&\quad \text{(each } \nu_{i,n}^+(x), \eta_{i,n}^+(y), \nu_{j,n}^-(x), \eta_{j,n}^-(y) \text{ are situation-independent, and} \\
&\quad \text{apply one-step regression on } R(x,y,S_1), \text{ where } S_1 = do(\alpha_{n-1}, S_2))
\end{aligned}
$$

$$\equiv \exists y.\Big\{\bigvee_{i=1}^{m_+}(\nu_{i,n}^+(x)\wedge\eta_{i,n}^+(y))\vee\Big(\bigvee_{i=1}^{m_+}(\nu_{i,n-1}^+(x)\wedge\eta_{i,n-1}^+(y))\vee\mathcal{R}[R(x,y,S_2)]$$
$$\wedge\Big(\bigwedge_{j=1}^{m_-}(\neg\nu_{j,n}^-(x)\vee\neg\eta_{j,n}^-(y))\Big)\Big)\wedge\Big(\bigwedge_{j=1}^{m_-}(\neg\nu_{j,n}^-(x)\vee\neg\eta_{j,n}^-(y))\Big)\Big\}\wedge\Phi_1(y)[S_0]$$
(use the result in Statement (2) for each $\phi_i^+(x,y,\alpha_{n-1})$ and $\phi_j^-(x,y,\alpha_{n-1})$) (17)

$$\equiv\cdots\left(\text{let }\gamma_l^+=\bigvee_{i=1}^{m_+}\left(\nu_{i,l}^+(x)\wedge\eta_{i,l}^+(y)\right),\gamma_l^-=\bigwedge_{j=1}^{m_-}\left(\neg\nu_{j,l}^-(x)\vee\neg\eta_{j,l}^-(y)\right),l=1..n\right)$$
$$\equiv\exists y.\left\{\gamma_n^+\vee\left(\gamma_{n-1}^+\vee\left(\cdots\vee\left(\gamma_1^+\vee R(x,y,S_0)\wedge\gamma_1^-\right)\wedge\cdots\right)\wedge\gamma_{n-1}^-\right)\wedge\gamma_n^-\right\}\Phi_1(y)[S_0] \quad (18)$$

$$\equiv\left\{\exists y.(\gamma_n^+\vee\gamma_{n-1}^+\wedge\gamma_n^-\vee\cdots\vee\gamma_1^+\wedge\bigwedge_{i=2}^{n}\gamma_i^-\vee R(x,y)\wedge\bigwedge_{i=1}^{n}\gamma_i^-)\wedge\Phi_1(y)\right\}[S_0] \quad (19)$$

$$\equiv\cdots\text{(use distributive law to obtain a sort of DNF format)} \quad (20)$$

$$\equiv\left\{\exists y.\left(\bigvee_{i=1}^{u}\Phi_{S,i}(x,y)\right)\wedge\Phi_1(y)\right\}[S_0]\qquad\text{for some index } u \quad (21)$$

(each $\Phi_{S,i}(x,y)$ is a conjunction of some of the sub-formulas in the set $\{R(x,y)\}\cup\{\nu_{i,l}^+(x),\eta_{i,l}^+(x),\nu_{j,l}^-(y),\eta_{j,l}^-(y)\mid i=1..m_+,j=1..m_-,l=1..n\}$)

$$\equiv\left\{\bigvee_{i=1}^{u}(\exists y.\Phi_{S,i}(x,y)\wedge\Phi_1(y))\right\}[S_0] \quad (22)$$

$$\equiv\left\{\bigvee_{i=1}^{u}\Psi_{S,i}(x)\right\}[S_0] \quad (23)$$

where each $\nu_{i,l}^+(x)\wedge\eta_{i,l}^+(y)$ ($\nu_{j,l}^-(x)\wedge\eta_{j,l}^-(y)$, respectively) is equivalent to $\mathcal{R}[\epsilon[\rho[\phi_i^+(x,y,\alpha_k)]]]$ ($\mathcal{R}[\epsilon[\rho[\phi_j^-(x,y,\alpha_k)]]]$, respectively). Here, each $\nu_{i,l}^+(x)$ ($\nu_{j,l}^-(x)$, respectively) is in $FO_{DL}^x$ with at most one free variable $x$, and each $\eta_{i,l}^+(y)$ ($\eta_{j,l}^-(y)$, respectively) is in $FO_{DL}^y$ with at most one free variable $y$, according to the proof for the cases (1)–(16) and (1')–(16') in Table 12. Notice that in order to obtain an equivalent formula of $\mathcal{R}[W][-S_0]$ in $FO_{DL}^x$, we need to perform the following steps of deductions. First, we transform $\mathcal{R}[R(x,y,S)]$ in Step (17)

into a sort of disjunctive normal form (DNF) (from Step (17) to Step (21)) based on the assumption that each $\nu^+_{i,l}$, ($\eta^+_{i,l}(y)$, $\nu^-_{j,l}(x)$, $\eta^-_{j,l}(y)$, respectively) is "atomic", i.e., when each of these sub-formulas is considered as an atom, after using the distributive law, the resulting sub-formula $\bigvee_{i=1}^{u} \Phi_{S,i}(x, y)$ in Step 21 is a DNF formula. Since the resulting formula is too long, we omit the details and only provide one example of a sub-formula in Step (19) . For instance, we perform the distributive law over subformula $\gamma^+_{n-1} \wedge \gamma^-_n$:

$$\begin{aligned}
&\gamma^+_{n-1} \wedge \gamma^-_n \\
&\equiv \left(\bigvee_{i=1}^{m_+} (\nu^+_{i,n-1}(x) \wedge \eta^+_{i,n-1}(y))\right) \wedge \left(\bigwedge_{j=1}^{m_-} (\neg\nu^-_{j,n}(x) \vee \neg\eta^-_{j,n}(y))\right) \\
&\equiv \bigvee_{i=1..m_+, k=1..2^{m_-}} \nu^+_{i,n-1}(x) \wedge \eta^+_{i,n-1}(y) \wedge \bigwedge_{j \in N_k} \neg\nu^-_{j,n}(x) \wedge \bigwedge_{l \in \overline{N_k}} \neg\eta^-_{l,n}(y),
\end{aligned}$$

where $N_k \subseteq \{1, 2, \cdots, m_-\}$ $(k = 1..2^{m_-})$ enumerates all sub-sets of $\{1, 2, \cdots, m_-\}$, and $\overline{N_k} = \{1, 2, \cdots, m_-\} - N_k$, i.e., is the complement set of $N_k$. Next, we distribute $\Phi_1(y)[S_0]$ into the resulting DNF formula (from Step (21) to Step (22)). Finally, we push $\exists y$ inside into each conjunctive clause and minimize the scope of each quantifier $\exists y$ (from Step (21) to Step (23)). In Step (21), after using the commutative law of conjunctions, each $\Phi_{S,i}(x, y)$ is either of the form $\nu_{S,i}(x) \wedge \eta_{S,i}(y)$ or of the form $\nu_{S,i}(x) \wedge R(x, y) \wedge \eta_{S,i}(y)$ for some $\nu_{S,i}(x) \in FO^x_{DL}$ and some $\eta_{S,i}(y) \in FO^y_{DL}$ . From Step (22) to Step (23), there are two cases for each index $i$: if $\Phi_{S,i}(x, y)$ is of the form $\nu_{S,i}(x) \wedge \eta_{S,i}(y)$, then $\Psi_{S,i}(x)$ is $\nu_{S,i}(x) \wedge (\exists y.\eta_{S,i}(y) \wedge \Phi_1(y))$; else if $\Phi_{S,i}(x, y)$ is of the form $\nu_{S,i}(x) \wedge R(x, y) \wedge \eta_{S,i}(y)$, then $\Psi_{S,i}(x)$ is $\nu_{S,i}(x) \wedge (\exists y.R(x, y) \wedge \eta_{S,i}(y) \wedge \Phi_1(y))$. Hence, the resulting formula in Step (23), with $S_0$ suppressed, is in $FO^x_{DL}$, and we denote the formula (with $S_0$ suppressed) as $\Phi_W$.

Note that we did not use the result of Statement (2) to estimate the upper bound, because for each one-step regression, in order to get an equivalent formula in $FO_{DL}$ that is uniform in $S_1$, we had to distribute $W'_1(y)$ (see Eq. 15) at every inductive step, which will finally result in a formula uniform in $S_0$ with larger size.

Now we estimate the size of $\Phi_W$ when $R$ is a fluent according to the way it is constructed above. First, for any $n \geq 0$ and any situation $S$ where $sitLength(S) = n$, we estimate an upper bound on the size of the DNF formula (denoted as $g(n)$ below), which is equivalent to $\mathcal{R}[R(x, y, S)]$ and constructed specifically according to the above steps (17–21). Note that for each $i = 1..m_+$, $j = 1..m_-$ and $l = 1..n$, $size(\nu^+_{i,l}(x))$, ($size(\eta^+_{i,l}(y))$, $size(\nu^-_{j,l}(x))$ $size(\eta^-_{j,l}(y))$, respectively) is no more than $h + 2$ according to the above for the cases (1)–(16) and (1')–(16') in Table 12. Moreover, according to the definition of function $size()$ in Section 5.3, the logical constructors should also be counted. Also, for any $m_-$ and $m_+$ for any role $R$, we always have $m_- < h$ and $m_+ < h$ (recall that constant number $h = \max(2, sizeSSA(\mathcal{D}))$). According to Step 23, $f(m, n) \leq (f(m - 3, n) + 3)g(n)$, where $f(m - 3, n) = size(\Phi_1(y))$.

Overall, when $m \geq 2$, $f(m, n) \leq \max(f(m - 1, n) + 1, 2f(m - 1, n) + 1, f(m - 3, n) + 3, g(n)(f(m - 3, n) + 3)A)$.

Below, we show that $g(n) \leq c_1 2^{nh}$ for some constant $c_1 = (2(h+3) + (h+4)h^2 + 2)2^{2h}$. Moreover, we can perform a similar estimation for the case when $W[-S]$ is of the form $\forall y.R(x, y) \supset W_1(y)$.

$$
\begin{aligned}
g(n) &\leq 2(h+3)m_+ + (2^{m_-})m_+(2(h+3) + m_-(h+4)) \\
&\quad + (2^{m_-})^2 m_+(2(h+3) + 2m_-(h+4)) + \ldots + (2^{m_-})^{n-1} m_+(2(h+3) \\
&\quad + (n-1)m_-(h+4)) + (2^{m_-})^n(2 + nm_-(h+4)) \\
&= 2(h+3)m_+ \sum_{i=0}^{n-1}(2^{m_-})^i + (h+4)m_- \sum_{i=1}^{n} i(2^{m_-})^i + 2(2^{m_-})^n \\
&= 2(h+3)m_+ \sum_{i=0}^{n-1}(2^{m_-})^i + (h+4)m_+m_- \sum_{i=1}^{n} i(2^{m_-})^i + 2(2^{m_-})^n \\
&< 2(h+3)h \sum_{i=0}^{n-1}(2^h)^i + (h+4)h^2 \sum_{i=1}^{n} i(2^h)^i + 2(2^h)^n \\
&\leq 2(h+3)h(2^h)^n + (h+4)h^2 n(2^h)^{n+1} + 2(2^h)^n \\
&\leq 2(h+3)h(2^h)^n + (h+4)h^2(2^h)^{n+2} + 2(2^h)^n \\
&\leq (2(h+3) + (h+4)h^2 + 2)(2^h)^{n+2} \\
&\leq c_1 2^{nh} \quad \text{(let constant number } c_1 = (2(h+3) + (h+4)h^2 + 2)2^{2h}\text{)}.
\end{aligned}
$$

We can perform a similar estimation for the case when $W[-S]$ is in $FO^y_{DL}$. Overall, for $m \geq 1, n \geq 1$, the upper bound of $f(m, n)$ is

$$
\begin{aligned}
f(m,n) &\leq \max(f(m-1,n)+1, 2f(m-1,n)+1, \\
&\qquad f(m-3,n)+3, g(n)(f(m-3,n)+3)) \\
&\leq c_1 2^{hn}(f(m-1,n)+3) \\
&\leq c_1 2^{hn}(c_1 2^{hn}(f(m-2,n)+3)+3) \\
&\leq \cdots \\
&\leq c_1 2^{hn}(c_1 2^{hn}(\cdots(2^{hn}(f(1,n)+3)) + \cdots + 3) + 3) \\
&= (c_1 2^{hn})^{m-1} f(1,n) + 3\sum_{i=0}^{m-2}(c_1 2^{hn})^i \\
&\leq (c_1 2^{hn})^{m-1}(f(1,n)+3) \\
&\leq c_1^{m-1} 2^{hn(m-1)}(f(3h, n-1)+3) \\
&\leq c_1^{m-1} 2^{hn(m-1)}(c_1^{3h-1} 2^{hn(3h-1)}(f(1,n-1)+3)+3) \\
&\leq c_1^{m-1} 2^{hn(m-1)}(c_1^{3h-1} 2^{hn(3h-1)}(f(3h,n-2)+3)+3) \\
&\leq \cdots
\end{aligned}
$$

$$
\begin{aligned}
&\leq c_1^{m-1} 2^{hn(m-1)}((c_1^{3h-1} 2^{hn(3h-1)})^n f(1,0) + 3\sum_{i=0}^{n}(c_1^{3h-1} 2^{hn(3h-1)})^i) \\
&\leq c_1^{m-1} 2^{hn(m-1)}((c_1^{3h-1} 2^{hn(3h-1)})^n + 3(c_1^{3h-1} 2^{hn(3h-1)})^{n+1}) \\
&\leq 4c_1^{m-1} 2^{hn(m-1)}(c_1^{3h-1} 2^{hn(3h-1)})^{n+1} \\
&= 4c_1^{(n+1)(3h-1)+(m-1)} 2^{hn(m-1)+h(3h-1)n(n+1)} \\
&= 2^{c_2((n+1)(3h-1)+(m-1))+hn(m-1)+h(3h-1)n(n+1)+2}
\end{aligned}
$$

where constant numbers $c_2 = log_2 c_1$ and $c_1 = (2(h+3) + (h+4)h^2 + 2)2^{2h}$. When $n = 0$, $f(m,0) = 1$, which is clearly no more than $2^{c_2((n+1)(3h-1)+(m-1))+hn(m-1)+h(3h-1)n(n+1)+2}$ as well. Hence, we finally have

$$f(m,n) \in O(2^{hmn+3h^2n^2}), \text{ where constant } h = \max(2, sizeSSA(\mathcal{D})).$$

That is, the size of the equivalent formula of $\mathcal{R}[W]$ that we are looking for is no more than exponential in the size of the given formula $W$. □

## References


1. Artale, A., Franconi, E.: A survey of temporal extensions of description logics. Ann. Math. Artif. Intell. **30**(1–4), 171–210 (2001)
2. Baader, F., Lutz, C., Miliĉić, M., Sattler, U., Wolter, F.: A description logic based approach to reasoning about web services. In: Proceedings of the WWW 2005 Workshop on Web Service Semantics (WSS2005). Chiba City, Japan (2005)
3. Baader, F., Lutz, C., Miliĉić, M., Sattler, U., Wolter, F.: Integrating description logics and action formalisms: first results. In: Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05), pp. 572–577. Pittsburgh, PA, USA (2005)
4. Baader, F., Miličić, M., Lutz, C., Sattler, U., Wolter, F.: Integrating description logics and action formalisms for reasoning about web services. Technical Report LTCS-05-02, Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, Germany (2005). Available at http://lat.inf.tu-dresden.de/research/reports.html
5. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications, 2nd Edn. Cambridge University Press, Cambridge (2007)
6. Bacchus, F., Halpern, J., Levesque, H.: Reasoning about noisy sensors and effectors in the situation calculus. Artif. Intell. **111**, 171–208 (1999)
7. van Benthem, J.: Modal correspondence theory. Ph.D. thesis, Mathematisch Instituut & Instituut voor Grondslagenonderzoek, University of Amsterdam (see also van Benthem, J.: Correspondence theory. In: Gabbay, D.M. Guenthner, F. (eds.) Handbook of Philosophical Logic, 2nd edn., vol. 3, pp. 325–408. Kluwer Academic, Norwell, 1976)
8. van Benthem, J.: Modal Logic and Classical Logic. Bibliopolis, Naples (1985)
9. Berardi, D., Calvanese, D., de Giacomo, G., Lenzerini, M., Mecella, M.: e-service composition by description logics based reasoning. In: Calvanese, D., de Giacomo, G., Franconi, E. (eds.) Proceedings of the 2003 International Workshop in Description Logics (DL-2003). Rome, Italy (2003)
10. Blackburn, P., van Benthem, J.: Modal logic: a semantic perspective. In: Blackburn, P., van Benthem, J., Wolter, F. (eds.) Handbook of Modal Logic, pp. 1–84. Elsevier, Amsterdam (2007)
11. Börger, E., Grädel, E., Gurevich, Y.: The Classical Decision Problem. Perspectives in Mathematical Logic, Springer, ISBN 3-540-42324-9, 2nd printing, 2001 (1997)

12. Borgida, A.: On the relative expressiveness of description logics and predicate logics. Artif. Intell. **82**(1–2), 353–367 (1996)
13. Boutilier, C., Reiter, R., Soutchanski, M., Thrun, S.: Decision-theoretic, high-level robot programming in the situation calculus. In: Proceedings of the 17th National Conference on Artificial Intelligence (AAAI'00), pp. 355–362. Austin, Texas (2000)
14. Calvanese, D., Franconi, E., Haarslev, V., Lembo, D., Motik, B., Turhan, A.Y., Tessaris, S. (eds.): In: Proceedings of the 2007 International Workshop on Description Logics (DL2007), Brixen-Bressanone, near Bozen-Bolzano, Italy, 8–10 June 2007, CEUR Workshop Proceedings, vol. 250 (2007). http://ceur-ws.org/Vol-250/
15. Calvanese, D., de Giacomo, G., Lenzerini, M., Rosati, R.: Actions and programs over description logic ontologies. In: [14] (2007)
16. Castilho, M.A., Herzig, A., Varzinczak, I.J.: It depends on the context! a decidable logic of actions and plans based on a ternary dependence relation. In: Proceedings of the 9th International Workshop on Non-Monotonic Reasoning (NMR-02), pp. 343–348, Toulouse, France, 19–21 April 2002
17. Chang, L., Lin, F., Shi, Z.: A dynamic description logic for representation and reasoning about actions. In: Zhang, Z., Siekmann, J.H. (eds.) Knowledge Science, Engineering and Management, Second International Conference, KSEM 2007. Lecture Notes in Computer Science, vol. 4798, pp. 115–127, Springer, Melbourne, Australia, 28–30 November 2007
18. Chang, L., Shi, Z., Qiu, L., Lin, F.: Dynamic description logic: embracing actions into description logic. In: [14] (2007)
19. Demolombe, R.: Belief change: from situation calculus to modal logic. J. Appl. Non-Class. Log. **13**(2), 187–198 (2003)
20. Demolombe, R., Herzig, A., Varzinczak, I.J.: Regression in modal logic. J. Appl. Non-Class. Log. **13**(2), 165–185 (2003)
21. Drescher, C., Thielscher, M.: Integrating action calculi and description logics. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI-2007. Lecture Notes in Computer Science, vol. 4667, pp. 68–83. Springer, Berlin (2007)
22. Eiter, T., Erdem, E., Fink, M., Senko, J.: Updating action domain descriptions. In: Kaelbling, L.P., Saffiotti, A. (eds.) Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05), pp. 418–423. Professional Book Center (2005)
23. Finzi, A., Pirri, F., Reiter, R.: Open world planning in the situation calculus. In: Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00), pp. 754–760. Austin, Texas, USA (2000)
24. Fischer, M.J., Ladner, R.E.: Propositional dynamic logic of regular programs. J. Comput. Syst. Sci. **18**(2), 194–211 (1979)
25. Gabbay, D.: Expressive functional completeness in tense logic. In: Mönnich, U. (ed.) Aspects of Philosophical Logic: Some Logical Forays into Central Notions of Linguistics and Philosophy, "Synthese Library", vol. 147, pp. 91–117. Reidel (1981)
26. Gabbay, D.M., Shehtman, V.B.: Products of modal logics, part 1. Log. J. IGPL **6**(1), 73–146 (1998)
27. Gabbay, D.M., Shehtman, V.B.: Products of modal logics. Part 2: relativised quantifiers in classical logic. Log. J. IGPL **8**(2), 165–210 (2000)
28. Gabbay, D.M., Shehtman, V.B.: Products of modal logics. Part 3: products of modal and temporal logics. Stud. Log. **72**(2), 157–183 (2002)
29. de Giacomo, G., Lenzerini, M.: PDL-based framework for reasoning about actions. In: Gori, M., Soda, G. (eds.) Lecture Notes in Computer Science, vol. 992, pp. 103–114. AIIA, Springer, Berlin (1995)
30. de Giacomo, G., Iocchi, L., Nardi, D., Rosati, R.: A theory and implementation of cognitive mobile robots. J. Log. Comput. **9**(5), 759–785 (1999)
31. de Giacomo, G., Lenzerini, M., Poggi, A., Posati, R.: On the update of description logic ontologies at the instance level. In: Proceedings of the 21st National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference (AAAI-06), pp. 1271–1276. AAAI Press, Boston, US (2006)
32. Gil, Y.: Description logics and planning. AI Mag. **26**(2), 73–84 (2005)
33. Grädel, E., Kolaitis, P.G., Vardi, M.Y.: On the decision problem for two-variable first-order logic. Bull. Symb. Log. **3**(1), 53–69 (1997)
34. Grädel, E., Otto, M., Rosen, E.: Two-variable logic with counting is decidable. In: Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science (LICS'97), pp. 306–317. Warsaw, Poland (1997)

35. Grüninger, M.: Ontology of the process specification language. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 575–592. Springer, Berlin (2004)
36. Grüninger, M., Menzel, C.: The process specification language (PSL): theory and applications. AI Mag. **24**(3), 63–74 (2003)
37. Gu, Y., Soutchanski, M.: Decidable reasoning in a modified situation calculus. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07), pp. 1891–1897. Hyderabad, India (2007). http://www.cs.ryerson.ca/~mes/publications/DecidableSitcalc_ijcai07.pdf
38. Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. MIT, Cambridge (2000)
39. Hemaspaandra, E.: The price of universality. Notre Dame J. Form. Log. **37**(2), 174–203 (1996)
40. Henkin, L.: Logical systems containing only a finite number of symbols. Tech. rep., Département de Mathématiques, Université de Montréal, Les Presses de l'Université de Montréal (1967)
41. Herzig, A., Varzinczak, I.J.: Metatheory of actions: beyond consistency. Artif. Intell. **171**(16–17), 951–984 (2007)
42. Herzig, A., Perrussel, L., Varzinczak, I.J.: Elaborating domain descriptions. In: Brewka, G. (ed.) 17th European Conference on Artificial Intelligence (ECAI-06). Frontiers in Artificial Intelligence and Applications, vol. 141, pp. 397–401, IOS Press, Riva del Garda, Italy, 29 August–1 September 2006
43. Horrocks, I., Sattler, U.: Ontology reasoning in the SHOQ(D) description logic. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), pp. 199–204. Morgan Kaufmann, San Francisco (2001)
44. Horrocks, I., Patel-Schneider, P., van Harmelen, F.: From SHIQ and RDF to OWL: the making of a web ontology language. J. Web Semant. **1**(1), 7–26 (2003)
45. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: Doherty, P., Mylopoulos, J., Welty, C.A. (eds.) Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR-06), pp. 57–67. AAAI, Menlo Park (2006)
46. Hustadt, U., de Nivelle, H., Schmidt, R.A.: Resolution-based methods for modal logics. Log. J. IGPL **8**(3), 265–292 (2000)
47. Hustadt, U., Schmidt, R.A., Georgieva, L.: A survey of decidable first-order fragments and description logics. J. Relat. Methods Comput. Sci. **1**, 251–276 (2004)
48. Kakas, A.C., Michael, L., Miller, R.: Modular-$\mathcal{E}$: an elaboration tolerant approach to the ramification and qualification problems. In: Baral, C., Greco, G., Leone, N., Terracina G. (eds.) Proceedings of the 8th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-05), Springer. Lecture Notes in Computer Science, vol. 3662, pp. 211–226, Diamante, Italy, 5–8 September 2005
49. Kemke, C.: A formal theory for describing action concepts in terminological knowledge bases. In: Xiang Y., Chaib-draa, B. (eds.) Advances in Artificial Intelligence: 16th Conference of the Canadian Society for Computational Studies of Intelligence. Lecture Notes in Computer Science, vol. 2671, pp. 458–465, 11–13 June 2003, Springer, Halifax (2003)
50. Kurucz, A.: Combining modal logics. In: Blackburn, P., van Benthem, J., Wolter, F. (eds.) Handbook of Modal Logic, pp. 869–924. Elsevier, Amsterdam (2007)
51. Levesque, H., Reiter, R., Lespérance, Y., Lin, F., Scherl, R.: GOLOG: a logic programming language for dynamic domains. J. Log. Program. **31**, 59–84 (1997)
52. Lin, F.: Discovering state invariants. In: Dubois, D., Welty, C.A., Williams, M.A. (eds.), pp. 536–544. KR, AAAI, Menlo Park (2004)
53. Lin, F.: Situation calculus. In: van Harmelen, F., Lifschitz, V., Porter, B. (eds.) Handbook of Knowledge Representation, pp. 649–669. Elsevier, Amsterdam (2008)
54. Lin, F., Reiter, R.: State constraints revisited. J. Log. Comput. **4**(5), 655–678 (1994)
55. Lin, F., Reiter, R.: How to progress a database. Artif. Intell. **92**, 131–167 (1997)
56. Liu, H., Lutz, C., Miličić, M., Wolter, F.: Reasoning about actions using description logics with general TBoxes. In: Fisher, M., van der, Hoek, W., Konev, B., Lisitsa, A. (eds.) Logics in Artificial Intelligence, 10th European Conference. Lecture Notes in Computer Science, vol. 4160, pp. 266–279. JELIA, Springer, Berlin (2006)
57. Liu, H., Lutz, C., Miličić, M., Wolter, F.: Updating description logic ABoxes. In: Doherty, P., Mylopoulos, J., Welty, C. (eds.) Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR-06), pp 46–56. AAAI, Menlo Park (2006)
58. Liu, Y., Lakemeyer, G.: On first-order definability and computability of progression for local-effect actions and beyond. In: Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09), pp. 860–866. Pasadena, USA (2009)

59. Liu, Y., Levesque, H.J.: Tractable reasoning with incomplete first-order knowledge in dynamic systems with context-dependent actions. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), pp. 522–527. Edinburgh, Scotland (2005)
60. Lutz, C., Sattler, U.: The complexity of reasoning with Boolean modal logics. In: Wolter, F., Wansing, H., de Rijke, M., Zakharyaschev, M. (eds.) Advances in Modal Logic, vol. 3, pp. 329–348. World Scientific, Singapore (2000)
61. Lutz, C, Sattler, U., Wolter, F.: Description logics and the two-variable fragment. In: McGuiness, D., Pater-Schneider, P., Goble, C., Möller, R. (eds.) Proceedings of the 2001 International Workshop in Description Logics (DL-2001), pp. 66–75. Stanford, California, USA (2001)
62. McCain, N., Turner, H.: A causal theory of ramifications and qualifications. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95), pp. 1978–1984, 20–25 August 1995, Montréal, Québec, Canada (1995)
63. McCarthy, J.: Programs with common sense. In: Mechanisation of Thought Processes, Proceedings of the Symposium of the National Physics Laboratory, Her Majesty's Stationery Office. Reprinted in [66], pp. 77–84. London, U.K. (1959)
64. McCarthy, J.: Situations, actions and causal laws. Memo 2, Stanford University, Department of Computer Science, reprinted in: "Semantic Information Processing" (M. Minsky, ed.), pp. 410–417. MIT, Cambridge (1963)
65. McCarthy, J.: Applications of circumscription to formalizing common sense knowledge. Artif. Intell. **28**, 89–116 (1986)
66. McCarthy, J.: Formalization of common sense. Ablex, Norwood (papers by John McCarthy edited by V. Lifschitz, 1990)
67. McCarthy, J.: Actions and other events in situation calculus. In: Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR2002), pp. 615–628. Morgan Kaufmann Publishers, Toulouse (2002). Available at http://www-formal.stanford.edu/jmc/sitcalc.html
68. McCarthy, J., Hayes, P.: Some philosophical problems from the standpoint of artificial intelligence. In: Meltzer, B., Michie, D. (eds.) Machine Intelligence, vol. 4, pp. 463–502. Edinburgh University Press, Edinburgh, Reprinted in [66] (1969)
69. McIlraith, S., Son, T.: Adapting Golog for composition of semantic web services. In: Fensel, D., Giunchiglia, F., McGuinness, D., Williams, M.A. (eds.) Proceedings of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002), pp. 482–493, Morgan Kaufmann, Toulouse, 22–25 April 2002
70. McIlraith, S.A.: Integrating actions and state constraints: a closed-form solution to the ramification problem (sometimes). Artif. Intell. **116**(1–2), 87–121 (2000)
71. Miličić, M.: Complexity of planning in action formalisms based on description logics. In: Dershowitz, N., Voronkov, A. (eds.) Proceedings of the 14th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2007). Lecture Notes in Computer Science, vol. 4790, pp. 408–422. Springer, Berlin (2007)
72. Morgenstern, L., Riecken, D.: SNAP: An action-based ontology for e-commerce reasoning. In: Formal Ontologies Meet Industry, Proceedings of the 1st International Workshop FOMI 2005. Verona, Italy (2005)
73. Narayanan, S., McIlraith S.: Analysis and simulation of web services. Comput. Networks **42**, 675–693 (2003)
74. de Nivelle, H., Pratt-Hartmann, I.: A resolution-based decision procedure for the two-variable fragment with equality. In: R Goré, A.L., Nipkow, T. (eds.) IJCAR'01: Proceedings of the First International Joint Conference on Automated Reasoning. Lecture Notes in Artificial Intelligence, vol. 2083, pp. 211–225. Springer, London (2001)
75. Ohlbach, H.J., Nonnengart, A., de Rijke, M., Gabbay, D.M.: Encoding two-valued nonclassical logics in classical logic. In: Handbook of automated reasoning, pp. 1403–1486. Elsevier, Amsterdam (2001)
76. Pacholski, L., Szwast, W., Tendera, L.: Complexity of two-variable logic with counting. In: Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science (LICS-97), pp. 318–327. A journal version: SIAM J. Comput. **29**(4), 1083–1117 (1999). Warsaw, Poland (1997)
77. Pacholski, L., Szwast, W., Tendera, L.: Complexity results for first-order two-variable logic with counting. SIAM J. Comput. **29**(4), 1083–1117 (2000)
78. Pirri, F., Reiter, R.: Some contributions to the metatheory of the situation calculus. J. ACM **46**(3), 325–364 (1999)

79. Pratt, V.R.: A practical decision method for propositional dynamic logic: preliminary report. In: Proceedings of the 10th Annual ACM Symposium on Theory of Computing, pp. 326–337. ACM, San Diego (1978)
80. Pratt-Hartmann, I.: Complexity of the two-variable fragment with counting quantifiers. J. Logic, Lang. Inf. **14**(3), 369–395 (2005). doi:10.1007/s10849-005-5791-1
81. Prendinger, H., Schurz, G.: Reasoning about action and change: a dynamic logic approach. J. Logic, Lang. Inf. **5**(2), 209–245 (1996)
82. Reiter, R.: Knowledge in Action: Logical Foundations for Describing and Implementing Dynamical Systems. MIT, Cambridge (2001)
83. Sandewall, E.: Features and Fluents: The Representation of Knowledge about Dynamical Systems. Oxford University Press, Oxford (1994)
84. Schaerf, A.: Reasoning with individuals in concept languages. Data Knowl. Eng. **13**(2), 141–176 (1994)
85. Schiffel, S., Thielscher M.: Reconciling situation calculus and fluent calculus. In: 21st National Conference on Artificial Intelligence (AAAI-2006), pp. 287–292. AAAI, Boston (2006)
86. Schild, K.: A correspondence theory for terminological logics: preliminary report. In: Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91), pp. 466–471. Morgan Kaufmann, Sydney (1991)
87. Schmidt, R.A., Tishkovsky, D.: Deciding ALBO with tableau. In: [14] (2007)
88. Schmidt, R.A., Tishkovsky, D.: A general tableau method for deciding description logics, modal logics and related first-order fragments. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) Automated Reasoning, 4th International Joint Conference (IJCAR-08). Lecture Notes in Computer Science, vol. 5195, pp. 194–209. Springer, Sydney, 12–15 August 2008
89. Schmidt-Schaubß, M., Smolka, G.: Attributive concept descriptions with complements. Artif. Intell. **48**(1), 1–26 (1991)
90. Shanahan, M.: Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia. MIT, Cambridge (1997)
91. Spaan, E.: Complexity of modal logics. Ph.D. thesis, Department of Mathematics and Computer Science, University of Amsterdam (1993)
92. Thielscher, M.: Challenges for Action Theories. Lecture Notes in Computer Science, vol. 1775. Springer, Berlin (2000)
93. Tobies, S.: A NExpTime-complete description logic strictly contained in $C^2$. In: Flum, J, Rodríguez-Artalejo M (eds.) Computer Science Logic, 13th International Workshop (CSL-99). Lecture Notes in Computer Science, vol. 1683, pp. 292–306. Springer, Berlin (1999)
94. Tobies, S.: The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. J. Artif. Intell. Res. **12**, 2000 (2000)
95. Tobies, S.: Complexity results and practical algorithms for logics in knowledge representation. Ph.D. thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany (2001)
96. Varzinczak, I.J.: Action theory contraction and minimal change. In: Brewka, G., Lang, J. (eds.) Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference (KR-2008), pp. 651–661. AAAI, Menlo Park (2008)
97. Vassos, S., Lakemeyer, G., Levesque, H.J.: First-order strong progression for local-effect basic action theories. In: Proceedings of 11th International Conference on Principles of Knowledge Representation and Reasoning (KR-08), pp. 662–672, Sydney, Australia, 16–19 September 2008
98. Winslett, M.S.: Updating Logical Databases. Academic, San Diego (1990)
99. Wolter, F., Zakharyaschev, M.: Dynamic description logics. In: Zakharyaschev, M., Segerberg, K., de Rijke, M., Wansing, H. (eds.) Advances in Modal Logic, vol. 2, pp. 431–446. CSLI, Stanford (1998)
100. Zolin, E.: Description logic complexity navigator (2007). Available at http://www.cs.man.ac.uk/~ezolin/dl/