A Logic For Decidable Reasoning About Actions [*]

Yilan Gu
Department of Computer Science,
University of Toronto,
10 King's College Road,
Toronto, ON, M5S 3G4, Canada
yilan@cs.toronto.edu

Mikhail Soutchanski[†]
Department of Computer Science,
Ryerson University,
245 Church Street, ENG281,
Toronto, ON, M5B 2K3, Canada
Tel.: +1-416-9795000, ext 7954
Fax: +1-416- 979-5064
mes@scs.ryerson.ca

October 6, 2008

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 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 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.

[*] This research has been partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and by the Ryerson University.

[†] A corresponding author.

Contents

“*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: “[50] 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 [51] and elaborated in [55]. The name situation calculus was first used in [55] but wasn't defined there. [52] proposed to solve the frame and qualification problems by circumscription, but the proposed solution to the frame problem was incorrect. [76] and [70] describe several situation calculus formalisms and give references” (see the footnote 4 in [54]).

In this paper we would like to consider the SC from [70] 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) [31, 30]. It is used to provide a well-defined semantics for Web services and a foundation for a high-level programming language Golog [41, 56, 61, 7]. However, because the situation calculus is formulated in a general predicate logic, reasoning about effects of sequences of actions is undecidable (unless some 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 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) [28, 64], we demonstrate that by reducing reasoning about effects of actions to reasoning in this fragment, one can always guarantee decidability 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).

The second motivation for our paper comes from description logics. Description Logics (DLs) [4] 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) [36] 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 [65, 68]) and offer considerable expressive power going far beyond propositional logic, while ensuring that reasoning is decidable [9]. DLs have been mostly used to describe static knowledge bases. 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 [25], 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 [2], another proposal 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. 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 5.2. In Section 5.3, 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 $\mathcal{L}_{sc}^{C^2}$. In Section 6, we consider an example that illustrates potential applications to Semantic Web Services. 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

[1] Reiter [70] uses the notation $s' \sqsubseteq s$, but we use $s' \preceq s$ to avoid confusion with the inclusion relation $\sqsubset$ that is commonly used in description logic literature. In this paper, we use $\sqsubset$ to denote the inclusion relation between concepts or roles.

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 and a term s of sort *situation*, we say W is a formula *uniform* in s iff it does not mention the predicates $Poss$, 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 [70]). 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 ϕ. 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 [70]: action precondition axioms $\mathcal{D}_{ap}$, successor state axioms (SSAs) $\mathcal{D}_{ss}$, initial theory $\mathcal{D}_{S_0}$, unique name axioms for actions $\mathcal{D}_{una}$, domain independent foundational axioms for situations Σ.

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

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

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

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

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 α has a positive effect on fluent F, if F becomes true in the situation resulting from executing this action. Similarly, α 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 [70], Section 3.2.4 and Section 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

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

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 are any.

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

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

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.[2]

Foundational axioms for situations Σ: 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 [70] 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 ([70]). A recursive definition of the regression operator $\mathcal{R}$ on any *regressable formula* W is given in [70]. 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 α_i is of sort action; (2) for every atom of the form $Poss(\alpha, \sigma)$ in W, α 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 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

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

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 [66] 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 Logics

In this section we review a few popular expressive description logics [4] and related fragments of FOL [9].

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 $\geqslant nR.C$ 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 $C_1 \sqcup C_2$, $C_1 \sqsubseteq C_2$, $\leqslant nR.C$, $\exists\, R.\, C$, $\forall\, R.\, C$ (and other complex concepts) can be easily defined. For example,

$$\begin{array}{ll}
C_1 \sqcup C_2 \overset{def}{=} \neg(\neg C_1 \sqcap \neg C_2) & \leqslant nR.C \overset{def}{=} \neg(\geqslant(n+1)R.C) \\
C_1 \sqsubseteq C_2 \overset{def}{=} \neg C_1 \sqcup C_2 & \exists^{=n}\, R.\, C \overset{def}{=} (\leqslant nR.C) \sqcap (\geqslant nR.C) \\
\exists R.C \overset{def}{=} \ \geqslant 1R.C & \top \overset{def}{=} AC \sqcup \neg AC \text{ for some } AC \in N_C \\
\forall R.C \overset{def}{=} \neg\exists R.\neg C & \bot \overset{def}{=} AC \sqcap \neg AC \text{ for some } AC \in N_C
\end{array}$$

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 $C_1 \equiv C_2$.[3] 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 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 [85]. For example, it has been shown that the complexity of solving these two problems is PSPACE-complete in $\mathcal{ALC}$, in $\mathcal{ALCO}$ and in $\mathcal{ALCQO}$ [75, 81, 3], but is EXPTIME-complete in $\mathcal{ALCQIO}$ [80, 81]. However, it is also known that the complexity for these two problems is EXPTIME-complete in $\mathcal{ALC}(U)$ (i.e., $\mathcal{ALC}$ plus the universal rule) [78, 34, 48] and in $\mathcal{SHOQ}$ [35]. 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 $\mathcal{ALCQO}(U)$.

3.2 C^2 and Its Relationship to Description Logics

The *two-variable FOL* FO^2 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

[3] Sometimes, *general inclusion axioms* of the form $C_1 \sqsubseteq C_2$ are also allowed, where C_1, C_2 are complex concepts.

can be reused arbitrarily often. The *two-variable FOL with counting* C^2 extends FO^2 by allowing FO counting quantifiers $\exists^{\geq m}$ and $\exists^{\leq m}$ for all $m \geq 1$ [28, 64, 65, 38, 68]. Because the semantics of 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 $\mathcal{K}$ [72]. For this reason, it is important to note that the *standard translation* from a basic modal logic to FOL is proposed in [5]. Later, it was realized that a basic modal logic can be translated to FO^2 [20, 6]. The standard translation and other results in modal logic are extensively discussed in [8] and in [63].

Now we consider some relationships between C^2 and DLs. In [9], an expressive description logic $\mathcal{B}$ is defined[4], 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 [9]. Moreover, in [9], 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 [9], 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 [49], another DL is considered[5] 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. [29] and Pacholski et. al. [64] show that the satisfiability problem for C^2 is decidable and recently in [68] 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.[6] See additional background on DLs and discussion of connections between DLs with C^2 in [4, 9].

[4] In [9], the language $\mathcal{B}$ is denoted as $\mathcal{DL} - \{\textbf{trans}, \text{compose}\}$, 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.

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

[6] In [4], 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.

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 additional situation argument).[7] 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., actions, situations and objects) 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 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 *basic action theory* (BAT) $\mathcal{D}$ using the following seven groups of axioms in $\mathcal{L}_{sc}^{C^2}$: $\mathcal{D} = \Sigma \cup \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_T \cup \mathcal{D}_R \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}$. 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.[8]

Action precondition axioms $\mathcal{D}_{ap}$: For each action A in $\mathcal{L}_{sc}^{C^2}$, there is one axiom of the form $Poss(A, s) \equiv \Pi_A[s]$ (or $Poss(A(x), s) \equiv \Pi_A(x)[s]$, or $Poss(A(x, y), s) \equiv \Pi_A(x, y)[s]$, respectively), if A is an action constant (or unary, or binary action term, respectively), where Π_A (or $\Pi_A(x)$, or $\Pi_A(x, y)$, respectively) is a C^2 formula with no free variables (or with at most x, or with at most x, y as the only free variables,

[7] 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.

[8] 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.

respectively). This set of axioms characterizes the preconditions of all actions.

Successor state axioms $\mathcal{D}_{ss}$: Let variable vector $\vec{x}$ be x, or y, or $\langle x, y\rangle$. An SSA is specified for each primitive dynamic concept that is not defined in TBox (see below) and each dynamic role $F(\vec{x}, do(a, s))$. According to the detailed syntactic form of the SSAs Eq. (3), without loss of generality, we can assume that the axiom has the form

$$\begin{aligned} F(\vec{x}, do(a, s)) \ \equiv \ & \bigvee_{i=1}^{m_+} [\exists x][\exists y](a{=}PosAct_i(\vec{t}_{(i,+)}) \wedge \psi_i^+(\vec{x}_{(i,+)})[s]) \vee \\ & F(\vec{x}, s) \wedge \neg(\bigvee_{j=1}^{m_-} [\exists x][\exists y](a{=}NegAct_j(\vec{t}_{(j,-)}) \wedge \psi_j^-(\vec{x}_{(j,-)})[s])). \end{aligned} \qquad (4)$$

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) is equal to 0, the corresponding disjunctive sub-formula is equivalent to $false$.

Acyclic TBox axioms $\mathcal{D}_T$: Similar to the TBox axioms in DL, we may define new concepts using TBox axioms. Any group of TBox axioms $\mathcal{D}_T$ may include two sub-classes: 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

$$G(x) \ \equiv \ \phi_G(x), \qquad (5)$$

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

$$G(x, s) \ \equiv \ \phi_G(x)[s],$$

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 LHS 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 DL, we may also specify a group of axioms, called RBox axioms below, to support a role taxonomy. Each role inclusion axiom is represented as

$$R_1(x, y)[s] \ \supset \ R_2(x, y)[s],$$

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 $(\mathcal{D} - \mathcal{D}_R) \models \forall s. R_1(x, y)[s] \supset R_2(x, y)[s]$. This is the common approach to state constraints, e.g., it was taken in [70]. 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 [57], 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 [43, 42]. 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 (assuming that we suppress the only situation term S_0 in all fluents). 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 Σ 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.

5 Reasoning about Actions using Regression

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

5.1 Modified Regression with Lazy Unfolding

Given a formula W of $\mathcal{L}_{sc}^{C^2}$ in the domain $\mathcal{D}$, the definition of W being regressable (called $\mathcal{L}_{sc}^{C^2}$ *regressable* below) is slightly different from the definition of W being regressable in $\mathcal{L}_{sc}$ (see [70]).

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

(1) *Each term of sort situation in W is ground.*

(2) *Other than the action terms occurred in the predicate $Poss$, 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, α has the syntactic form $A(t_1,\cdots,t_n)$ for some n-ary function symbol A of $\mathcal{L}_{sc}^{C^2}$, where $n \leq 2$. Moreover, each t_i is either variable x, variable y or some constant O if there is any.*

(4) *Each 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 α_i is of sort action.*

(5) *W does not quantify over situations.*

(6) *W does not mention the relation symbols "$\prec$" or "$=$" between terms of sort situation.*

The requirements in conditions (2) and (3) are obvious, because our language is restricted to $\mathcal{L}_{sc}^{C^2}$. The intuition for adding condition (1) is as follows. Consider the following counter example.

Example 1 Assume $\mathcal{D}$ is a BAT of $\mathcal{L}_{sc}^{C^2}$, which includes an SSA

$$F(x, do(a, s)) \equiv a = A(x) \wedge (\exists y.G(x, y, s)) \vee F(x, s) \tag{6}$$

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

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

of $\mathcal{L}_{sc}^{C^2}$. 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. (6) so that it is different from any variables appearing in W_0. That is, the one step regression on F using Eq. (6) should be

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

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. (6) to assure the regressed formula is still of language $\mathcal{L}_{sc}^{C^2}$, then the one step regression on F using Eq. (6) without renaming will result in the following formula:

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

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 1, we add (1) to Def. 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 regression operator $\mathcal{R}$ 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 [70] with the *lazy unfolding technique* (see [4]) 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 [70], 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 [70] has the assumption that the quantified variables on the 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 assures 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 for each $\mathcal{L}_{sc}^{C^2}$ regressable formula. The possibility of reusing variables is guaranteed by the general format of the SSAs given in the previous section and the additional conditions (1-3) in Def. 1.

The complete formal definition of our $\mathcal{R}$ is as follows,[9] where σ denotes the term of sort *situation*, and α denotes the term of sort *action*. Note that below, if $\Phi(\vec{x})$ represents a formula Φ with free variables among a variable vector $\vec{x}$ at most, then for any vector of terms $\vec{t}$ such that $|\vec{t}| = |\vec{x}|$, $\Phi(\vec{t}\,)$ represents the

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

resulting formula obtained by substituting each x_i in vector $\vec{x}$ with t_i in vector $\vec{t}$ if x_i occurs in Φ. For example, in particular, if $\Phi(\vec{x})$ in fact has no free variables, then no substitution happens and $\Phi(\vec{t}\ )$ is the same as $\Phi(\vec{x})$.

- 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

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

- 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 condition (2,3) in Def. 1, each term in $\vec{t}$ can only be a variable x, y or some constant if any. Then,

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

where O and O_1 are constants and $\widetilde{\phi}$ denotes a *dual formula* for formula ϕ 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 ϕ, 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 (1)–(3) of $\mathcal{L}_{sc}^{C^2}$ regressable formula, 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 σ, 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. Then, we use the lazy unfolding technique as follows:

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

c. If W is a primitive dynamic concept (a dynamic role, respectively) 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 α of sort *action* and ground term σ 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. (4). Because of the restriction of the language $\mathcal{L}_{sc}^{C^2}$, 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 $F(t_1, do(\alpha, \sigma))$,

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

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

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

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 $\mathcal{D}_{una}$,[10] we define the regression of W as

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

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

$$\mathcal{R}[W] \;=\; W. \tag{7}$$

Our intention here is to get a C^2 formula (with situation terms 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. (7) 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.

Note that in this definition (case (**c.**) and case (**d.**)), when $\vec{t}$ is $\langle x,x\rangle$ (or $\langle y,y\rangle$, respectively), we define regression by using 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

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

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

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

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 2 Assume that $\mathcal{D}$ is a BAT of $\mathcal{L}_{sc}^{C^2}$, which includes an SSA

$$F(x,y,do(a,s)) \;\equiv\; a = A(x) \wedge (\exists x.F_1(x,y,s)) \vee F(x,y,s). \tag{8}$$

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

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. (8), we get $A(C) = A(x) \wedge (\exists x.F_1(x, x, S_0)) \vee F(x, x, S_0))$, which obviously will not be logically equivalent to W w.r.t. $\mathcal{D}$. Indeed, in Eq. (8), the variable y in F_1 is free, but once substituted by x directly, it becomes quantified by $\exists x.$ at the front of F_1, which should not happen.

Based on the above definition, 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 of Theorem 2. This immediately follows from Def. 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 mentions only finitely many ground situation terms. □

According to the definition of the modified regression operator, the following statement can be proved by induction over the structure of W. The detailed proof 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 of Theorem 4. Part of our proof is almost word-by-word repetition of the laborious proof of the regression theorem given in [66]. Therefore, we will only briefly explain the idea of what has been done in [66] and provide details for what is different.

In [66], 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 "a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ 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 [66], Pirri and Reiter proved the following lemma that "suppose W is a regressable sentence with the background BAT $\mathcal{D}$ that is uniform in S_0, then $\mathcal{D} \models W$ iff $\mathcal{D}_{S_0} \cup \mathcal{D}_{una} \models W$" by using the above result. 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 $\mathbb{M}_0$ such that $\mathbb{M}$ has the same domain on *object* sort. 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, $S_0^{\mathbb{M}} = []$ (i.e., the empty sequence). 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 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 $\mathcal{D} \models W$ 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 of Theorem 5. 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 6, 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 school enrolment services.

5.2 Some Computational Complexity Results

We now consider the computational complexity of solving the projection problems in $\mathcal{L}_{sc}^{C^2}$. We 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
$size(W) = size(W_1 \supset W_2) + size(W_2 \supset W_1) + 1$.

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 σ. In particular, $sitLength(S_0) = 0$. For any formula W, let function

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

Given any BAT $\mathcal{D}$, for any fluent F whose SSA is of the form Eq. (4), let function $numFluent(F)$ be the number of fluents (including repeated ones) appearing in Φ_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 $\mathcal{D} \models W$ is in the complement of* NEXPTIME.

Proof of Theorem 6. 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 [68], 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 $\mathcal{D} \models W$ is in co-NTIME(2^{n_1}), i.e., the complement of NEXPTIME w.r.t. the size of $\mathcal{R}[W]$.

However, in the worst case, computing $\mathcal{R}[W]$ takes EXPTIME w.r.t. n, where $n = height(W)$, 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 with fixed steps of $\mathcal{L}_{sc}^{C^2}$ regression. 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 from above by a constant. Let $k = numFluentSSA(\mathcal{D})$ and $h = \max(2, sizeSSA(\mathcal{D}))$. Both k and h are constants for the given BAT $\mathcal{D}$, and $h > k$. Moreover, since all action functions in $\mathcal{L}_{sc}^{C^2}$ have no more than two arguments, the regression on equalities between action terms (see **a.** in the above definition of the $\mathcal{L}_{sc}^{C^2}$ regression operator) no more than triples the size of the regressed atomic formulas (including at most 1 conjunction operator and at most 2 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 in a regression tree (k out of these h branches have one atomic fluent as its node). The next application of the regression operator replaces fluents in these k nodes by the RHS of the corresponding SSAs, and so on. If the length of the longest situation term in W before regression is n, then 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). 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

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

Clearly, $3h(mk^n)$ 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 scenario, answering the query whether $\mathcal{D} \models W$ is in the complement of 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 $\mathcal{D} \models W$ is in the complement of NEEXPTIME (non-deterministic doubly-exponential time). □

We further consider some special cases of BATs that have better computational complexity results, but less expressive power.

We say that the SSA for a fluent F is *context-free* if the SSA of F has the syntactic form

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

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 [70]). 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. (4) 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 a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$, suppose that the SSA for a fluent F is context-free. Then, the computational complexity of answering the queries of the form $F(\vec{X}, \sigma)$ is in co-*NEXPTIME*, where $\vec{X}$ is a vector of object constants and σ is a ground situation term.*

Proof of Theorem 7. The result follows from the analysis of the computational complexity of the projection problem in [70] (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 $F(\vec{X}, \sigma)$ 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 *Given a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$, 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 $\mathcal{D} \models W$ is in co-*NEXPTIME.

5.3 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 mechanism) 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 FO^2, denoted FO_{DL}. The language of FO_{DL} includes constants, monadic and dyadic predicates. Moreover, it is a union of two sub-languages: $FO_{DL} = FO_{DL}^x \cup FO_{DL}^y$, where the detailed definition of FO_{DL}^x is provided below and FO_{DL}^y is obtained by renaming every x with y and every y with x for every formula in FO_{DL}^x. The set FO_{DL}^x is a minimal set of formulas built inductively as follows:

- $true$ and $false$ are in FO_{DL}^x.
- If AC is a monadic predicate name, then $AC(x)$ is in FO_{DL}^x.
- If b is a constant, then $x = b$ is in FO_{DL}^x.
- If ϕ is in FO_{DL}^x, then $\neg\phi$ is in FO_{DL}^x.
- If ϕ and ψ are in FO_{DL}^x, then $\phi\wedge\psi$ and $\phi\vee\psi$ are in FO_{DL}^x.
- If $\phi(x)$ is in FO_{DL}^x, 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 ϕ, then $\exists y.R(x,y)\wedge\widetilde{\phi}(y)$ and $\forall y.R(x,y)\supset\widetilde{\phi}(y)$ are in FO_{DL}^x.
- If ϕ is in FO_{DL}^x, $\widetilde{\phi}$ is the dual formula of ϕ, obtained by renaming every x (both free and bound) with y and every y (both free and bound) with x in ϕ, then $[\exists y.]\widetilde{\phi}(y)$ and $[\forall y.]\widetilde{\phi}(y)$ are in FO_{DL}^x, 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 FO^2. Notice that for any $\phi \in FO_{DL}^x$ ($\phi \in FO_{DL}^y$, respectively), $\widetilde{\phi}$ is in FO_{DL}^y (FO_{DL}^x, respectively). Moreover, it is easy to see that any sentence (i.e., closed formula) in FO_{DL} is in both FO_{DL}^x and FO_{DL}^y. 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 FO^2 (see Section 3.2). For this reason we would like to investigate a new SC based on FO_{DL}.

Definition 2 *We say that the SSA for a fluent F is* $\mathcal{ALCO}(U)$-restricted *if the SSA of F has the form of Eq. (4), where each context condition ψ_i^+ (or ψ_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. (5) 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. (5) 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.*

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

Lemma 2 *Consider a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_T$ are $\mathcal{ALCO}(U)$-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 Φ_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 Φ_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 Φ_W is no more than exponential in the size of W.*

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 $\mathcal{D} \models W$ can be solved in* EEXPTIME.

Proof of Theorem 8. 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 Φ 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 Φ 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 Ψ 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 Ψ is in FO_{DL}, because $\mathcal{D}_{S_0}$ is in FO_{DL} when the situation term S_0 is suppressed and Φ is in FO_{DL}. Ψ 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 π 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 Φ. 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 Φ). Again, because the size of Φ is exponential in the size of W, deciding whether $\mathcal{D}_{S_0} \models \mathcal{R}[W]$ is in the complement of EEXPTIME (wrt the size of W), which is the same as EEXPTIME.
□

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}. Formally, $FO_{DL+} = FO_{DL+}^x \cup FO_{DL+}^y$, where FO_{DL+}^x is a minimal set of formulas built inductively below, and $FO_{DL+}^y = \{\widetilde{\phi} \mid \phi \in FO_{DL+}^x\}$.

- $true$ and $false$ are in FO_{DL+}^x.

- 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 ϕ is in FO^x_{DL+}, then $\neg\phi$ is in FO^x_{DL+}.
- If ϕ and ψ 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 ϕ, 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 ϕ is in FO^x_{DL+}, $\widetilde{\phi}$ i dual formula of ϕ, obtained by renaming every x (both free and bound) with y and every y (both free and bound) with x in ϕ, 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.[11]

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. (4), where each context condition ψ_i^+ (or ψ_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 $\mathcal{D}_{ss}$ is both $\mathcal{ALCQO}(U)$-restricted and context-free. We say that a concept definition of the form Eq. (5) 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. (5) 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 $\mathcal{L}_{sc}^{C^2}$ whose $\mathcal{D}_{ss}$ and $\mathcal{D}_T$ are $\mathcal{ALCQO}(U)$-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 Φ_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 Φ_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 Φ_W is no more than exponential in the size of W.*

[11] 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.

Proof of Lemma 4. The proof is exactly the same as the proof of Lemma 2 (see Appendix B.4), where FO_{DL} (FO_{DL}^{x}, or FO_{DL}^{y}, respectively) is replaced by FO_{DL+} (FO_{DL+}^{x}, or FO_{DL+}^{y}, 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 $\mathcal{L}_{sc}^{C^2}$ 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 $\mathcal{L}_{sc}^{C^2}$ 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 $\mathcal{D} \models W$ can be solved in* EEXPTIME.

6 An Example of Regression in $\mathcal{L}_{sc}^{C^2}$

In this section, we give an example of a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ to illustrate the ideas described above.

Example 1 Consider some 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 that 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: $hasGrade(x, y, s)$ (x has a grade for course y in the situation s), $tuitPaid(x, y, s)$ (x paid tuition fee 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 is as follows (most of the axioms are self-explanatory).

Precondition Axioms:

$$Poss(reset, s) \equiv true,$$
$$Poss(admit(x), s) \equiv person(x) \wedge \neg incoming(x, s),$$
$$Poss(payTuit(x, y), s) \equiv incoming(x, s) \wedge \neg tuitPaid(x, y, s),$$
$$Poss(drop(x, y), s) \equiv enrolled(x, y, s) \wedge \neg completed(x, y, s),$$
$$Poss(enterA(x, y), s) \equiv enrolled(x, y, s) \wedge \neg completed(x, y, s),$$

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

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

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: 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. They ensure that for each student and each course there is no more than one grade assigned. The SSA for the fluent $hasGrade(x, y, s)$ is also similar and for this reason it is omitted.

$$incoming(x, do(a, s)) \equiv a = admit(x) \vee incoming(x, s),$$

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

$$completed(x,y,do(a,s)) \equiv a{=}enterA(x,y) \vee a{=}enterB(x,y)$$
$$\vee a{=}enterC(x,y) \vee a{=}enterD(x,y) \vee$$
$$completed(x,y,s) \wedge a \neq enroll(x,y),$$

$$gradeA(x,y,do(a,s)) \equiv a{=}enterA(x,y) \vee gradeA(x,y,s) \wedge$$
$$\neg(a{=}enterB(x,y) \vee a{=}enterC(x,y) \vee a{=}enterD(x,y)),$$

and the SSA for fluent $gradeB(x,y,s)$ (for $gradeC(x,y,s)$ and for $gradeD(x,y,s)$, respectively) is similar to the one for fluent $gradeA(x,y,s)$.

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

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

An example of the initial theory $\mathcal{D}_{S_0}$ could be the conjunctions of the following sentences: $(\forall x,y)\neg paidTuit(x,y,S_0)$, $(\forall x)incoming(x,S_0) \supset x = P_2 \vee x = P_3$, $preReq(CS_1,CS_4) \vee preReq(CS_3,CS_4)$, $(\forall x)x \neq CS_4 \supset \neg(\exists y).prePeq(y,x)$, $person(P_1), \cdots, person(P_m)$.

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:

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

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 $A_1; A_2; \cdots; A_n$ 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.

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: [56] assumes that the complete information is available about the world when effects of a composite service are computed, and [25, 18] consider the propositional fragment of the SC.

In [56, 61], 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 SC too. It is surprisingly straightforward to define almost all Golog operators starting from our C^2-based SC. The only restriction in comparison with the original Golog [41, 70] is that we cannot define the operator $(\pi x)\delta(x)$, non-deterministic choice of an action argument, because $\mathcal{L}_{sc}^{C^2}$ 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], 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 and then defines services (actions) meta-theoretically: an atomic service is defined as the triple of sets of description logic formulas. To solve the executability and projection problems the paper introduces an approach similar to regression, and reduces this problem to description logic reasoning. 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 paper and [2] have common goals, our developments start differently and proceed in different directions. We start from the syntactically restricted FO language (that is significantly more expressive than $\mathcal{ALCQIO}$), use it to construct the modified SC (where actions are terms), define BATs in this language and show that by augmenting (appropriately modified) regression with lazy unfolding one can reduce the executability and projection problems to the satisfiability problem in C^2, which is decidable. Furthermore, C^2 formulas can be translated to $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$, if desired. Because our regression operator unfolds fluents "on demand" and uses only relevant part of the (potentially huge) TBox , we avoid potential computational problems that could occur if the TBox axiom were eliminated in advance. The advantage of [2] 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 two-fold: the convenience of representing actions as terms, and the expressive power of $\mathcal{L}_{sc}^{C^2}$. Because C^2 and $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ are equally expressive, there are some (situation suppressed) formulas in our SC that cannot be expressed in $\mathcal{ALCQIO}$ (that does not allow complex roles). In particular, [79] 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 $\mathcal{ALCQI}$. The more recent papers of the same research group continue to explore the research direction initiated in [2]: [59, 58] investigate complexity of planning in a description logic based action formalism, [45] 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 [2].

Propositional dynamic logic (**PDL**) was derived from dynamic logic and has several nice properties: **PDL** has the finite model property and is decidable [33]. Its satisfiability problem is ExpTime-complete [19, 67]. It turned out to be popular not only for reasoning about regular programs, but also as a logic of action [24, 69]. 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 multimodal logic. But, in **PDL** quantification over actions is not allowed. More recently, [12, 16, 15] 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 [70]. 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 paper, 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 [84], 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, [84] 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 [21, 22, 23] for the definition and survey of results). The issues related to combining modal logics in a more general context are reviewed in [40]. 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 α is an action term and C is a concept, but there is no similar constructor for roles. However, for any atomic formula ϕ 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 α, $[\alpha]\phi$ is also a formula. The main contribution of [84] 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 practical purposes. In [14, 13], the authors propose a logic that is similar to [3, 2, 84]. In the proposed logic, one not only can reason about complex actions similar to [84], but also can characterize actions by preconditions and conditional effects as in [2]. Also, a tableau algorithm for deciding satisfiability proposed in [14, 13] is based on an elaborated combination of previously known tableau algorithms.

In our paper [32], 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 [44] for an incomplete KB given *local-effect* SSAs defined in [47], . 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 KB (a set of disjunctions of equality-based formulas). Recently, [82] considers a notion of *strong progression*, a slight variant of the classical progression. In [82], 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. Whether or not a similar result applies to our modified SC, i.e., whether or not the strong progression is C^2 *definable* for a BAT $\mathcal{D}$ in $\mathcal{L}_{sc}^{C^2}$ with local-effect SSAs, requires further study. It is easy to check that in all special cases from [44, 77, 47] when the progression is FOL definable, similar results can also be formulated for our $\mathcal{L}_{sc}^{C^2}$ simply because we restrict the language to two object variables only. There is research on updating in the description logic community that is somewhat related to computing progression in the SC. [46] considers update of an ABox in a DL with an acyclic TBox following [83] 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). [26] uses a less expressive DL language $DL\text{-}Lite$, but defines update for the case when TBox consists of GCIs in comparison to acyclic TBox that is required in [46]. It is shown that the result of an update is always expressible by a $DL\text{-}Lite$ ABox and a polynomial-time algorithm is provided that computes the update over a $DL\text{-}Lite$ KB. The more recent paper [11] 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 $DL\text{-}Lite$.

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 [32], Drescher and Thielscher [17] explored reasoning about actions based on a description logic, but they concentrate on the *fluent calculus* [71] 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, 4], 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 [27]. Both [1] and [27] 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 [39] 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 [7], 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.

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 [60].

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 efficiently on practical problems. There are several different directions that can be explored. First, according to [9] 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 5.3 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 [73, 74], for (un)satisfiability checking in $\mathcal{ALCQIO}(\sqcup, \sqcap, \neg, |, id)$ and in $\mathcal{ALCO}(U)$. 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 $\mathcal{D}_{S_0}$ [37, 62]. 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.

Finally, we would like to explore how our version of the SC can accommodate events considered by John McCarthy in [54].

Acknowledgments

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.

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 [4].

Name	Syntax	Interpretation
Top	$\top$	$\Delta^{\mathcal{I}}$
Bottom	$\bot$	$\emptyset$
Nominal	$\{b\}$	$b^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
Negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
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	$\forall R.C$	$\{\delta \in \Delta^{\mathcal{I}} \mid \forall \delta_1.(\delta, \delta_1) \in R^{\mathcal{I}} \supset \delta_1 \in C^{\mathcal{I}}\}$

Figure 1: The semantics of some common description logic concept 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	$R\|_C$	$R^{\mathcal{I}} \cap \Delta^{\mathcal{I}} \times C^{\mathcal{I}}$
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	$R_1 \circ R_2$	$\{(\delta_1, \delta_2) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid$ $\exists \delta \in \Delta^{\mathcal{I}}.(\delta_1, \delta) \in R_1^{\mathcal{I}} \wedge (\delta, \delta_2) \in R_2^{\mathcal{I}}\}$

Figure 2: The semantics of some common description logic role constructors.

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	$R(b_1, b_2)$	$(b_1^{\mathcal{I}}, b_2^{\mathcal{I}}) \in R^{\mathcal{I}}$

Figure 3: The semantics of terminological and assertional axioms.

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 of Lemma 5. Similar to the proof in [9], 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).

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

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

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

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

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

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

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

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

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

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

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

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 ϕ 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 τ 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 of Lemma 6. 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_Γ. 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 $C^2\}$.

$\Gamma(x)$	C_Γ	$\Gamma(x)$	C_Γ
$AC(x)$, $AC \in N_C$	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_Ψ
$\neg\Psi(x)$	$\neg C_\Psi$	$\exists^{\geq n} y.\Psi(x)$	C_Ψ
		$\exists^{\leq n} y.\Psi(x)$	C_Ψ

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

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 Ψ. So, it is translated as $\exists(R_\Psi)^-.\top$ ($\geqslant n(R_\Psi)^-.\top$, and $\leqslant n(R_\Psi)^-.\top$, respectively).

Formulae of the form $\Gamma(x,y)$ with two free variables are translated to roles R_Γ relating x to y according to Table 7.

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

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

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 [9], 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,

$$C_1 \times C_2 \overset{def}{=} ((R \sqcup \neg R)|_{C_1})^- \sqcap (R \sqcup \neg R)|_{C_2}$$

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), $\mathcal{I}_1 \models \Gamma() \equiv true$ iff $(C_\Gamma)^{\mathcal{I}} = \Delta^{\mathcal{I}}$, and $\mathcal{I}_1 \models \Gamma() \equiv false$ iff $(C_\Gamma)^{\mathcal{I}} = \emptyset$. Table 8 provides such translations.

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

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

In contexts where we require roles, the translation is just $R_{\Gamma()} = C_{\Gamma()} \times C_{\Gamma()}$.

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

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 τ 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 of Theorem 1. It is a direct consequence of combining Lemma 5 and Lemma 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 of Theorem 3. 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

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

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,

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

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,

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

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 σ, 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

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

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,

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

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, respectively) 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 α of sort *action* and ground term σ 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. (4). Because of the restriction of the language $\mathcal{L}_{sc}^{C^2}$, 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,

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

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 subsection, we prove Lemma 1 presented in Section 5.3. Notice that in the proof of this Lemma, we provide purely syntactic translation functions between $\mathcal{ALCO}(U)$ and FO_{DL}.

Lemma 1 (Section 5.3) *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 of Lemma 1. We first prove that there is a syntactic translation function from $\mathcal{ALCO}(U)$ to FO_{DL}.

A syntactic translation τ from $\mathcal{ALCO}(U)$ to FO_{DL} for any concept C is defined as follows: $\tau(C) \overset{def}{=} \tau^x(C)$ 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 τ – $\tau^y()$ makes y be the free variable of the monadic predicate (see Table 9).

Then, the translation of terminological and assertional axioms can be defined as:
$\tau(C(b)) \overset{def}{=} \tau(\exists U.(C \cap \{b\}))$ for any concept assertion $C(b)$;
$\tau(R(b,b')) \overset{def}{=} \tau(\exists U.((\exists R.\{b'\}) \cap \{b\}))$ for any role assertion $R(b,b')$;
$\tau(C_1 \sqsubseteq C_2) \overset{def}{=} \tau(\neg\exists U.(C_1 \wedge \neg C_2))$ for any concept inclusion axiom $C_1 \sqsubseteq C_2$ if there is any;
$\tau(C_1 \equiv C_2) \overset{def}{=} \tau(\forall U.((\neg C_1 \cup C_2) \cap (\neg C_1 \cup C_1)))$ for any concept equality axiom $C_1 \equiv C_2$ if there is any.

In addition, according to the definition of τ 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 π from FO_{DL} to $\mathcal{ALCO}(U)$ for any formula $\Phi \in FO_{DL}$ is defined in Table 10.

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

Figure 9: A syntactic translation from $\mathcal{ALCO}(U)$ to FO_{DL}.

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 π defined above. □

B.4 Restricting Syntax of BATs to Gain Computational Advantages

In this subsection, we will prove Lemma 2 in Section 5.3. But first, we define an operator ϵ on any $\mathcal{L}_{sc}^{C^2}$ 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 $\mathcal{D}$.

For any given BAT $\mathcal{D}$ and an $\mathcal{L}_{sc}^{C^2}$ regressable formula W in it, we define ϵ 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

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

Note that ϵ 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 ϵ.

Φ	$\pi(\Phi)$
$AC(x)$, $AC(x)$ 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	$\forall U.\pi(\Psi(x))$

Figure 10: A syntactic translation from FO_{DL} to $\mathcal{ALCO}(U)$.

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 of Property 1. 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 ϵ; 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 ϵ; 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 ϵ. 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 ϵ, 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

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

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 2 in Section 5.3. 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 of Lemma 7. 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 $\mathcal{R}[W] = false$ by the definition of $\mathcal{R}$ in Section 5.1, and $\mathcal{R}[\epsilon[W]] = \mathcal{R}[false] = false$ by the definitions of ϵ 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 ϵ and $\mathcal{R}$, it is easy to see that $\mathcal{R}[W] = \mathcal{R}[\epsilon[W]] = true$;
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 ϵ 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 ϵ, 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

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

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 $\mathcal{L}_{sc}^{C^2}$ regressable formula W in $\mathcal{D}$. □

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

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

- 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

$$\begin{array}{rclrcl} \rho[W_1 \vee W_2] &=& \rho[W_1] \vee \rho[W_2], & \rho[\neg W'] &=& \neg\rho[W'], \\ \rho[W_1 \wedge W_2] &=& \rho[W_1] \wedge \rho[W_2], & \rho[Qv.W'] &=& Qv.\rho[W']. \end{array}$$

- 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 σ, 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 $\mathcal{L}_{sc}^{C^2}$, term t can only be a variable x, y or a constant. Then, we use the lazy unfolding technique as follows:

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

  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 α of sort *action* and term σ of sort *situation*. Then there must be an SSA Eq. (1) for fluent F, whose detailed syntax is Eq. (4). Because of the restriction of the language $\mathcal{L}_{sc}^{C^2}$, the terms t_1 and t_2 can only be a variable x, y or a constant O and α can only be an action function with no more than two arguments of sort *object*. Then, when W is a concept,

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

  and, when W is a role,

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

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

Property 3 *For any given BAT $\mathcal{D}$ and an $\mathcal{L}_{sc}^{C^2}$ regressable formula W in $\mathcal{D}$, we have that $\rho[W]$ is still $\mathcal{L}_{sc}^{C^2}$ 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 2. Because the proof is rather obvious, it is omitted here.

Property 4 *Consider a BAT $\mathcal{D}$ in the language of $\mathcal{L}_{sc}^{C^2}$, if a given $\mathcal{L}_{sc}^{C^2}$ regressable formula W is uniform in $do(\alpha, S)$ for some ground action α 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 $\mathcal{L}_{sc}^{C^2}$ regression operator $\mathcal{R}$ defined in Section 5.1, and any $\mathcal{L}_{sc}^{C^2}$ regressable formula W in $\mathcal{D}$. Then, $\mathcal{R}[W] = \mathcal{R}[\rho[W]]$.*

Moreover, according to the definition of ρ, it is straightforward to prove the following property of ρ.

Property 5 *Consider any $\mathcal{L}_{sc}^{C^2}$ regressable formula W with a background BAT $\mathcal{D}$. Assume that there is no appearance of $Poss$ in W. Let $m = size(W)$, $h = \max(2, sizeSSA(\mathcal{D}))$, and $h_1 = \max_G\{size(\Phi_G) \mid G(x) \equiv \Phi_G(x)$ is a TBox axiom$\}$ if there are TBox axioms, or $h_1 = 0$. Notice that h and h_1 are fixed when $\mathcal{D}$ is given. Then, it takes no more than $(h_1 + 1)m$ steps to obtain $\rho[W]$, whose size is no more than $h(h_1 + 1)m$.*

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

Corollary 2 *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, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]]$.*

Proof of Corollary 2. 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 2 in Section 5.3.

Lemma 2 (Section 5.3) *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 W be any $\mathcal{L}_{sc}^{C^2}$ regressable formula in $\mathcal{D}$ that is uniform in a ground situation S and has no appearance of $Poss$. Let $n = sitLength(S)$ and $m = size(W)$. Thene if W with the situation term S suppressed is in FO_{DL}, there is a Φ_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 Φ_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 Φ_W is no more than exponential in the size of W.*

Proof of Lemma 2. 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 fixed steps of $\mathcal{L}_{sc}^{C^2}$ regression. This can cause 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 a formula always exists, and then estimate the size of the formula. 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.

We will first prove the following more specific statement (**Statement (1)**) below wrt the given $\mathcal{D}$: "Consider any ground situation S and a $\mathcal{L}_{sc}^{C^2}$ 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_{DL}^x (FO_{DL}^y, respectively), there is a formula φ in FO_{DL}^x (FO_{DL}^y, 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 $\mathcal{L}_{sc}^{C^2}$ 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. 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 $\mathcal{L}_{sc}^{C^2}$ regressable formula W' that is uniform in S_1 and has no appearance of $Poss$. We prove Statement (1) for any $\mathcal{L}_{sc}^{C^2}$ 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_{DL}^y is a dual formula to a formula in FO_{DL}^x, the proof for Statement (1) where $W^{[-S]}$ is in FO_{DL}^y is "*dual*" to the proof for Statement (1) where $W^{[-S]}$ is in FO_{DL}^x, 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 detailed proof for Statement (1) where $W^{[-S]} \in FO_{DL}^x$, and omit details for the proof of Statement (1) where $W^{[-S]} \in FO_{DL}^y$.

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

"For any $\mathcal{L}_{sc}^{C^2}$ 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_{DL}^x (FO_{DL}^y, respectively), then there is a formula φ in FO_{DL}^x (FO_{DL}^y, respectively) which can be found in no more than $c \cdot size(\varphi)$ steps for some constant positive integer c. Moreover, $\varphi[S_1]$ is equivalent to $\epsilon[\rho[W]]$, and $\varphi[S_1]$ is $\mathcal{L}_{sc}^{C^2}$ 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_{DL}^x 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_{DL}^x; and, $(\mathcal{R}[W])^{[-S_0]}$ is still $true$ or $false$, which is in FO_{DL}^x. 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_{DL}^x; and, $(\mathcal{R}[W])^{[-S_0]}$ is still $x = b$, which is in FO_{DL}^x. 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_{DL}^x; and, $(\mathcal{R}[W])^{[-S_0]} = W = W^{[-S]}$, which is in FO_{DL}^x. 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. (4), whose context conditions (with situation terms suppressed) are all in FO_{DL}. Depending on whether the context conditions are in FO_{DL}^x (e.g., cases (1-12) in Table 11) or in FO_{DL}^y (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

$$F(x, do(a, s)) \equiv \bigvee_{i=1}^{m_+} \phi_i^+(x, a, s) \vee F(x, s) \wedge \neg(\bigvee_{j=1}^{m_-} \phi_j^-(x, a, s)), \tag{9}$$

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 the cases we described in Note 1. 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_{DL}^x (FO_{DL}^y, 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 binary action function name. Moreover, in Table 11, $[\exists y.]$ represents that $\exists y.$ only appears when $\psi(y)$ has a free variable y. To show that we have exhausted all the possibilities, the cases we listed 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 F_{DL}^x (according to the definition of F_{DL}^x).

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

Figure 11: Some possible syntactic forms for $\phi_i^+(x, a, s)$ or $\phi_j^-(x, a, s)$ in Eq. (9)

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 ϕ_i^+ and/or ϕ_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 ϕ_i^+ and/or ϕ_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 ϕ_i^+ and/or ϕ_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 ϕ_i^+ and/or ϕ_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 $\mathcal{L}_{sc}^{C^2}$ 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 exists some ν_i^+ (or ν_j^-, respectively) in FO_{DL}^x for any i (j, respectively) such that $\nu_i^+[S_1]$ ($\nu_j^-[S_1]$, respectively) is logically equivalent to $\epsilon[\phi_i^+(x, \alpha, S_1)]$ ($\epsilon[\phi_j^-(x, \alpha, S_1)]$, respectively). Moreover, finding the equivalent formula takes a fixed number of steps of deduction w.r.t. 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 α is not A, then in each of the aforementioned cases (1-12), (1'-12') and Note 1, ϵ of each formula (ϕ_i^+ or ϕ_j^-) equals $false$, which is still in FO^x_{DL}. Hence, below we only discuss the condition that α 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
$$\begin{aligned} & \epsilon[\alpha = A \wedge \psi(x)[S_1]] \\ = \ & \epsilon[\alpha = A] \wedge \epsilon[\psi(x)[S_1]]] \\ = \ & true \wedge \psi(x)[S_1] \ \equiv \ \psi(x)[S_1]. \end{aligned}$$

Clearly, $\psi(x)$ is in FO^x_{DL}, which takes at most one step of logical deduction 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^y_{DL}, we have that $x = O \wedge \psi(x)$ is in FO^x_{DL}, and this takes no steps of logical deduction.

(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 closed formula $(\exists x.x = O \wedge \psi(x))$ is in FO^x_{DL}. Because $x = O \wedge \psi(x)$ is in FO^x_{DL}, then by definition, $(\exists y.y = O \wedge \widetilde{\psi}(y))$ (denoted as φ) is in FO^x_{DL}. Again by the definition of FO^x_{DL}, $[\exists y.]\widetilde{\varphi}$ (i.e., $(\exists x.x = O \wedge \psi(x))$) is in FO^x_{DL}. It takes no steps of logical deduction.

(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 deduction.

(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 deduction to minimize the quantification scope of $\exists y$.

(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 deduction.

(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 deduction 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 deduction 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

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

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

(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

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

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 of logical deduction to minimize the scope of $\exists x$.

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

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

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

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 deduction to minimize the scope of $\exists x$.

(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 α and s is substituted by a ground situation S_1, by the definition of regression, ϵ and ρ,

$$\begin{aligned}
\epsilon[\rho[F(x,S)]] &= \epsilon[\rho[F(x,do(\alpha,S_1))]] \\
&= \epsilon\left[\bigvee_{i=1}^{m_+}\phi_i^+(x,\alpha,S_1)\vee F(x,S_1)\wedge\neg(\bigvee_{j=1}^{m_-}\phi_j^-(x,\alpha,S_1))\right] \\
&\qquad\text{(by the definition of } \rho\text{),} \\
&= \bigvee_{i=1}^{m_+}\epsilon[\phi_i^+(x,\alpha,S_1)]\vee F(x,S_1)\wedge\neg(\bigvee_{j=1}^{m_-}\epsilon[\phi_j^-(x,\alpha,S_1))] \\
&\qquad\text{(by the definition of } \epsilon\text{),} \\
&\equiv \bigvee_{i=1}^{m_+}\nu_i^+(x)[S_1]\vee F(x,S_1)\wedge\neg(\bigvee_{j=1}^{m_-}\nu_j^-(x)[S_1] \\
&\qquad\text{(according to the proof of cases (1-12,1'-12')),} \\
&\equiv (\bigvee_{i=1}^{m_+}\nu_i^+(x)\vee F(x)\wedge\neg(\bigvee_{j=1}^{m_-}\nu_j^-(x))[S_1], \qquad (10)
\end{aligned}$$

where each $\nu_i^+(x)$ ($\nu_j^-(x)$, respectively) is a formula in FO_{DL}^x that has at most one free variable x, and it 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. (10) is regressable, uniform in S_1, and in FO_{DL}^x (with S_1 suppressed) according to the definition of the set FO_{DL}^x. 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 2 (i.e., $\mathcal{R}[F(x,S)] = \mathcal{R}[\epsilon[\rho[F(x,S)]]]$), we have $(\mathcal{R}[F(x,do(\alpha,S_1))])^{[-S_0]}$ will still be equivalent to some formula in FO_{DL}^x.

Similarly, we can prove Statements (1) and (2) for $W^{[-S]}$ that is in FO_{DL}^y 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. There are in total four cases (a-d) as follows.

a. $W^{[-S]}$ is of the form $\neg W_1$, where $W_1 \in FO_{DL}^x$.

Then, 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 $\phi_1 \in FO_{DL}^x$ 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 2, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[\neg\phi_1[S_1]]$. Next, by the induction hypothesis on S_1, $\mathcal{R}[W]^{[-S_0]}$ is equivalent to some formula in FO_{DL}^x, 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 $W_1, W_2 \in FO_{DL}^x$.

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 $\phi_1, \phi_2 \in FO_{DL}^x$ 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 $c \cdot (size(\phi_1) + size(\phi_2) + 1)$ steps for some integer c. Hence, $\epsilon[\rho[W]] \equiv (\phi_1 \wedge \phi_2)[S_1]$, and Statement (2) is true for W. Then, according to Corollary 2, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[(\phi_1 \wedge \phi_2)[S_1]]$. Next, by the induction hypothesis on S_1, $\mathcal{R}[W]^{[-S_0]}$ is equivalent to some formula in FO_{DL}^x, 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 $[\exists y.]W_1(y)$, $\epsilon[\rho[W]] = [\exists y.]\epsilon[\rho[W_1(y)[S]]]$. 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. Hence, $\epsilon[\rho[W]] \equiv ([\exists y.]\phi_1(y))[S_1]$, and Statement (2) is true for W. Then, according to Corollary 2, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[([\exists y.]\phi_1(y))[S_1]]$. 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, 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 2, $\mathcal{R}[W] = \mathcal{R}[\epsilon[\rho[W]]] \equiv \mathcal{R}[([\exists y.]R(x,y) \wedge \phi_1(y))[S_1]]$. 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 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} (e.g., cases (1-16) in Table 12) or FO^y_{DL} (e.g., 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

$$R(x,y,do(a,s)) \;\equiv\; \bigvee_{i=1}^{m_+} \phi_i^+(x,y,a) \vee R(x,y,s) \wedge \neg(\bigvee_{j=1}^{m_-} \phi_j^-(x,y,a)), \qquad (11)$$

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 cases (1) and (1'), A represents some constant action function. In cases (2-7) and (2'-7') , A represents some unary action function name. And, in cases (8-16) and (8'-16'), A represents some binary action function name. Again, to show that we have exhausted all the possibilities, the cases we listed in Table 12 still include some other duplications. For example, case (6) in fact is the same as case (3) by renaming.

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

Figure 12: Some possible syntactic forms for $\phi_i^+(x, y, a)$ or $\phi_j^-(x, y, a)$ in Eq. (11)

Note 2 For any formula $\psi_1(x)$ ($\psi_1(y)$, respectively) in FO_{DL}^x (FO_{DL}^y, 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_{DL}^x and FO_{DL}^y, which can be considered as a special case of the form $\psi(x) \in FO_{DL}^x$ or $\psi(y) \in FO_{DL}^y$. 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 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 ϕ_i^+ and/or ϕ_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 ϕ_i^+ and/or ϕ_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 ϕ_i^+ and/or ϕ_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 ϕ_i^+ and/or ϕ_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 $\mathcal{L}_{sc}^{C^2}$ regressable formula W. Moreover, such deduction takes at most linear number of steps in 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 some formula of the form $(\nu(x) \wedge \eta(y))$ for some $\nu(x) \in FO_{DL}^x$ and $\eta(y)) \in FO_{DL}^y$. In particular, 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 $\nu(x) \wedge \eta(y)$: $\nu(x)$ (let $eta(y)$ be $true$), $\eta(y)$ (let $nu(x))$ be $true$), $\nu(x) \wedge y = O$ for some constant O, or $x = O \wedge \eta(y)$ for some constant O.

From now on, without particular emphasis, all the cases we discuss below are the cases in Table 12.

Moreover, for similar proofs, some detailed steps are skipped.

Here is one trivial sub-case: if the function name of α is not A, then in each of the aforementioned cases (1-16), (1'-16') and in Note 2, ϵ of each formula (ϕ_i^+ or ϕ_j^-) equals $false$, which is still in FO_{DL}^x. Hence, below we only discuss the condition that α 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_{DL}^x 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
$$\begin{aligned} & \epsilon[\alpha = A(y) \wedge \psi(x)] \\ = \; & y = O \wedge \psi(x). \end{aligned}$$

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

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

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

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

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

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

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

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

Clearly, $y = O_2\wedge\exists x(x = O_1)\wedge\psi(x)$ is of the form $\nu(x)\wedge\eta(y)$ (let $\eta(y)$ 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 deduction 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 is constant to the size of the resulting formula. Now, we prove that Statement (2) is true for case (d.2).

$$
\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] \\
&\quad \text{(by the induction hypothesis on } W_1(y)[S_1], \text{ let } \epsilon[\rho[W_1(y)[S]]] \equiv W_1'(y)[S_1] \\
&\quad \text{be a formula uniform in } S_1 \text{ and } W_1'(y) \text{ is in } FO^y_{DL}; \text{ moreover, } W_1'(y) \\
&\quad \text{can be found in no more than } c\cdot size(W_1'(y))) \text{ steps} \\
&= \exists y.\epsilon[(\bigvee_{i=1}^{m_+}\phi_i^+(x,y,\alpha)\vee R(x,y,S_1)\wedge\neg(\bigvee_{j=1}^{m_-}\phi_j^-(x,y,\alpha))]\wedge W_1'(y)[S_1] \\
&= \exists y.(\bigvee_{i=1}^{m_+}\epsilon[\phi_i^+(x,y,\alpha)]\vee R(x,y,S_1)\wedge\neg(\bigvee_{j=1}^{m_-}\epsilon[\phi_j^-(x,y,\alpha)])\wedge W_1'(y)[S_1] \\
&\equiv \exists y.(\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)))\wedge W_1'(y)[S_1] \\
&\quad \text{(according the proof above, for each } i, \text{ let } \epsilon[\phi_i^+(x,y,\alpha)] \equiv (\nu_i^+(x)\wedge\eta_i^+(y)) \\
&\quad \text{be a formula uniform in } S_1 \text{ for some } \nu_i^+(x)\in FO^x_{DL} \text{ and } \eta_i^+(y)\in FO^y_{DL}; \\
&\quad \text{for each } j, \text{ let } \epsilon[\phi_j^-(x,y,\alpha)] \equiv (\nu_j^-(x)\wedge\eta_j^-(y)) \text{ be a formula uniform} \\
&\quad \text{in } S_1 \text{ for some } \nu_j^-(x)\in FO^x_{DL} \text{ and } \eta_j^-(y)\in FO^y_{DL}) \\
&\equiv \exists y.(\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)))[S_1] \\
&\equiv \{\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)\}[S_1] \\
&\equiv \{\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)\}[S_1] \\
&\equiv \{\bigvee_{i=1}^{m_+}\nu_i^+(x)\wedge\exists y(\eta_i^+(y)\wedge W_1'(y))\vee \\
&\quad \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))\}[S_1] \qquad (12) \\
&= W'[S_1],
\end{aligned}
$$

where each N_k ($1\le k\le 2^{m_-}$) enumerates a subset of $\{1,2,\cdots,m_-\}$, and $\overline{N_k}=\{1,2,\cdots,m_-\}-N_k$, i.e., $\overline{N_k}$ is the complement set of N_k. It is clear the formula on the RHS of Eq. (12) (denoted as $W'[S_1]$ above) is equivalent to $\epsilon[\rho[W]]$, is $\mathcal{L}_{sc}^{C^2}$ regressable, and is in FO^x_{DL} 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 2, 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^x_{DL}.

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 Statements (1) and (2) are true when $W^{[-S]}$ is in FO^y_{DL} and is not atomic.

And overall, we proved for Statement (1).

Now, consider any $\mathcal{L}_{sc}^{C^2}$ regressable W that is uniform in a ground situation S. When $W^{[-S]}$ is in FO_{DL}^{x}, assume that we have found some Φ_W that is in FO_{DL}^{x}, such that $\mathcal{R}[W] \equiv \Phi_W[S_0]$. Below we estimate the upper bound on the size of Φ_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 Φ_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 2, 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 $\Phi_1 \in FO_{DL}^{x}$, 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_{DL}^{x}, 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_{DL}^{x}, or $W_1(y)$ is in FO_{DL}^{y}. 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_{DL}^{x} constructed. For example, $\mathcal{R}[\neg W_1[S]] = \neg\mathcal{R}[W_1[S]]$, if we find a formula Φ_1 in FO_{DL}^{x} that is equivalent to $\mathcal{R}[W_1[S]]^{[-S_0]}$, 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_{DL}^{x}. 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_{DL}^{x} 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_{DL}^{x} and R is a dynamic predicate name. According to the definition of $size$ in Section 5.2, 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

$$\begin{aligned} \mathcal{R}[W] &= \mathcal{R}[\exists y.R(x,y)[S] \wedge W_1(y)[S]] \\ &= \exists y.\mathcal{R}[R(x,y)[S]] \wedge \mathcal{R}[W_1(y)[S]] \\ &= \begin{cases} \exists y.\mathcal{R}[R(x,y,S)] \wedge \mathcal{R}[W_1(y)[S]] & \text{if } R \text{ is a fluent,} \\ \exists y.R(x,y) \wedge \mathcal{R}[W_1(y)[S]] & \text{otherwise.} \end{cases} \end{aligned} \tag{13}$$

Assume that $\mathcal{R}[W_1(y)[S]]^{[-S_0]}$ is equivalent to some $\Phi_1(y) \in FO_{DL}^{y}$. 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. (11) and $S = do([\alpha_1, \alpha_2, \cdots, \alpha_n], S_0)$.

Let $S_i = do([\alpha_1, \cdots, \alpha_{n-i}], S_0)$ for any $1 \leq i \leq n-1$, then

$$\mathcal{R}[W] \;=\; \exists y.\mathcal{R}[R(x,y,S)] \wedge \mathcal{R}[W_1(y)[S]] \tag{14}$$

$$\equiv \quad \exists y.\mathcal{R}[R(x,y,S)] \wedge \Phi_1(y)[S_0]$$

$$\equiv \quad \exists y.\{\bigvee_{i=1}^{m_+} \mathcal{R}[\phi_i^+(x,y,\alpha_n)] \vee \mathcal{R}[R(x,y,S_1)] \wedge \neg(\bigvee_{j=1}^{m_-} \mathcal{R}[\phi_j^-(x,y,\alpha_n)])\} \wedge \Phi_1(y)[S_0]$$

$$\equiv \quad \exists y.\{\bigvee_{i=1}^{m_+} \mathcal{R}[\epsilon[\rho[\phi_i^+(x,y,\alpha_n)]]] \vee \mathcal{R}[R(x,y,S_1)] \wedge \neg(\bigvee_{j=1}^{m_-} \mathcal{R}[\epsilon[\rho[\phi_j^-(x,y,\alpha_n)]]])\}$$
$$\wedge \Phi_1(y)[S_0]$$

$$\equiv \quad \exists y.\{\bigvee_{i=1}^{m_+} (\nu_{i,n}^+(x) \wedge \eta_{i,n}^+(y)) \vee \mathcal{R}[R(x,y,S_1)] \wedge$$
$$(\bigwedge_{j=1}^{m_-} (\neg\nu_{j,n}^-(x) \vee \neg\eta_{j,n}^-(y)))\} \wedge \Phi_1(y)[S_0]$$

$$\equiv \quad \exists y.\{\bigvee_{i=1}^{m_+} (\nu_{i,n}^+(x) \wedge \eta_{i,n}^+(y)) \vee (\bigvee_{i=1}^{m_+} \mathcal{R}[\phi_i^+(x,y,\alpha_{n-1})] \vee \mathcal{R}[R(x,y,S_2)] \wedge$$
$$\neg(\bigvee_{j=1}^{m_-} \mathcal{R}[\phi_j^-(x,y,\alpha_{n-1})])) \wedge (\bigwedge_{j=1}^{m_-} (\neg\nu_{j,n}^-(x) \vee \neg\eta_{j,n}^-(y)))\} \wedge \Phi_1(y)[S_0]$$

$$\equiv \quad \exists y.\{\bigvee_{i=1}^{m_+} (\nu_{i,n}^+(x) \wedge \eta_{i,n}^+(y)) \vee (\bigvee_{i=1}^{m_+} (\nu_{i,n-1}^+(x) \wedge \eta_{i,n-1}^+(y)) \vee \mathcal{R}[R(x,y,S_2)] \wedge$$
$$(\bigwedge_{j=1}^{m_-} (\neg\nu_{j,n}^-(x) \vee \neg\eta_{j,n}^-(y)))) \wedge (\bigwedge_{j=1}^{m_-} (\neg\nu_{j,n}^-(x) \vee \neg\eta_{j,n}^-(y)))\} \wedge \Phi_1(y)[S_0]$$

$$\equiv \quad \cdots\cdots \text{(for each } 1 \leq l \leq n, \text{ let } \gamma_l^+ = \bigvee_{i=1}^{m_+} (\nu_{i,l}^+(x) \wedge \eta_{i,l}^+(y))$$
$$\text{and let } \gamma_l^- = \bigwedge_{j=1}^{m_-} (\neg\nu_{j,l}^-(x) \vee \neg\eta_{j,l}^-(y)))$$

$$\equiv \quad \exists y.\{\gamma_n^+ \vee (\gamma_{n-1}^+ \vee (\cdots \vee (\gamma_1^+ \vee R(x,y,S_0) \wedge \gamma_1^-) \wedge \cdots) \wedge \gamma_{n-1}^-) \wedge \gamma_n^-\}$$
$$\wedge \Phi_1(y)[S_0] \tag{15}$$

$$\equiv \quad \{\exists y.(\gamma_n^+ \vee \gamma_{n-1}^+ \wedge \gamma_n^- \vee \cdots \vee \gamma_1^+ \wedge \bigwedge_{i=2}^{n} \gamma_i^- \vee R(x,y) \wedge \bigwedge_{i=1}^{n} \gamma_i^-) \wedge \Phi_1(y)\}[S_0] \tag{16}$$

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

$$\equiv \quad \{\exists y.(\bigvee_{i=1}^{u} \Phi_{S,i}(x,y)) \wedge \Phi_1(y)\}[S_0] \qquad \text{for some index } u \tag{17}$$

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

$$\equiv \quad \{\bigvee_{i=1}^{u} (\exists y.\Phi_{S,i}(x,y) \wedge \Phi_1(y))\}[S_0] \tag{18}$$

$$\equiv \quad \{\bigvee_{i=1}^{u} \Psi_{S,i}(x)\}[S_0] \tag{19}$$

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 cases (1-16,1'-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 (14) into a sort of disjunctive normal form (DNF) (from Step (14) to Step (17)) 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 17 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 (16) . For instance, we perform the distributive law over $\gamma_{n-1}^+ \wedge \gamma_n^-$, and obtain

$$\bigvee_{i=1..m_+,k=1..2^{m_-}} \nu_{i,n-1}^+(x) \wedge \eta_{i,n-1}^+(y) \wedge \bigwedge_{j\in N_k} \neg\nu_{j,n}^-(x) \wedge \bigwedge_{l\in \overline{N_k}} \neg\eta_{l,n}^-(y),$$

where $N_k \subseteq \{1,2,\cdots,m_-\}$ ($1 \le k \le 2^{m_-}$) 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 (17) to Step (18)). Finally, we push $\exists y$ inside into each conjunctive clause and minimize the scope of each quantifier $\exists y$ (from Step(17) to Step (19)). In Step (17), 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 $\nu_{S,i}(x) \in FO_{DL}^x$ and some $\eta_{S,i}(y) \in FO_{DL}^y$. From Step (18) to Step (19), 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 $\nu_{S,i}(x) \wedge (\exists y.\eta_{S,i}(y) \wedge \Phi_1(y))$; 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 (19), with S_0 suppressed, is in FO_{DL}^x, and we denote the formula (with S_0 suppressed) as Φ_W.

Now we estimate the size of Φ_W when R is a fluent according to the way it is constructed above. First, for any $n \ge 0$ and any situation S where $sitLength(S) = n$, we denote the size of the DNF formula that is equivalent to $\mathcal{R}[R(x,y,S)]$, constructed specifically according to the above steps (14-17), as $g(n)$. Note that for each $1 \le i \le m_+$, $1 \le j \le m_-$ and $1 \le l \le n$, $size(\nu_{i,l}^+(x))$, ($size(\eta_{i,l}^+(y))$, $size(\nu_{j,l}^-(x))$ $size(\eta_{j,l}^-(y))$, respectively) is no more than $h+2$ according to the above for cases (1-16,1'-16') in Table 12. Moreover, according to the definition of function $size()$ in Section 5.2, 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 19, $f(m,n) \le (f(m-3,n)+3)g(n)$, where $f(m-3,n) = size(\Phi_1(y))$. Below, we show that $g(n) \le c_1 2^{nh}$, where $c_1 = (2(h+3)+(h+4)h^2+2)2^{2h}$. Moreover, we can perform a similar estimation for the case when $W^{[-S]}$ is of the form $\forall y.R(x,y) \supset W_1(y)$.

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

We can perform a similar estimation for the case when $W^{[-S]}$ is in FO^y_{DL}. Overall, under any case, we have

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

where constant numbers $c_2 = log_2 c_1$ and $c_1 = (2(h+3) + (h+4)h^2 + 2)2^{2h}$. 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 (2001) A survey of temporal extensions of description logics. Annals of Mathematics and Artificial Intelligence 30(1-4)

[2] Baader F, Lutz C, Miliĉić M, Sattler U, Wolter F (2005) Integrating description logics and action formalisms: First results. In: Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05), Pittsburgh, PA, USA, pp 572–577, extended version is available as LTCS-Report-05-02 at http://lat.inf.tu-dresden.de/research/reports.html

[3] Baader F, Milicic M, Lutz C, Sattler U, Wolter F (2005) Integrating description logics and action formalisms for reasoning about web services. LTCS-Report LTCS-05-02, Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, Germany

[4] Baader F, Calvanese D, McGuinness D, Nardi D, Patel-Schneider PF (eds) (2007) The Description Logic Handbook: Theory, Implementation, and Applications. Second edition, Cambridge University Press

[5] van Benthem J (1976) Modal Correspondence Theory, PhD thesis. Mathematisch Instituut & Instituut voor Grondslagenonderzoek, University of Amsterdam

[6] van Benthem J (1983) Modal logic and and classical logic. Bibliopolis

[7] Berardi D, Calvanese D, Giacomo GD, Lenzerini M, Mecella M (2003) e-service composition by description logics based reasoning. In: Calvanese D, Giacomo GD, Franconi E (eds) Proceedings of the 2003 International Workshop in Description Logics (DL-2003), Rome, Italy

[8] Blackburn P, van Benthem J (2007) Modal logic: a semantic perspective. In: Blackburn P, van Benthem J, Wolter F (eds) Handbook of Modal Logic, Elsevier Science, pp 1–84

[9] Borgida A (1996) On the relative expressiveness of description logics and predicate logics. Artificial Intelligence 82(1-2):353–367

[10] Calvanese D, Franconi E, Haarslev V, (2007) 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, CEUR-WS.org

[11] Calvanese D, Giacomo GD, Lenzerini M, Rosati R (2007) Actions and programs over description logic ontologies. In: [10]

[12] Castilho MA, Herzig A, Varzinczak IJ (2002) It depends on the context! a decidable logic of actions and plans based on a ternary dependence relation. In: Benferhat S, Giunchiglia E (eds) NMR, pp 343–348

[13] Chang L, Lin F, Shi Z (2007) A dynamic description logic for representation and reasoning about actions. In: Zhang Z, Siekmann JH (eds) KSEM, Springer, Lecture Notes in Computer Science, vol 4798, pp 115–127

[14] Chang L, Shi Z, Qiu L, Lin F (2007) Dynamic description logic: Embracing actions into description logic. In: [10]

[15] Demolombe R (2003) Belief change: from situation calculus to modal logic. Journal of Applied Non-Classical Logics 13(2):187–198

[16] Demolombe R, Herzig A, Varzinczak IJ (2003) Regression in modal logic. Journal of Applied Non-Classical Logics 13(2):165–185

[17] Drescher C, Thielscher M (2007) Integrating action calculi and description logics. In: Hertzberg J, Beetz M, Englert R (eds) KI-2007, Springer, Lecture Notes in Computer Science, vol 4667, pp 68–83

[18] Finzi A, Pirri F, Reiter R (2000) Open world planning in the situation calculus. In: AAAI/IAAI, AAAI Press / The MIT Press, pp 754–760

[19] Fischer MJ, Ladner RE (1979) Propositional dynamic logic of regular programs. Journal of Computer and System Sciences 18(2):194–211

[20] Gabbay D (1981) 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, Reidel, pp 91–117

[21] Gabbay DM, Shehtman VB (1998) Products of modal logics, part 1. Logic Journal of the IGPL 6(1):73–146

[22] Gabbay DM, Shehtman VB (2000) Products of modal logics. part 2: Relativised quantifiers in classical logic. Logic Journal of the IGPL 8(2)

[23] Gabbay DM, Shehtman VB (2002) Products of modal logics. part 3: Products of modal and temporal logics. Studia Logica 72(2):157–183

[24] Giacomo GD, Lenzerini M (1995) PDL-based framework for reasoning about actions. In: Gori M, Soda G (eds) AI*IA, Springer, Lecture Notes in Computer Science, vol 992, pp 103–114

[25] Giacomo GD, Iocchi L, Nardi D, Rosati R (1999) A theory and implementation of cognitive mobile robots. Journal of Logic and Computation 9(5):759–785

[26] Giacomo GD, Lenzerini M, Poggi A, Posati R (2006) 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), AAAI Press, Buston, US, pp 1271–1276

[27] Gil Y (2005) Description logics and planning. AI Magazine 26(2):73–84

[28] Grädel E, Kolaitis PG, Vardi MY (1997) On the decision problem for two-variable first-order logic. Bulletin of Symbolic Logic 3:53–69

[29] Grädel E, Otto M, Rosen E (1997) Two-variable logic with counting is decidable. In: Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science (LICS'97), Warsaw, Poland, pp 306–317

[30] Grüninger M (2004) Ontology of the process specification language. In: Staab S, Studer R (eds) Handbook on Ontologies, Springer, pp 575–592

[31] Grüninger M, Menzel C (2003) The process specification language (PSL): Theory and applications. AI Magazine 24(3):63–74

[32] Gu Y, Soutchanski M (2007) Decidable reasoning in a modified situation calculus. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07), Hyderabad, India, pp 1891–1897, http://www.cs.ryerson.ca/ mes/publications/DecidableSitcalc_ijcai07.pdf

[33] Harel D, Kozen D, Tiuryn J (2000) Dynamic Logic. THe MIT Press

[34] Hemaspaandra E (1996) The price of universality. Notre Dame Journal of Formal Logic 37(2):174–203

[35] Horrocks I, Sattler U (2001) Ontology reasoning in the SHOQ(D) description logic. In: In Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001, Morgan Kaufmann, pp 199–204

[36] Horrocks I, Patel-Schneider P, van Harmelen F (2003) From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics 1(1):7–26

[37] Hustadt U, de Nivelle H, Schmidt RA (2000) Resolution-based methods for modal logics. Logic Journal of the IGPL 8(3)

[38] Hustadt U, Schmidt RA, Georgieva L (2004) A survey of decidable first-order fragments and description logics. Journal on Relational Methods in Computer Science 1:251–276

[39] Kemke C (2003) 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, Springer, Lecture Notes in Computer Science, Volume 2671, June 11-13, Halifax, Canada, pp 458–465

[40] Kurucz A (2007) Combining modal logics. In: Blackburn P, van Benthem J, Wolter F (eds) Handbook of Modal Logic, Elsevier Science, pp 869–924

[41] Levesque H, Reiter R, Lespérance Y, Lin F, Scherl R (1997) GOLOG: A logic programming language for dynamic domains. Journal of Logic Programming 31:59–84

[42] Lin F (2004) Discovering state invariants. In: Dubois D, Welty CA, Williams MA (eds) KR, AAAI Press, pp 536–544

[43] Lin F, Reiter R (1994) State constraints revisited. J Log Comput 4(5):655–678

[44] Lin F, Reiter R (1997) How to progress a database. Artificial Intelligence 92:131–167

[45] Liu H, Lutz C, Milicic M, Wolter F (2006) Reasoning about actions using description logics with general tboxes. In: Fisher M, van der Hoek W, Konev B, Lisitsa A (eds) JELIA, Springer, Lecture Notes in Computer Science, vol 4160, pp 266–279

[46] Liu H, Lutz C, Milicic M, Wolter F (2006) 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), AAAI Press, pp 46–56

[47] Liu Y, Levesque HJ (2005) 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), Edinburgh, Scotland, pp 522–527

[48] Lutz C, Sattler U (2000) The complexity of reasoning with boolean modal logics. In: Wolter F, Wansing H, de Rijke M, Zakharyaschev M (eds) Advances in Modal Logic, World Scientific, pp 329–348

[49] Lutz C, Sattler U, Wolter F (2001) 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), Stanford, California, USA, pp 66–75

[50] McCarthy J (1959) 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 [53], London, U.K., pp 77–84

[51] McCarthy J (1963) Situations, actions and causal laws. Tech. rep., Stanford University

[52] McCarthy J (1986) Applications of circumscription to formalizing common sense knowledge. Artificial Intelligence 28:89–116

[53] McCarthy J (1990) Formalization of common sense: papers by John McCarthy edited by V. Lifschitz. Ablex, Norwood, N.J.

[54] McCarthy J (2002) Actions and other events in situation calculus. In: Eighth International Conference on Principles of Knowledge Representation and Reasoning (KR2002), Toulouse, France, available at http://www-formal.stanford.edu/jmc/sitcalc.html

[55] McCarthy J, Hayes P (1969) Some philosophical problems from the standpoint of artificial intelligence. In: Meltzer B, Michie D (eds) Machine Intelligence, vol 4, Edinburgh University Press, Reprinted in [53], pp 463–502

[56] McIlraith S, Son T (2002) Adapting Golog for composition of semantic web services. In: Fensel D, Giunchiglia F, McGuinness D, Williams MA (eds) Proceedings of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002), Morgan Kaufmann, April 22-25, Toulouse, France, pp 482–493

[57] McIlraith SA (2000) Integrating actions and state constraints: A closed-form solution to the ramification problem (sometimes). Artif Intell 116(1-2):87–121

[58] Milicic M (2007) Complexity of planning in action formalisms based on description logics. In: Dershowitz N, Voronkov A (eds) LPAR, Springer, Lecture Notes in Computer Science, vol 4790, pp 408–422

[59] Milicic M (2007) Planning in action formalisms based on DLs: First results. In: [10]

[60] Morgenstern L, Riecken D (2005) SNAP: An action-based ontology for e-commerce reasoning. In: Formal Ontologies Mett Industry, Proceedings of the 1st International Workshop FOMI 2005, June 9-10, Verona, Italy

[61] Narayanan S, McIlraith S (2003) Analysis and simulation of web services. Computer Networks 42:675–693

[62] de Nivelle H, Pratt-Hartmann I (2001) A resolution-based decision procedure for the two-variable fragment with equality. In: R Goré AL, Nipkow T (eds) IJCAR'01: Proceedings of the First International Joint Conference on Automated Reasoning, Springer-Verlag, LNAI, V. 2083, London, UK, pp 211–225

[63] Ohlbach HJ, Nonnengart A, de Rijke M, Gabbay DM (2001) Encoding two-valued nonclassical logics in classical logic pp 1403–1486

[64] Pacholski L, Szwast W, Tendera L (1997) Complexity of two-variable logic with counting. In: Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science (LICS-97), A journal version: SIAM Journal on Computing, v 29(4), 1999, p. 1083–1117, Warsaw, Poland, pp 318–327

[65] Pacholski L, Szwast W, Tendera L (2000) Complexity results for first-order two-variable logic with counting. SIAM Journal on Computing 29(4):1083–1117

[66] Pirri F, Reiter R (1999) Some contributions to the metatheory of the situation calculus. Journal of the ACM 46(3):325–364

[67] Pratt VR (1978) A practical decision method for propositional dynamic logic: Preliminary report. In: STOC, ACM, pp 326–337

[68] Pratt-Hartmann I (2005) Complexity of the two-variable fragment with counting quantifiers. Journal of Logic, Lang and Inf 14(3):369–395, DOI http://dx.doi.org/10.1007/s10849-005-5791-1

[69] Prendinger H, Schurz G (1996) Reasoning about action and change: A dynamic logic approach. Journal of Logic, Language and Information 5(2):209–245

[70] Reiter R (2001) Knowledge in Action: Logical Foundations for Describing and Implementing Dynamical Systems. The MIT Press

[71] Schiffel S, Thielscher M (2006) Reconciling situation calculus and fluent calculus. In: AAAI, AAAI Press

[72] Schild K (1991) A correspondence theory for terminological logics: Preliminary report. In: In Proc. of IJCAI-91, pp 466–471

[73] Schmidt RA, Tishkovsky D (2007) Deciding albo with tableau. In: [10]

[74] Schmidt RA, Tishkovsky D (2008) A general tableau method for deciding description logics, modal logics and related first-order fragments. In: Armando A, Baumgartner P, Dowek G (eds) IJCAR, Springer, Lecture Notes in Computer Science, vol 5195, pp 194–209

[75] Schmidt-SchaubßM, Smolka G (1991) Attributive concept descriptions with complements. Artif Intell 48(1):1–26, DOI http://dx.doi.org/10.1016/0004-3702(91)90078-X

[76] Shanahan M (1997) Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia. The MIT Press

[77] Shirazi A, Amir E (2005) First-order logical filtering. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), pp 589–595

[78] Spaan E (1993) Complexity of modal logics. PhD thesis, Department of Mathematics and Computer Science, University of Amsterdam

[79] Tobies S (1999) A nexptime-complete description logic strictly contained in C^2. In: Flum J, Rodríguez-Artalejo M (eds) CSL, Springer, Lecture Notes in Computer Science, vol 1683, pp 292–306

[80] Tobies S (2000) The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. J of Artificial Intelligence Research 12:2000

[81] Tobies S (2001) Complexity results and practical algorithms for logics in knowledge representation. PhD Thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany

[82] Vassos S, Lakemeyer G, Levesque HJ (2008) 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), Sydney, Australia

[83] Winslett MS (1990) Updating logical databases. The Academic Press

[84] Wolter F, Zakharyaschev M (2000) Dynamic description logics. In: Advances in Modal Logic, 2nd workshop held in Uppsala, Sweden, 1998, CSLI Publications, pp 431–446

[85] Zolin E (2007) Description logic complexity navigator. Available at HTTP:http://www.cs.man.ac.uk/ ezolin/dl/