Complexity of Planning in Action Formalisms Based on Description Logics

Maja Miličić *

Institut für Theoretische Informatik
TU Dresden, Germany
maja@tcs.inf.tu-dresden.de

Abstract. In this paper, we continue the recently started work on integrating action formalisms with description logics (DLs), by investigating planning in the context of DLs. We prove that the plan existence problem is decidable for actions described in fragments of $\mathcal{ALCQIO}$. More precisely, we show that its computational complexity coincides with the one of projection for DLs between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$ if operators contain only unconditional post-conditions. If we allow for conditional post-conditions, the plan existence problem is shown to be in 2-ExpSpace.

1 Introduction

Description Logics (DLs) are a well-known family of knowledge representation formalisms that may be viewed as decidable fragments of first-order logic (FO). The main strength of DLs is that they offer a nice compromise between expressiveness and complexity of reasoning [1].

The idea to investigate action formalisms based on description logics was inspired by the expressiveness gap between existing action formalisms: they were either based on FO logic and undecidable, like the Situation Calculus [14] and the Fluent Calculus [17], or decidable but only propositional.

First results on integrating DLs with action formalisms from [2] show that reasoning remains decidable even if an action formalism is based on the expressive DL $\mathcal{ALCQIO}$. In [2], ABoxes give incomplete descriptions of the current state of the world, and describe the pre- and post-conditions of actions. Domain constraints are captured by *acyclic* TBoxes, and post-conditions may contain only atomic concept and role assertions. This formalism is in fact a decidable fragment of SitCalc. It is shown in [2] that the projection and executability problem for actions can be reduced to standard DL reasoning problems. Further work in this line [11, 10] treat the problem of computing ABox updates and the ramification problem induced by GCIs.

However, in the mentioned DL-action-framework, planning, an important reasoning task, has not yet been considered. Intuitively, given an initial state $\mathcal{A}$, final state Γ and a finite set of actions Op, the *plan existence problem* is the following: "is there a plan (a sequence of actions from Op) which transforms

* The author is supported by the EU project TONES

$\mathcal{A}$ into a state where Γ is satisfied?". It is known that, already in the propositional case, planning is a hard problem. For example, the plan existence problem for propositional STRIPS-style actions with complete state descriptions is PSPACE-complete [4, 7], while it is EXPSPACE-complete for conformant planning (incomplete state descriptions) where actions have conditional post-conditions [8, 15].

The planning problem in DL action formalisms is not only interesting from the theoretical point of view. It is well known that the semantic web ontology language OWL [9] is based on description logics; thus actions described in DLs can be viewed as simple semantic web services. In this context, planning is a very important reasoning task as it supports, e.g., web service discovery which is needed for an automatic service execution.

This paper is, to our best knowledge, the first attempt to formally define the planning problem in a DL fragment of SitCalc. We investigate the computational complexity of the plan existence problem for the description logics "between" $\mathcal{ALC}$ and $\mathcal{ALCQIO}$. By using a compact representation of possible states obtained by action application, we show that, if we allow only for actions with unconditional post-conditions, in these logics the plan existence problem is decidable, and of the same computational complexity as projection. If conditional post-conditions are allowed, we show that the plan existence problem is in 2-EXPSPACE. In the last section we discuss possible ways of developing practical planning algorithms for DLs.

2 The Description Logic $\mathcal{ALCQIO}$

The action formalism used in this paper is not restricted to a particular DL. However, for our complexity results we consider the DL $\mathcal{ALCQIO}$ and a number of its sublanguages. The reason for choosing this family of DLs is that they are very expressive, but nevertheless admit practical reasoning. Moreover, $\mathcal{ALCQIO}$ forms the core of OWL-DL, the description logic variant of OWL. As discussed in [2], the additional OWL-DL constructors can be easily added, except for transitive roles which lead to semantic and computational problems. Indeed, DLs from this family underlie highly optimized DL systems such as FaCT++, RacerPro, and Pellet.

In DL, concepts are inductively defined with the help of a set of *constructors*, starting with a set $\mathsf{N_C}$ of *concept names*, a set $\mathsf{N_R}$ of *role names*, and a set $\mathsf{N_I}$ of *individual names*. The constructors determine the expressive power of the DL. Table 1 shows a minimal set of constructors from which all constructors of $\mathcal{ALCQIO}$ can be defined. The first row contains the only role constructor: in $\mathcal{ALCQIO}$, a *role* s is either a role name $r \in \mathsf{N_R}$ or the inverse r^- of a role name r. *Concepts* of $\mathcal{ALCQIO}$ are formed using the remaining constructors shown in Table 1, where r is a role, n a positive integer, and a an individual name. Using these constructors, several other constructors can be defined as abbreviations:

- $C \sqcup D := \neg(\neg C \sqcap \neg D)$ (disjunction),
- $\top := A \sqcup \neg A$ for a concept name A (top-concept),

Name	Syntax	Semantics
inverse role	r^-	$\{(y,x) \mid (x,y) \in r^{\mathcal{I}}\}$
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
at-least restriction	$(\geqslant n\ s\ C)$	$\{x \in \Delta^{\mathcal{I}} \mid \mathit{card}\{y \in C^{\mathcal{I}} \mid (x,y) \in s^{\mathcal{I}}\} \geq n\}$
nominal	$\{a\}$	$\{a^{\mathcal{I}}\}$

Table 1. Syntax and semantics of $\mathcal{ALCQIO}$.

- $\exists s.C := (\geqslant 1\ s\ C)$ (existential restriction),
- $\forall s.C := \neg(\exists s.\neg C)$ (value restriction),
- $(\leqslant n\ s\ C) := \neg(\geqslant (n+1)\ s\ C)$ (at-most restriction).

The DL that allows for negation, conjunction, and value restrictions is called $\mathcal{ALC}$. The availability of additional constructors is indicated by concatenating the corresponding letter: $\mathcal{Q}$ stands for number restrictions; $\mathcal{I}$ stands for inverse roles, and $\mathcal{O}$ for nominals. This explains the name $\mathcal{ALCQIO}$ for our DL, and also allows us to refer to sublanguages in a simple way.

The semantics of $\mathcal{ALCQIO}$-concepts and roles is defined in terms of an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$. The domain $\Delta^{\mathcal{I}}$ of $\mathcal{I}$ is a non-empty set of individuals and the interpretation function $\cdot^{\mathcal{I}}$ maps each concept name $A \in \mathsf{N_C}$ to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name $r \in \mathsf{N_R}$ to a binary relation $r^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$, and each individual name $a \in \mathsf{N_I}$ to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The extension of $\cdot^{\mathcal{I}}$ to arbitrary concepts and roles is inductively defined, as shown in the third column of Table 1. Here, the function *card* yields the cardinality of the given set. Note that the third column of Table 1 suggests a straightforward translation of DL concepts into first-order formulas with one free variable, as explicated e.g. in [1].

A *concept definition* is an identity of the form $A \doteq C$, where A is a concept name and C an $\mathcal{ALCQIO}$-concept. A *TBox* $\mathcal{T}$ is a finite set of concept definitions with unique left-hand sides. Concept names occurring on the left-hand side of a definition of $\mathcal{T}$ are called *defined in* $\mathcal{T}$ whereas the others are called *primitive in* $\mathcal{T}$. The TBox $\mathcal{T}$ is *acyclic* iff there are no cyclic dependencies between the definitions [1]. The *semantics* of TBoxes is defined in the obvious way: the interpretation $\mathcal{I}$ is a *model* of the TBox $\mathcal{T}$ ($\mathcal{I} \models \mathcal{T}$) iff it satisfies all its definitions, i.e., $A^{\mathcal{I}} = C^{\mathcal{I}}$ holds for all $A \doteq C$ in $\mathcal{T}$. In the case of acyclic TBoxes, any interpretation of the primitive concepts and of the role names can uniquely be extended to a model of the TBox [12].

An *ABox assertion* is of the form $C(a)$, $r(a,b)$ or $\neg r(a,b)$, where $a, b \in \mathsf{N_I}$, C is a concept, and r a role name.[1] An *ABox* is a finite set of ABox assertions. The interpretation $\mathcal{I}$ is a *model* of the ABox $\mathcal{A}$ ($\mathcal{I} \models \mathcal{A}$) iff it satisfies all its assertions, i.e., $a^{\mathcal{I}} \in C^{\mathcal{I}}$ ($(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$, $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin r^{\mathcal{I}}$) for all assertions $C(a)$ ($r(a,b)$, $\neg r(a,b)$) in $\mathcal{A}$. If φ is an assertion, then we write $\mathcal{I} \models \varphi$ to indicate that $\mathcal{I}$ satisfies φ.

[1] Disallowing inverse roles in ABox assertions is not a restriction since $r^-(a,b)$ can be expressed by $r(b,a)$.

Various reasoning problems are considered for DLs. For the purpose of this paper, it suffices to introduce ABox consistency and ABox consequence: the ABox $\mathcal{A}$ is *consistent* w.r.t. the TBox $\mathcal{T}$ iff there exists an interpretation $\mathcal{I}$ that is a model of both $\mathcal{T}$ and $\mathcal{A}$; the ABox assertion φ is a *consequence* of $\mathcal{A}$ w.r.t. $\mathcal{T}$ (written $\mathcal{T}, \mathcal{A} \models \varphi$) iff every model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ is also a model of φ.

3 Action Formalism

In this section, we present (a slightly extended version of) the action formalism from [2]. Since we focus on planning in this paper, the central notion become parameterized actions (operators), rather than ground actions like in [2].

The main ingredients of our framework are operators and actions (as defined below), ABoxes describing the current knowledge about the state of affairs in the application domain, and acyclic TBoxes for describing general knowledge about the application domain similar to state constraints in the SitCalc and Fluent Calculus.

Definition 1 (Action, operator). *Let* $\mathsf{N_X}$ *be a countably infinite sets of* variables*, disjoint with* $\mathsf{N_C}$*,* $\mathsf{N_R}$ *and* NI*. Moreover, let* $\mathcal{T}$ *be an acyclic TBox. A* primitive literal for $\mathcal{T}$ *is an ABox assertion*

$$A(a), \neg A(a), r(a,b), \text{ or } \neg r(a,b)$$

with A *a primitive concept name in* $\mathcal{T}$*,* r *a role name, and* $a, b \in \mathsf{N_I}$*. An* atomic atomic $\alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post})$ *for* $\mathcal{T}$ *consists of*

- *a finite set* pre *of ABox assertions, the* pre-conditions*;*
- *a finite set* occ *of* occlusions *of the form* $A(a)$ *or* $r(a,b)$*, with* A *primitive concept in* $\mathcal{T}$*,* r *role name, and* $a, b \in \mathsf{N_I}$*;*
- *a finite set* post *of* conditional post-conditions *of the form* φ/ψ*, where* φ *is an ABox assertion and* ψ *is a* primitive literal for $\mathcal{T}$*.*

A composite action *for* $\mathcal{T}$ *is a finite sequence* $\alpha_1, \ldots, \alpha_k$ *of atomic actions for* $\mathcal{T}$*. An* operator for $\mathcal{T}$ *is a parameterized atomic action for* $\mathcal{T}$*, i.e., an action in which definition variables from* $\mathsf{N_X}$ *may occur in place of individual names.*

We call post-conditions of the form $\top(t)/\psi$ *unconditional* and write just ψ instead.

Applying an action changes the state of affairs, and thus transforms an interpretation $\mathcal{I}$ into an interpretation $\mathcal{I}'$. Intuitively, the pre-conditions specify under which conditions the action is applicable. The post-condition φ/ψ says that, if φ is true in the original interpretation $\mathcal{I}$, then ψ is true in the interpretation $\mathcal{I}'$ obtained by applying the action. The rôle of occlusions is to indicate those primitive literals that can change arbitrarily.

When defining the semantics of actions, we assume that states of the world correspond to interpretations. Thus, the semantics of actions can be defined by means of a transition relation on interpretations. We do not give semantics of

operators explicitly and assume that they are grounded before their application. Let $\mathcal{T}$ be an acyclic TBox, $\alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post})$ an action for $\mathcal{T}$, and $\mathcal{I}$ an interpretation. For each primitive concept name A and role name r, set:

$$\begin{aligned}
A^+ &:= \{b^{\mathcal{I}} \mid \varphi/A(b) \in \mathsf{post} \wedge \mathcal{I} \models \varphi\} \\
A^- &:= \{b^{\mathcal{I}} \mid \varphi/\neg A(b) \in \mathsf{post} \wedge \mathcal{I} \models \varphi\} \\
I_A &:= (\Delta^{\mathcal{I}} \setminus \{b^{\mathcal{I}} \mid A(b) \in \mathsf{occ}\}) \cup (A^+ \cup A^-) \\
r^+ &:= \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid \varphi/r(a,b) \in \mathsf{post} \wedge \mathcal{I} \models \varphi\} \\
r^- &:= \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid \varphi/\neg r(a,b) \in \mathsf{post} \wedge \mathcal{I} \models \varphi\} \\
I_r &:= ((\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid r(a,b) \in \mathsf{occ}\}) \cup (r^+ \cup r^-)
\end{aligned}$$

The transition relation on interpretations should ensure that $A^+ \subseteq A^{\mathcal{J}}$ and $A^- \cap A^{\mathcal{J}} = \emptyset$ if $\mathcal{J}$ is the result of applying α in $\mathcal{I}$. It should also ensure that nothing else changes, with the possible exception of the occluded literals. Intuitively, I_A and I_r describe those parts of the model that are *not* exempted from this restriction by the presence of an occlusion. Since we restrict our attention to acyclic TBoxes, for which the interpretation of defined concepts is uniquely determined by the interpretation of primitive concepts and role names, it is not necessary to consider defined concepts when defining the transition relation.

Definition 2. *Let $\mathcal{T}$ be an acyclic TBox, $\alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post})$ an atomic action for $\mathcal{T}$, and $\mathcal{I}, \mathcal{I}'$ models of $\mathcal{T}$ respecting the unique name assumption (UNA) on individual names and sharing the same domain and interpretation of all individual names. We say that α* may transform *$\mathcal{I}$ to $\mathcal{I}'$ ($\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha} \mathcal{I}'$) iff, for each primitive concept A and role name r, we have*

$$\begin{aligned}
A^+ \cap A^- = \emptyset \text{ and } r^+ \cap r^- &= \emptyset \\
A^{\mathcal{I}'} \cap I_A &= ((A^{\mathcal{I}} \cup A^+) \setminus A^-) \cap I_A \\
r^{\mathcal{I}'} \cap I_r &= ((r^{\mathcal{I}} \cup r^+) \setminus r^-) \cap I_r.
\end{aligned}$$

The composite action $\alpha_1, \ldots, \alpha_k$ may transform *$\mathcal{I}$ to $\mathcal{I}'$ ($\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha_1,\ldots,\alpha_k} \mathcal{I}'$) iff there are models $\mathcal{I}_0, \ldots, \mathcal{I}_k$ of $\mathcal{T}$ with $\mathcal{I} = \mathcal{I}_0$, $\mathcal{I}' = \mathcal{I}_k$, and $\mathcal{I}_{i-1} \Rightarrow^{\mathcal{T}}_{\alpha_i} \mathcal{I}_i$ for $1 \leq i \leq k$.*

Note that the semantics is such that the changes are minimized w.r.t. the initial interpretations. Also note that this definition does not check whether the action is indeed executable, i.e., whether the pre-conditions are satisfied. It just says what the result of applying the action is, irrespective of whether it is executable or not. Since we use acyclic TBoxes to describe background knowledge, if occ is the empty set, there cannot exist more than one $\mathcal{I}'$ such that $\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha} \mathcal{I}'$. Thus, actions with empty occlusions are deterministic.

Like in [2], we assume that actions $\alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post})$ are *consistent with* $\mathcal{T}$ in the following sense: for every model $\mathcal{I}$ of $\mathcal{T}$, there exists $\mathcal{I}'$, such that $\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha} \mathcal{I}'$. It is not difficult to see that this is the case iff $\{\varphi_1/\psi, \varphi_2/\neg\psi\} \subseteq \mathsf{post}$ implies that the ABox $\{\varphi_1, \varphi_2\}$ is inconsistent w.r.t. $\mathcal{T}$.

Two standard reasoning problems about actions, *projection* and *executability*, are thoroughly investigated in [2] in the context of DLs. Executability is the problem of whether an action can be applied in a given situation, i.e. if preconditions are satisfied in the states of the world considered possible.

Formally, let $\mathcal{T}$ be an acyclic TBox, $\mathcal{A}$ an ABox, and let $\alpha_1, \ldots, \alpha_n$ be a composite action with $\alpha_i = (\mathsf{pre}_i, \mathsf{occ}_i, \mathsf{post}_i)$ atomic actions for $\mathcal{T}$ for $i = 1, \ldots, n$.

We say that $\alpha_1, \ldots, \alpha_n$ is *executable in $\mathcal{A}$ w.r.t. $\mathcal{T}$* iff the following conditions are true for all models $\mathcal{I}$ of $\mathcal{A}$ and $\mathcal{T}$:

- $\mathcal{I} \models \mathsf{pre}_1$
- for all i with $1 \leq i < n$ and all interpretations $\mathcal{I}'$ with $\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha_1,\ldots,\alpha_i} \mathcal{I}'$, we have $\mathcal{I}' \models \mathsf{pre}_{i+1}$.

Projection is the problem of whether applying an action achieves the desired effect, i.e., whether an assertion that we want to make true really holds after executing the action. Formally,the assertion φ is a *consequence of applying $\alpha_1, \ldots, \alpha_n$ in $\mathcal{A}$ w.r.t. $\mathcal{T}$* iff for all models $\mathcal{I}$ of $\mathcal{A}$ and $\mathcal{T}$ and for all $\mathcal{I}'$ with $\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha_1,\ldots,\alpha_n} \mathcal{I}'$, we have $\mathcal{I}' \models \varphi$.

In [2] it was shown that projection and executability are decidable for the logics between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$. More precisely, projection in $\mathcal{L}$ can be reduced to (in)consistency of an ABox relative to an acyclic TBox in $\mathcal{LO}$. The following theorem from [2] states that upper complexity bounds obtained in this way are optimal:

Theorem 1. *([2]) Projection and executability of composite actions are:*

(a) PSPACE*-complete in $\mathcal{ALC}$,$\mathcal{ALCO}$,$\mathcal{ALCQ}$, and $\mathcal{ALCQO}$;*
(b) EXPTIME*-complete in $\mathcal{ALCI}$ and $\mathcal{ALCIO}$;*
*(c) co-*NEXPTIME*-complete in $\mathcal{ALCQI}$ and $\mathcal{ALCQIO}$.*

Looking carefully at the reduction of projection in $\mathcal{L}$ to ABox inconsistency in $\mathcal{LO}$ from [2, 3], we conclude that the upper complexity bounds from Theorem 1 hold even for the "stronger" projection problem, namely the one where instead of a single ABox assertion φ, we have an ABox Γ. We will need this strengthened complexity result in the coming sections.

4 Planning Problem

We continue by defining the plan existence problem in the introduced framework. First we introduce a bit of notation. If o is an operator (for a TBox $\mathcal{T}$), we use $\mathsf{var}(o)$ to denote the set of variables in o. A substitution v for o is a mapping $v : \mathsf{var}(o) \to \mathsf{N_I}$. An action α that is obtained by applying a substitution v to o is denoted as $\alpha := o[v]$. Intuitively, the plan existence problem is: given an acyclic TBox $\mathcal{T}$ which describes the background knowledge, ABoxes $\mathcal{A}$ and Γ giving incomplete descriptions of the initial and the goal state, and a set of operators Op, is there a plan (sequence of actions obtained by instantiating operators

from Op) which "transforms" the stated described by $\mathcal{A}$ into a state where Γ is satisfied?

In this paper, we assume that operators can be instantiated with individuals from a finite set $\mathsf{Ind} \subset \mathsf{N_I}$. Moreover, we assume that $\mathcal{T}$, $\mathcal{A}$ and Γ contain only individuals from Ind (we say that they are based on Ind). For an operator o, we set $o[\mathsf{Ind}] := \{o[v] \mid v : \mathsf{var}(o) \to \mathsf{Ind}\}$ and for Op a set of operators, we set $\mathsf{Op}[\mathsf{Ind}] := \{o[\mathsf{Ind}] \mid o \in \mathsf{Op}\}$, i.e. $\mathsf{Op}[\mathsf{Ind}]$ is the set of all actions obtained by instantiating operators from Op with individuals from Ind. In the following definition, we formally introduce the notion of a planing task:

Definition 3 (Planning task). *A planning task is a tuple $\Pi = (\mathsf{Ind}, \mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma)$, where*

- Ind *is a finite set of individual names;*
- $\mathcal{T}$ *is an acyclic TBox based on* Ind*;*
- Op *is a finite set of atomic operators for $\mathcal{T}$;*
- $\mathcal{A}$ *(initial state) is an ABox based on* Ind*;*
- Γ *(goal) is an ABox based on* Ind.

A plan *in Π is a composite action $\alpha = \alpha_1, \ldots, \alpha_k$, such that $\alpha_i \in \mathsf{Op}[\mathsf{Ind}]$, $i = 1..k$. A plan $\alpha = \alpha_1, \ldots, \alpha_k$ in Π is a* solution *to the planning task Π iff:*

1. *α is executable in $\mathcal{A}$ w.r.t. $\mathcal{T}$; and*
2. *for all interpretations $\mathcal{I}$ and $\mathcal{I}'$ such that $\mathcal{I} \models \mathcal{A}, \mathcal{T}$ and $\mathcal{I} \Rightarrow_\alpha^\mathcal{T} \mathcal{I}'$, it holds that $\mathcal{I}' \models \Gamma$.*

Example 1. We illustrate the previous definition by the following example describing a (simplified) process of opening a bank account in the UK.

Let the set of individuals be defined as

$$\mathsf{Ind} = \{\mathsf{dirk}, \mathsf{uni\_liv}, \mathsf{yoga\_center}, \mathsf{UK}, \mathsf{el}, \mathsf{el}', \mathsf{l}, \mathsf{ba}\}.$$

The initial state - ABox $\mathcal{A}$ states that Dirk is a resident of the UK who has gotten two jobs – at the University of Liverpool and in the Yoga Center, but still does not hold a bank account in the UK.

$$\begin{aligned}\mathcal{A} := \{&\mathsf{resident}(\mathsf{dirk}, \mathsf{UK}), \mathsf{employs}(\mathsf{uni\_liv}, \mathsf{dirk}), \mathsf{employs}(\mathsf{yoga\_center}, \mathsf{dirk}),\\ &\mathsf{University}(\mathsf{uni\_liv}), \neg\exists\mathsf{holds}.(\mathsf{B\_acc} \sqcap \exists\mathsf{in}.\{\mathsf{UK}\})(\mathsf{dirk})\}\end{aligned}$$

Moreover, the set Op contains the operators for obtaining a lease, a letter from employer, and a bank account. The set of occlusions occ is empty for all three operators, so we will state only the sets of pre- and post-conditions pre and post.

- Suppose the pre-condition of obtaining a lease is that the customer x holds a letter from his employer. This is formalized by the operator $\mathsf{get\_Lease}$:

$$\begin{aligned}\mathsf{pre}:&\ \{\exists\mathsf{holds}.\mathsf{EmployerLetter}(x)\}\\ \mathsf{post}:&\ \{\mathsf{holds}(x, y), \mathsf{Lease}(y)\}\end{aligned}$$

– The operator get_Letter describes the process of an employee x getting a letter y from his employer z:

$$\begin{aligned}\mathsf{pre}: &\ \{\mathsf{employs}(z,x)\}\\ \mathsf{post}: &\ \{\mathsf{holds}(x,y), \mathsf{EmployerLetter}(y), \mathsf{signed}(z,y)\}\end{aligned}$$

– Suppose the pre-condition of opening a bank account is that the customer x is a resident in the UK and holds a proof of address. Moreover, suppose that, if x is rated as "reliable", then the bank account comes with a credit card, otherwise not. This service can be formalized by the following operator get_B_acc:

$$\begin{aligned}\mathsf{pre}: &\ \{\exists \mathsf{resident}.\{\mathsf{UK}\}(x), \exists \mathsf{holds}.\mathsf{Proof\_address}(x)\}\\ \mathsf{post}: &\ \{\mathsf{holds}(x,y), \mathsf{in}(y,\mathsf{UK}),\\ &\quad \mathsf{Reliable}(x)/\mathsf{B\_acc\_credit}(y),\\ &\quad \neg\mathsf{Reliable}(x)/\mathsf{B\_acc\_no\_credit}(y)\}\end{aligned}$$

The meaning of the concepts used in $\mathcal{A}$ and Op is defined in the following acyclic TBox $\mathcal{T}$:

$$\begin{aligned}\mathsf{Reliable} &\doteq \exists \mathsf{holds}.(\mathsf{B\_acc} \sqcap \mathsf{Good\_credit\_rating} \sqcap \exists \mathsf{in}.\{\mathsf{UK}\})\\ &\quad \sqcup \exists \mathsf{holds}.(\mathsf{EmployerLetter} \sqcap \exists \mathsf{signed}^{-}.\mathsf{University}\}\\ \mathsf{Proof\_address} &\doteq \mathsf{Electricity\_contract} \sqcup \mathsf{Lease}\\ \mathsf{B\_acc} &\doteq \mathsf{B\_acc\_credit} \sqcup \mathtt{B\_acc\_no\_credit}\end{aligned}$$

The first concept definition tells us that a person is rated as reliable if and only if he already holds a bank account in the UK with a good credit rating, or holds a letter stating that he is employed at the university. The second definition defines a proof of the address to be either en electricity contract or a lease, while the last one states that a bank account can come either with or without a credit card.

Finally, we have two goals, $\Gamma_1 = \{\exists \mathsf{holds}.(\mathsf{B\_acc} \sqcap \exists \mathsf{in}.\{\mathsf{UK}\})(\mathsf{dirk})\}$, requiring that Dirk holds a bank account in the UK, and a more ambitious one, $\Gamma_2 = \{\exists \mathsf{holds}.(\mathsf{B\_acc\_credit} \sqcap \exists \mathsf{in}.\{\mathsf{UK}\})(\mathsf{dirk})\}$, namely that Dirk holds a bank account in the UK with a credit card. We define corresponding planing tasks Π_1 and Π_2 as $\Pi_1 = (\mathsf{Ind}, \mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma_1)$ and $\Pi_2 = (\mathsf{Ind}, \mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma_2)$. It is not difficult to see that the plan:

$$\mathsf{get\_Letter}[x/\mathsf{dirk}, y/\mathsf{el}, z/\mathsf{yoga\_center}], \mathsf{get\_Lease}[x/\mathsf{dirk}, y/\mathsf{l}], \mathsf{get\_B\_acc}[x/\mathsf{dirk}, y/\mathsf{ba}]$$

is a solution to Π_1, but not Π_2, while the plan:

$$\mathsf{get\_Letter}[x/\mathsf{dirk}, y/\mathsf{el}', z/\mathsf{uni\_liv}], \mathsf{get\_Lease}[x/\mathsf{dirk}, y/\mathsf{l}], \mathsf{get\_B\_acc}[x/\mathsf{dirk}, y/\mathsf{ba}]$$

is a solution both to Π_1 and Π_2.

The *plan existence problem (PLANEX)*, c.f. [7], is the problem of whether a given planning task Π has a solution. If operators in Π contain conditional post-conditions, we will call it *conditional PLANEX*, and otherwise *unconditional PLANEX*.

5 Complexity of Planning: Unconditional Post-Conditions

In this section, we will focus on the plan existence problem in the case operators have only unconditional post-conditions. It turns out that unconditional PLANEX is not harder, at least in theory, than projection in the fragments of $\mathcal{ALCQIO}$ from Theorem 1.

Obviously, the plan existence problem is closely related to projection and executability. First we introduce some notation. Let $\mathcal{A}$ be an ABox, $\mathcal{T}$ an acyclic TBox, α a (possibly composite) action, and φ an ABox assertion. We will write $\mathcal{T}, \mathcal{A}^\alpha \models \varphi$ iff φ is a consequence of applying α in $\mathcal{A}$ w.r.t. $\mathcal{T}$. For an ABox $\mathcal{B}$, we write $\mathcal{T}, \mathcal{A}^\alpha \models \mathcal{B}$ iff $\mathcal{T}, \mathcal{A}^\alpha \models \varphi$ for all $\varphi \in \mathcal{B}$.

Let $\Pi = (\mathsf{Ind}, \mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma)$ be a planning task for which we want to decide if it has a solution. This means that we want to check if there is a sequence of actions from $\mathsf{Op}[\mathsf{Ind}]$ which transform the initial state (described by $\mathcal{A}$) into a state where goal Γ holds.

In the propositional case, planning is based on step-wise computation of the next state – which corresponds to computing updated ABoxes. However, in [11], it is shown that an updated ABox may be exponentially large in the size of the initial ABox and the update, which makes this approach unsuitable. We base our approach in this paper on the following observation: instead of computing a sequence of (exponentially large) updated ABoxes, it suffices to compute a sequence of *updates* which are applied to the initial ABox $\mathcal{A}$. Intuitively, these updates are lists of accumulated triggered post-conditions. Similarly, we keep track of accumulated occlusions. Thus, states of the search space can be compactly described as pairs: (occlusion, update).

We define the set of possible (negated) atomic changes as:

$$\mathcal{L}_{\mathsf{post}} := \{\psi, \neg\psi \mid \psi \in \mathsf{post},\ \alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post}),\ \alpha \in \mathsf{Op}[\mathsf{Ind}]\}$$

and the set of possible occlusions:

$$\mathcal{L}_{\mathsf{occ}} := \{\psi \mid \psi \in \mathsf{occ},\ \alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post}),\ \alpha \in \mathsf{Op}[\mathsf{Ind}]\}$$

An *update* for Π is a consistent subset of $\mathcal{L}_{\mathsf{post}}$. Let $\mathfrak{U}$ be a set of all updates for Π. Moreover, let $\mathfrak{O} := 2^{\mathcal{L}_{\mathsf{occ}}}$. Then $\mathfrak{O} \times \mathfrak{U}$ is our search space, the size of which $|\mathfrak{O}| \cdot |\mathfrak{U}|$ is exponential in the size of Π, since the sizes of $\mathcal{L}_{\mathsf{post}}$ and $\mathcal{L}_{\mathsf{occ}}$ are polynomial in Π. For a $\mathcal{U} \in \mathfrak{U}$, we set $\neg\mathcal{U} := \{\neg l \mid l \in \mathcal{U}\}$ and $\overline{\mathcal{U}} := \{l \mid l \in \mathcal{U} \cup \neg\mathcal{U} \text{ and } l \text{ positive}\}$.

Intuitively, $(\emptyset, \emptyset)$ represents the initial state, and all tuples $(\mathcal{O},\mathcal{U}) \in \mathfrak{O} \times \mathfrak{U}$ such that $\mathcal{T}, \mathcal{A}^{(\emptyset,\mathcal{O},\mathcal{U})} \models \Gamma$ represent goal states. In the next step, we define the transition relation "$\xrightarrow{\alpha}_{\mathcal{T},\mathcal{A}}$" on $\mathfrak{O} \times \mathfrak{U}$. Let $(\mathcal{O},\mathcal{U}),(\mathcal{O}',\mathcal{U}') \in \mathfrak{O} \times \mathfrak{U}$. We say that $(\mathcal{O},\mathcal{U}) \xrightarrow{\alpha} (\mathcal{O}',\mathcal{U}')$ iff:

(i) $\mathcal{O}' = (\mathcal{O} \cup \mathsf{occ}) \setminus \overline{\mathsf{post}}$
(ii) $\mathcal{U}' = (\mathcal{U} \setminus (\mathsf{occ} \cup \neg\mathsf{occ} \cup \neg\mathsf{post})) \cup \mathsf{post}$

Obviously, the relation "$\xrightarrow{\alpha}$" is functional for every α. In the following lemma, we show that "$\xrightarrow{\alpha}$" simulates "$\Rightarrow_\alpha^{\mathcal{T}}$" on the set $\mathfrak{O} \times \mathfrak{U}$. We omit the proof, which can be done by an easy induction.

Lemma 1. *Let $\Pi = (\mathsf{Ind}, \mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma)$ be a planning task and let $\alpha = \alpha_1, \ldots, \alpha_k$ be a plan in Π. Let $\mathcal{U}_0 = \mathcal{O}_0 := \emptyset$ and let $(\mathcal{O}_1,\mathcal{U}_1), \ldots, (\mathcal{O}_k,\mathcal{U}_k)$ be such such that*

$$(\mathcal{O}_0,\mathcal{U}_0) \stackrel{\alpha_1}{\rightarrow} (\mathcal{O}_1,\mathcal{U}_1) \cdots \stackrel{\alpha_k}{\rightarrow} (\mathcal{O}_k,\mathcal{U}_k)$$

Then the following holds:

(a) For all interpretations $\mathcal{I},\mathcal{I}'$ such that $\mathcal{I} \models \mathcal{A}$ and for all $1 \leq i \leq k$, we have that $\mathcal{I} \Rightarrow^{\mathcal{T}}_{\alpha_1,\ldots,\alpha_i} \mathcal{I}'$ iff $\mathcal{I} \Rightarrow^{\mathcal{T}}_{(\emptyset,\mathcal{O}_i,\mathcal{U}_i)} \mathcal{I}'$.
(b) $\mathcal{T}, \mathcal{A}^{(\emptyset,\mathcal{O}_i,\mathcal{U}_i)} \models \mathsf{pre}_{i+1}$ for all $i < k$ iff $\alpha_1, \ldots, \alpha_k$ is executable in $\mathcal{A}$ w.r.t. $\mathcal{T}$;

We now present a non-deterministic procedure which decides whether the planning task Π has a solution. The procedure searches for an executable sequence of actions from $\mathsf{Op}[\mathsf{Ind}]$ which transforms the initial state $S_0 = (\emptyset, \emptyset)$ into a state $\mathcal{S}_\Gamma = (\mathcal{O}_\Gamma,\mathcal{U}_\Gamma) \in \mathfrak{O} \times \mathfrak{U}$ such that $\mathcal{T}, \mathcal{A}^{\mathcal{S}_\Gamma} \models \Gamma$ [2] (goal state). We use ϵ do denote the empty action $(\emptyset, \emptyset, \emptyset)$. Since the search space $\mathfrak{O} \times \mathfrak{U}$ is of size $2^{|\mathcal{L}_{\mathsf{occ}}|} \cdot 3^{\frac{|\mathcal{L}_{\mathsf{post}}|}{2}}$ $(< 2^{|\mathcal{L}_{\mathsf{occ}}|+|\mathcal{L}_{\mathsf{post}}|})$, there is no need to search for longer sequences than $2^{|\mathcal{L}_{\mathsf{occ}}|+|\mathcal{L}_{\mathsf{post}}|}$.

PLANEX(Π)
  $i := 0$; $\mathcal{S}_0 := (\emptyset, \emptyset)$;
  while $i < 2^{|\mathcal{L}_{\mathsf{occ}}|+|\mathcal{L}_{\mathsf{post}}|}$
    guess $\alpha = (\mathsf{pre}, \mathsf{occ}, \mathsf{post}) \in \mathsf{Op}[\mathsf{Ind}] \cup \{\epsilon\}$
    if $\mathcal{T}, \mathcal{A}^{\mathcal{S}_i} \not\models \mathsf{pre}$
      then return FALSE
    compute $\mathcal{S}_{i+1}$ such that $\mathcal{S}_i \stackrel{\alpha}{\rightarrow} \mathcal{S}_{i+1}$
    $i := i + 1$
  if $\mathcal{T}, \mathcal{A}^{\mathcal{S}_i} \not\models \Gamma$
    then return FALSE
  return TRUE

It is not difficult to show that Lemma 1 implies that **PLANEX**(Π) returns TRUE iff Π has a solution. Clearly, **PLANEX**(Π) works in NPSpace with the "projection oracle". If projection is in PSpace, then PLANEX is obviously in NPSpace. By using Savitch's result [16] that PSpace = NPSpace, we obtain that PLANEX is then in PSpace. Similarly, if projection is in ExpTime, since NPSpace $\subseteq$ ExpTime, we have that PLANEX can be decided in ExpTime. Finally, we will show the less straightforward result that PLANEX is in co-NExpTime if projection is in co-NExpTime. To this end, we develop an alternative NExpTime algorithm which returns TRUE iff the planning task Π has no solution. Let $\mathfrak{S} = \mathfrak{O} \times \mathfrak{U}$ and $\mathfrak{Q} = \{\mathsf{pre}(\alpha) \mid \alpha \in \mathsf{Op}[\mathsf{Ind}]\} \cup \{\Gamma\}$. The alternative algorithm has three steps: (i) guess an (exponentially big) set T of tuples $(\mathcal{S}, \mathcal{Q})$ from $\mathfrak{S} \times \mathfrak{Q}$; (ii) check whether for all tuples $(\mathcal{S}, \mathcal{Q})$ it holds that $\mathcal{T}, \mathcal{A}^{\mathcal{S}} \not\models \mathcal{Q}$; (iii) check if the following holds: in every run of the original **PLANEX**(Π) procedure, there is at least one projection test $\mathcal{T}, \mathcal{A}^{\mathcal{S}} \models \mathcal{Q}$?

[2] from now on, if $\mathcal{S} = (\mathcal{O},\mathcal{U})$, we write $\mathcal{S}$ as an abbreviation for the action $(\emptyset, \mathcal{O},\mathcal{U})$

such that $(\mathcal{S}, \mathcal{Q}) \in T$. If (ii) and (iii) give positive answers, return TRUE. Since (ii) and (iii) can be checked in ExpTime, the steps (i)-(iii) can be executed in NExpTime. Thus, we obtained the following lemma:

Lemma 2. *Let $\mathcal{L} \in \{\mathcal{ALC}, \mathcal{ALCO}, \mathcal{ALCI}, \mathcal{ALCQ}, \mathcal{ALCIO}, \mathcal{ALCQO}, \mathcal{ALCQI}, \mathcal{ALCQIO}\}$. The unconditional PLANEX in $\mathcal{L}$ has the same upper complexity bound as projection in $\mathcal{L}$.*

We show that the upper complexity bounds established in Lemma 1 are tight by the following easy reduction of projection to PLANEX. Let $\mathcal{A}$ be an ABox, α an action with empty pre-conditions and empty occlusions and only with unconditional post-conditions, and φ an assertion.We define the planning task $\Gamma_{\mathcal{A},\alpha,\varphi}$ as $\Gamma_{\mathcal{A},\alpha,\varphi} := (\emptyset, \emptyset, \{\alpha\}, \mathcal{A}, \{\varphi\})$. It is not difficult to see that $\mathcal{A}^\alpha \models \varphi$ iff $\Gamma_{\mathcal{A},\alpha,\varphi}$ has a solution.

Since the lower bounds for projection from Theorem 1 hold already in the case of the empty TBox and an atomic action with empty pre-conditions and occlusions and only with unconditional post-conditions [2], we conclude that the complexity bounds from Lemma 2 are optimal, i.e. plan existence problem is of exactly the same computational complexity as projection.

Theorem 2. *The unconditional plan existence problem is:*

(a) PSpace*-complete in $\mathcal{ALC}$, $\mathcal{ALCO}$, $\mathcal{ALCQ}$, and $\mathcal{ALCQO}$;*
(b) ExpTime*-complete in $\mathcal{ALCI}$ and $\mathcal{ALCIO}$;*
*(c) co-*NExpTime*-complete in $\mathcal{ALCQI}$ and $\mathcal{ALCQIO}$.*

6 Complexity of Planning: Conditional Post-Conditions

If we allow for conditional post-conditions in operators, the complexity results from the previous section do not hold anymore. With conditional post-conditions, already in the propositional case, conformant PLANEX is ExpSpace-hard [8, 15]. In this section we will show that conditional PLANEX is decidable for DLs between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$. Decidability will be shown by an 2-ExpSpace algorithm.

Let $\Pi = (\mathsf{Ind}, \mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma)$ be a planning task for which we want to decide if it has a solution. For the sake of simplicity, we assume that occlusions in operators from Op are empty, i.e. operators are of the form $(\mathsf{pre}, \mathsf{post})$. Non-empty occlusions can be treated similarly as in the previous section. We will also use abbreviations introduced in the previous section. Moreover, we set

$$\mathcal{L}_{\mathsf{post}} := \{\psi, \neg\psi \mid \varphi/\psi \in \mathsf{post}, \alpha = (\mathsf{pre}, \mathsf{post}), \alpha \in \mathsf{Op}[\mathsf{Ind}]\}$$

and

$$\mathcal{C}_{\mathsf{post}} := \{\varphi \mid \varphi/\psi \in \mathsf{post}, \alpha = (\mathsf{pre}, \mathsf{post}), \alpha \in \mathsf{Op}[\mathsf{Ind}]\}.$$

An *update* in Π is a consistent subset of $\mathcal{L}_{\mathsf{post}}$. Let $\mathfrak{U}$ be the set of all updates in Π. A *context* $\mathcal{C}$ in Π is a consistent subset of $\mathcal{C}_{\mathsf{post}} \cup \neg\mathcal{C}_{\mathsf{post}}$ such that for every

$\varphi \in \mathcal{C}_{\mathsf{post}}$, it is the case that either $\varphi \in \mathcal{C}$ or $\neg\varphi \in \mathcal{C}$. Moreover let $\mathfrak{C}$ be the set of all contexts in Π. Let $\mathfrak{M}$ be the set of *admissible* mappings $m : \mathfrak{U} \to \mathfrak{C}$, where a mapping m is admissible iff there exists an interpretation $\mathcal{I}$, such that $\mathcal{I} \models \mathcal{A}, \mathcal{T}$ and for all $\mathcal{U} \in \mathfrak{U}$ it holds that $\mathcal{I}_{\mathcal{T}}^{\mathcal{U}} \models m(\mathcal{U})$. [3]. Intuitively, if $m(\mathcal{U}) = \mathcal{C}$, it means that after updating a model $\mathcal{I}$ of $\mathcal{A}$ and $\mathcal{T}$ with $\mathcal{U}$, all assertions from $\mathcal{C}$ will hold. Thus every admissible m describes a relevant class of possible initial models of $\mathcal{A}$ and $\mathcal{T}$. The number of different mappings $m : \mathfrak{U} \to \mathfrak{C}$ is at most $2^{|\mathcal{C}_{\mathsf{post}}| \cdot 2^{\mathcal{L}_{\mathsf{post}}}}$, and for every m it can be checked in ExpSpace if it is admissible, if projection is in ExpSpace. The search space $\mathfrak{S}$ is the set of all mappings $\mathcal{S} : \mathfrak{M} \to \mathfrak{U}$, the size of which $|\mathfrak{S}| \leq 2^{|\mathcal{L}_{\mathsf{post}}| \cdot 2^{|\mathcal{C}_{\mathsf{post}}| \cdot 2^{\mathcal{L}_{\mathsf{post}}}}}$. Similarly as in the previous section, we define the transition relation $\xrightarrow{\alpha}$ on $\mathfrak{S} \times \mathfrak{S}$. For $\mathcal{S}, \mathcal{S}' \in \mathfrak{S}$, $m \in \mathfrak{M}$, and $\alpha = (\mathsf{pre}, \mathsf{post})$ we set $\mathsf{post}_{\alpha,\mathcal{S},m} := \{\psi \mid \varphi/\psi \in \mathsf{post}, \varphi \in m(\mathcal{S}(m))\}$. We say that $\mathcal{S} \xrightarrow{\alpha} \mathcal{S}'$ iff

$$\mathcal{S}'(m) = (\mathcal{S}(m) \setminus \neg\mathsf{post}_{\alpha,\mathcal{S},m}) \cup \mathsf{post}_{\alpha,\mathcal{S},m} \quad \text{for all } m \in \mathfrak{M}$$

Moreover, for $\mathcal{B}$ be an ABox, we say that $\mathcal{T}, \mathcal{A}^{\mathcal{S}} \models^* \mathcal{B}$ iff for all $m \in \mathfrak{M}$ the following holds: for all interpretations $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{A}, \mathcal{T}$, if for all $\mathcal{U} \in \mathfrak{U}$ it holds that $\mathcal{I}_{\mathcal{T}}^{\mathcal{U}} \models m(\mathcal{U})$, then $\mathcal{I}_{\mathcal{T}}^{\mathcal{S}(m)} \models \mathcal{B}$.

condPLANEX(Π)
  $i := 0$; $\mathcal{S}_0(m) = \emptyset$ for all $m \in \mathfrak{M}$;
  while $i < 2^{|\mathcal{L}_{\mathsf{post}}| \cdot 2^{|\mathcal{C}_{\mathsf{post}}| \cdot 2^{\mathcal{L}_{\mathsf{post}}}}}$
    if $\mathcal{T}, \mathcal{A}^{\mathcal{S}_i} \models^* \Gamma$
      then return TRUE
    guess $\alpha = (\mathsf{pre}, \mathsf{post}) \in \mathsf{Op}[\mathsf{Ind}]$
    if $\mathcal{T}, \mathcal{A}^{\mathcal{S}_i} \not\models^* \mathsf{pre}$
      then return FALSE
    compute $\mathcal{S}_{i+1}$ such that $\mathcal{S}_i \xrightarrow{\alpha} \mathcal{S}_{i+1}$
    $i := i + 1$
  return FALSE

It is not difficult to show that **condPLANEX**(Π) indeed decides if Π has a solution. The search space $\mathfrak{S}$ is 3-exponential in the size of Π, thus **condPLANEX** requires 2-NExpSpace with a "$\models^*$ oracle". It is not difficult to see that $\models^*$ can be decided in 2-ExpSpace if projection is in ExpSpace. Thus we have that conditional PLANEX in 2-NExpSpace. Since 2-NExpSpace =2-ExpSpace [16], we obtain the following theorem:

Theorem 3. *The conditional plan existence problem is in 2-*ExpSpace *in the DLs between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$.*

7 Individuals not Part of Input

The previous decidability and complexity results are obtained under assumption that the set of individuals Ind used to instantiate operators is finite and a part

[3] $\mathcal{I}_{\mathcal{T}}^{\mathcal{U}}$ denotes the unique interpretation $\mathcal{I}'$ such that $\mathcal{I} \Rightarrow_{\mathcal{U}}^{\mathcal{T}} \mathcal{I}'$

of the input. This assumption is rather natural and in the line with the standard definitions of planning tasks for STRIPS operators from [4, 7]. Intuitively, Ind is the set of individuals the planning agent has control over.

Alternatively, one can omit individuals from the input and define a planning task Π as $\Pi = (\mathcal{T}, \mathsf{Op}, \mathcal{A}, \Gamma)$. The *extended plan existence problem* is the one of whether there is a solution for Π, defining a plan for Π to be a sequence of actions $\alpha_1, \ldots, \alpha_k$, where each α_i is obtained by instantiating an operator from Op with individuals from an infinitely countable set $\mathsf{N_I}$.

In the case of the datalog STRIPS, it is shown that the extended plan existence problem is undecidable [7, 6]. However, this undecidability result does not automatically carry over to the action formalism instantiated by DLs used in this paper. Indeed, the undecidability result from [7, 6] relies on the closed world assumption and negative pre-conditions. By using these two, one can define operators which are applicable only if instantiated with "unused" individuals. Such operators would have $\neg\mathsf{Used}(x)$ among its pre-conditions, and $\mathsf{Used}(x)$ in the list of post-conditions. Like this, one can enforce the usage of infinitely many individuals.

In the case of DLs considered in the previous sections, due to the open world assumption (OWA), it is not possible to state that all individuals not appearing in the initial ABox are instances of the concept $\neg\mathsf{Used}$. However, in the presence of the universal role U, we can make assertions over the whole domain. For example, the assertion $\forall U.\neg\mathsf{Used}(a)$ can ensure that all element domains are unused in the initial state. We will show that extended planning in $\mathcal{ALC}_U$ (extension of $\mathcal{ALC}$ with the universal role) is undecidable. Undecidability is shown by reducing the halting problem of a deterministic Turing machine to the extended plan existence problem, similar to [6].

Let $M = (Q, \Sigma, \delta, q_0, q_f)$ be a deterministic Turing machine, where

- $Q = \{q_0, \ldots, q_n\}$ a finite set of states;
- $\Sigma = \{\mathsf{blank}, a_1, \ldots, a_m\}$ a finite alphabet;
- $\delta : Q \times \Sigma \to Q \times \Sigma \times \{L, R\}$ is a transition function;
- q_0 is the initial state;
- $q_f \in Q$ is the final state.

Let $a = a_{i_0}, \ldots a_{i_k} \in \Sigma^*$ be an input word. We will define a planning task $\Pi_{M,a} = (\emptyset, \mathsf{Op}_{M,a}, \mathcal{A}_{M,a}, \Gamma_{M,a})$ such that a planner for Π simulates moves of the Turing Machine M. In the reduction, we use concept names $Q_0, \ldots, Q_n$, $\mathsf{Blank}, A_1, \ldots, A_m$, Used, Last, M, Done, and the role name right. We define the initial state $\mathcal{A}_{M,a}$, the goal $\Gamma_{M,a}$, and the set of operators $\mathsf{Op}_{M,a}$ as:

$$
\begin{aligned}
\mathcal{A}_{M,a} &:= \{(M \sqcap \forall U.\neg\mathsf{Used})(t_0)\} \cup \{A_{i_0}(t_0), \ldots, A_{i_k}(t_k)\} \\
&\quad \cup \{\mathsf{right}(t_0, t_1), \ldots \mathsf{right}(t_{k-1}, t_k)\} \\
\Gamma_{M,a} &:= \{\mathsf{Done}(t_0)\} \\
\mathsf{Op}_{M,a} &:= \{\mathsf{start}, \mathsf{create\_succ}(x.y), \mathsf{done}(x), \mathsf{done\_to\_left}(x, y)\} \cup \\
&\qquad \bigcup_{\delta(q,a)=(q',b,R)} \{\mathsf{right}_{q,a,q',b}(x, y)\} \cup \bigcup_{\delta(q,a)=(q',b,L)} \{\mathsf{left}_{q,a,q',b}(x, y)\}
\end{aligned}
$$

where the single operators (of the form $(\mathsf{pre}, \mathsf{post})$, $\mathsf{occ} = \emptyset$ for all operators) are defined as follows :

$$\begin{aligned}
\mathsf{start} &:= (\{M(t_0)\}, \{\mathsf{Used}(t_0), ..., \mathsf{Used}(t_k), \mathsf{Last}(t_k), \neg M(t_0), Q_0(t_0)\}) \\
\mathsf{create_succ}(x.y) &:= (\{\mathsf{Last}(x), \neg\mathsf{Used}(y)\}, \\
&\qquad \{\mathsf{right}(x,y), \neg\mathsf{Last}(x), \mathsf{Last}(y), \mathsf{Used}(y), \mathsf{Blank}(y)\} \\
\mathsf{right}_{q,a,q',b}(x,y) &:= (\{Q(x), A(x), \mathsf{right}(x,y)\}, \{\neg Q(x), \neg A(x), B(x), Q'(y)\} \\
\mathsf{left}_{q,a,q',b}(x,y) &:= (\{Q(x), A(x), \mathsf{right}(y,x)\}, \{\neg Q(x), \neg A(x), B(x), Q'(y)\} \\
\mathsf{done}(x) &:= (\{Q_f(x)\}, \{\mathsf{Done}(x)\}) \\
\mathsf{done_to_left}(x,y) &:= (\{\mathsf{Done}(x), \mathsf{right}(y,x)\}, \{\mathsf{Done}(y)\})
\end{aligned}$$

It is not difficult to show that the following lemma holds:

Lemma 3. *The Turing machine M halts on the input a iff there is a solution to the planning task $\Pi_{M,a} = (\emptyset, \mathsf{Op}_{M,a}, \mathcal{A}_{M,a}, \Gamma_{M,a})$.*

Thus, we obtained the following theorem:

Theorem 4. *The extended plan existence problem is undecidable in $\mathcal{ALC}_U$.*

We conjecture that in the fragments of $\mathcal{ALCQIO}$ (i.e. without the universal role) the extended plan existence problem is decidable. However, a proof is yet to be done.

8 Conclusion and Future Work

In this paper, we have shown that the plan existence problem (PLANEX) is decidable in the action formalism based on fragments of $\mathcal{ALCQIO}$. More precisely, PLANEX is shown to be of the same computational complexity as projection in the logics between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$ if operators have only unconditional post-conditions. It is also shown that occlusions do not make planning harder. If operators have conditional post-conditions, planning is shown to be in 2-ExpSpace. At the moment, it remains an open problem if this complexity bound is optimal. Finally, we have shown that the extended plan existence problem is undecidable in DLs providing for the universal role but we conjecture that it is decidable in the fragments of $\mathcal{ALCQIO}$ (without the universal role).

A future work will include a development and implementation of efficient planners for description logics. Unfortunately, the complexity results we obtained are quite discouraging. Unlike the propositional case, for DLs between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$, looking for polynomial-length plans is not easier than PLANEX, since the hardness results from Theorem 2 hold already for the plans of constant length. Thus it looks reasonable to start with "small" DLs, like $\mathcal{EL}$ or $\mathcal{EL}^{(\neg)}$, for which projection is in co-NP, and try to adapt some of the known techniques for SAT-based conformant planning [5, 13].

Acknowledgements: Thanks to Carsten Lutz for inspiring discussions. Many thanks to Ricard Gavaldà for his help concerning complexity classes.

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
2. F. Baader, C. Lutz, M. Milicic, U. Sattler, and F. Wolter. Integrating description logics and action formalisms: First results. In *Proceedings of AAAI-05*, Pittsburgh, PA, USA, 2005.
3. F. Baader, C. Lutz, M. Milicic, U. Sattler, and F. Wolter. Integrating description logics and action formalisms for reasoning about web services. Technical Report LTCS 05-02, TU Dresden, 2005. See http://lat.inf.tu-dresden.de/research/reports.html.
4. T. Bylander. The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1-2):165–204, 1994.
5. C. Castellini, E. Giunchiglia, and A. Tacchella. Sat-based planning in complex domains: Concurrency, constraints and nondeterminism. *Artif. Intell.*, 147(1-2):85–117, 2003.
6. K. Erol, D. S. Nau, and V. S. Subrahmanian. Complexity, decidability and undecidability results for domain-independent planning: A detailed analysis. Technical Report CS-TR-2797, University of Maryland College Park, 1991.
7. K. Erol, D. S. Nau, and V. S. Subrahmanian. Complexity, decidability and undecidability results for domain-independent planning. *Artificial Intelligence*, 76(1-2):75–88, 1995.
8. P. Haslum and P. Jonsson. Some results on the complexity of planning with incomplete information. In *Proceedings of ECP'99*, pages 308–318, 1999.
9. I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.
10. H. Liu, C. Lutz, M. Milicic, and F. Wolter. Reasoning about actions using description logics with general TBoxes. In *Proceedings of JELIA 2006*, volume 4160 of *LNAI*, pages 266–279, 2006.
11. H. Liu, C. Lutz, M. Milicic, and F. Wolter. Updating description logic ABoxes. In *Proceedings of KR'06*, pages 46–56, 2006.
12. B. Nebel. Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249, 1990.
13. H. Palacios and H. Geffner. Compiling uncertainty away: Solving conformant planning problems using a classical planner (sometimes). In *Proc. of AAAI'06*, 2006.
14. R. Reiter. *Knowledge in Action*. MIT Press, 2001.
15. J. Rintanen. Complexity of planning with partial observability. In *Proceedings of (ICAPS 2004)*, pages 345–354, 2004.
16. W. J. Savitch. Relationship between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4:177–192, 1970.
17. M. Thielscher. Introduction to the Fluent Calculus. *Electronic Transactions on Artificial Intelligence*, 2(3–4):179–192, 1998.