Verification of Golog Programs over Description Logic Actions

Dissertation

zur Erlangung des akademischen Grades
Doktoringenieur (Dr.-Ing.)

vorgelegt an der
Technischen Universität Dresden
Fakultät Informatik

eingereicht von
Dipl.-Inf. Benjamin Zarrieß
geboren am 4. November 1985 in Stendal

verteidigt am 2. November 2017

Gutachter:
Prof. Dr.-Ing. Franz Baader
Technische Universität Dresden

Prof. Sebastian Sardiña, Ph.D.
RMIT University Melbourne

Dresden, im Juli 2018

Acknowledgements

This work was financially supported by the Deutsche Forschungsgemeinschaft (DFG) in the research unit FOR 1513 on Hybrid Reasoning for Intelligent Systems.

I am indebted to my supervisor Franz Baader for his generous support and patience. To my colleague Jens Claßen for all his advice and guidance during the whole period of this work.

I am thankful to Sebastian Sardiña for his work as the external reviewer of this thesis. I would also like to thank my colleagues Anni-Yasmin Turhan, Barbara Morawska, İsmail İlkan Ceylan, Stefan Borgwardt and Marcel Lippmann for their support.

Contents

Chapter 1

Introduction

The central subject of this thesis is the *verification problem* in the Golog family of programming languages [Lev+97; DLL00]. Informally, the verification problem asks whether or not a program is doing what it is supposed to do. Formal verification is concerned with methods that can provide mathematical proofs or disproofs of program correctness. For this, one first has to formally specify the required properties of the program. Then one would like to verify that all possible executions of the given program satisfy the specification. However, when considering fully automated verification one is faced with Rice's theorem. It states that all non-trivial, semantic properties of programs are undecidable. No algorithm that takes an arbitrary program and a formal specification as input and *decides* the correctness of the program w.r.t. the specification can exist. One approach to cope with this problem is to use algorithms for the general problem that are sound but might come without a termination guarantee, or are even incomplete. Another approach is to identify expressive relevant fragments of the language that admit efficient decision procedures for the verification problem. In this thesis we are interested in the latter approach for programs written in the Golog language.

Golog is a family of action programming languages designed for the high-level control of autonomous agents who act in a dynamic and incompletely known world. For example, it has been successfully used for the high-level control of mobile robots [Bur+99]. To write a Golog program the programmer first provides a *logical action theory* that consists of a knowledge base that incompletely describes the initial state of the world, and a description of preconditions and effects of user-definable primitive actions, which are considered to be atomic and stand for the basic abilities of the agent to change the world. To describe a complex (possibly open-ended) task programming constructs such as loops, if-then-else statements, and constructs for non-deterministic choice are available to combine actions defined in the underlying action theory. Resolving the non-determinism is then left to the interpreter of the program. Since the initial state is only incompletely known, evaluating the branching conditions and choosing the next legal action of the program requires the ability to reason about the preconditions and effects of the primitive actions. However, especially due to the non-deterministic nature of Golog and the uncertainty about the initial state, testing can be very difficult or even impossible. Therefore, automated verification of Golog programs is an active and challenging research topic. For example, if a Golog program is used for the high-level control of a mobile robot, it is highly desirable to be able to guarantee that the program satisfies certain properties before actually executing it on the robot.

In Golog, the underlying action formalisms is the *Situation Calculus* [MH69; Rei01a] which employs first-order logic as its underlying knowledge representation language. This choice results in a very expressive and flexible language, but it causes basic reasoning problems to be undecidable and infeasible in practice. In particular, the verification problem is undecidable as well.

In this thesis, we eliminate this particular source of undecidability by using action formalisms based on *Description Logics* (DLs). DLs [Baa+10] can be viewed as decidable fragments of first-order logic. The main objective of this thesis is to identify non-trivial expressive DL-based fragments of Golog and suitable temporal specification languages for which the verification problem is decidable.

In the remainder of this chapter, a brief, informal introduction to Golog and verification is given, and some of the relevant literature is reviewed. In Section 1.1, we give an impression of how actions are represented in an action theory formulated in the Situation Calculus. Section 1.2 is an introduction to action formalisms based on DLs. Afterwards in Section 1.3, we consider some details about the Golog family of programming languages. Section 1.4 gives an abstract overview of verification methods and how they have already been applied to Golog. In Section 1.5, an outline of the thesis is presented and the main contributions are summarized.

1.1 Action Theories in the Situation Calculus

When writing a Golog program for the control of an agent, the programmer first provides an action theory formulated in the *Situation Calculus*. An action theory provides incomplete information about the initial state of the world and describes the basic abilities of agents to change the world in a way that enables reasoning about the outcome of actions. In this section, we briefly introduce some of the basic concepts of the classical Situation Calculus, consider epistemic extensions for representing sensing actions and mention some of the other existing action formalisms and their relation to the Situation Calculus.

Reiter's Basic Action Theories

The Situation Calculus [MH69] is a well-established language for representing and reasoning about change in (second-order) predicate logic. We mainly consider the variant introduced by Reiter and his colleagues in [Rei91] and [PR99]. A detailed presentation with many additional results can be found in the book [Rei01a].

The language distinguishes first-order terms of three different sorts: *object*, *action* and *situation*. *Fluent predicate symbols* (*fluents* for short) are available to describe properties and relations between objects that might change as the result of an action execution. Formally, *actions* are first-oder terms such as, for example, the following expressions:

$$\texttt{repair}(x), \quad \texttt{drop}(\mathsf{box}),$$

where `repair` and `drop` are action names viewed as function symbols with arguments of sort object, x is a variable of sort object and box is an *object name* viewed as a constant symbol. Of course, only ground actions can be executed. Also variables of sort action are available ranging over the (possibly infinite) domain of actions. Unique name axioms for the finite set of relevant action names are part of the action theory.

A *situation* is formalized as a first-order term as well. It is a sequence of ground actions denoting the history of actions that have been executed so far starting in the initial situation represented by the constant S_0. To be able to talk about the world in a particular situation each fluent has an argument of sort situation in the last position. For example, the sentence

$$\neg Broken(\mathsf{box}, S_0) \wedge Broken\big(\mathsf{box}, do(\mathtt{drop}(\mathsf{box}), S_0)\big)$$

says that initially the box is not broken but is broken after doing the action $\mathtt{drop}(\mathsf{box})$ in the initial situation, where $do(\mathtt{drop}(\mathsf{box}), S_0)$ is the situation term representing the situation after doing the action $\mathtt{drop}(\mathsf{box})$ in S_0. The domain of all situations is an infinite tree rooted in S_0 and is built using the function symbol *do*. For example, if first the action α_0 and then afterwards α_1 is executed in some situation s, then the resulting situation term is denoted by $do(\alpha_1, do(\alpha_0, s))$.

It is a common prerequisite that a useful representation of action effects requires a solution to the so called *frame problem*. The problem deals with the question of how to obtain a representation of action effects such that only the actual changes caused by an action need to be explicitly represented but not the facts that remain unchanged by an action execution. Treating situations and actions as terms in the logic (called *reification*) and allowing quantification over them is one of the key features of the Situation Calculus, which allows for an axiomatic solution of the frame problem in terms of *successor state axioms* (SSA). For example, an SSA for the fluent *Broken* could be the following one

$$Broken(x, do(a,s)) \equiv \Big(\big(a \approx \mathtt{drop}(x)\big) \vee \big(Broken(x,s) \wedge a \not\approx \mathtt{repair}(x)\big)\Big), \tag{1.1}$$

where x (sort object), a (sort action) and s (sort situation) are universally quantified variables. It says that x is broken after doing a if and only if x was dropped, or it was already broken before and was not repaired. It follows that only the actions $\mathtt{drop}(x)$ and $\mathtt{repair}(x)$ affect the fluent *Broken*. In case $\mathtt{drop}$ is not mentioned in the SSA of any other fluent, we can say that $\mathtt{drop}(o)$ causes $Broken(o)$ to be true and changes nothing else, where o is an arbitrary object name.

Also so-called *non-local effects* can be defined. For example, we want to express that executing $\mathtt{drop}(\mathsf{box})$ also breaks *all* fragile objects inside box. Consider a fluent $In(x, y, s)$ expressing that x is inside of y in situation s, and a fluent $Fragile(x, s)$ for a fragile object x in situation s. To express that dropping an object causes *all* fragile things inside it to be broken as well, the axiom (1.1) is modified by replacing $a \approx \mathtt{drop}(x)$ by the formula

$$\exists y. \big(a \approx \mathtt{drop}(y) \wedge (x \approx y \vee In(x,y,s) \wedge Fragile(x,s))\big). \tag{1.2}$$

To axiomatize the preconditions of an action a special predicate *Poss* is available. For instance, the axiom

$$Poss(\mathtt{drop}(x), s) \equiv (Heavy(x,s) \wedge Slippery(x,s)),$$

where x and s are universally quantified, says that $\mathtt{drop}(x)$ is possible if and only if x is heavy and slippery.

A *basic action theory* as a whole consists of

- a sentence talking only about the initial situation,
- successor state axioms (exactly one for each relevant fluent predicate),
- one precondition axiom for each relevant action name and

- so-called foundational axioms.

The set of foundational axioms consist of unique name axioms for action names and a second-order induction axiom that defines the tree-shaped structure of the domain of all situations.

One benefit of this axiomatization is that *reasoning* about the effects of actions reduces to a deduction problem in second-order logic. Consider a basic action theory Σ describing the dynamic domain. The *projection* problem is the problem of deciding whether a given fluent formula ϕ is true after a given ground action sequence σ has been performed in the initial situation. One needs to check whether Σ entails the formula $\phi[do(\sigma, S_0)]$ obtained from ϕ by instantiating the situation arguments of the fluents in ϕ with the situation term $do(\sigma, S_0)$. Due to the definitional form of successor state axioms they can be used as rewrite rules. With a technique called *regression* it is possible to reduce the projection problem to a first-order entailment problem only w.r.t. the sentence describing the initial situation.

Note that the notion of change axiomatized in a basic action theory is based on several assumptions. First, all changes in the world are caused by the execution of an action. Moreover, actions are

- deterministic,
- take place instantaneously,
- preconditions and effects of an action only depend on the situation in which the action is executed, and
- there occurs only exactly one action at a time.

In this thesis, we only consider actions that are based on these assumptions. Note that they apply only to the *primitive actions*. Some of the limitations can be addressed when it comes to *complex actions* defined as Golog programs (Section 1.3).

Epistemic Situation Calculus

Typically, an agent has only incomplete information about its surroundings. In an epistemic extension of an action formalism one wants to represent not only the *actions* but also the *perception* of an agent and its abilities to acquire new information from the environment by means of sensing. The logical formalization requires an explicit distinction between the world-changing and the knowledge-changing effects of an action while solving the frame problem for both.

An epistemic extension of the Situation Calculus has been introduced in [SL03]. Asking about what is known or not known by the agent after some sequence of actions has occurred (*epistemic projection*) is one of the reasoning tasks that can be characterized as a deduction problem in the epistemic Situation Calculus based on an appropriate extension of a basic action theory. In [SL03] the notion of *knowledge* and *sensing* is based on several simplifying assumptions. Among them are the following ones:

- subjective truth implies objective truth, i.e. there are no false beliefs,
- the agent knows the effects of all actions, i.e. the basic action theory is known,

- if an action is executed, then the agent is alway aware of it, i.e. the history of executed actions is always known,
- actions only provide binary sensing results.

Moreover, only a single agent is considered. Sensing amounts to observing the truth value of an axiom in the environment. For example, one can define an action that describes the ability of an agent to observe whether the object named box is heavy or not, which corresponds to observing whether $Heavy(\mathsf{box})$ is true in the current situation.

The axiomatization of knowledge in [SL03] uses a possible-world semantics based on a knowledge fluent K that has two arguments of sort situation. The atomic formula $K(s,s')$ expresses that in situation s the situation s' is considered possible. Intuitively, some formula is *known* to be true in some situation s if it is true in all situation that are considered possible in s. For the fluent K a successor state axiom is defined such that regressing knowledge for solving projection is possible. For instance, purely sensing actions are axiomatized in a way such that they only affect K and no other fluent.

In [LL04; LL11] another variant of the epistemic Situation Calculus called $\mathcal{ES}$ has been introduced. It uses first-order modal logic. Instead of reification of states as situation terms with appropriate axiomatizations, $\mathcal{ES}$ offers model operators with a special semantics for representing and reasoning about actions and knowledge. As argued in [LL04; LL05], the result is a language with a more "workable semantics", where some meta-theoretic proofs are much simpler. Reiter's basic action theories can be formulated in $\mathcal{ES}$ as well.

In Chapter 6, a decidable epistemic action formalism (w.r.t. epistemic projection) is defined with the same underlying assumptions as listed above. The corresponding relationship with the epistemic Situation Calculus is investigated in Section 6.3.

Other Action Formalisms and their Relation to the Situation Calculus

Roughly speaking, the existing action formalisms in the literature can be divided into two categories: on the one hand the ones that use an axiomatic approach to solve the frame problem, and on the other hand the ones that solve it based on a more operational semantics of actions. Another distinctive feature of action formalisms is the base logic that is available to formulate the domain-dependent knowledge.

As introduced in the previous section, Reiter's basic action theories are an axiomatic approach with first-order logic as its base logic. For example, Thielscher's *Fluent Calculus* [Thi98] and the *Event Calculus* [KS86] belong to the same group but use different axiomatization techniques. The relationship between the three formalisms has been studied in [Thi11]. The *action language* $\mathcal{A}$ [GL93] is a formalism based on propositional logic and logic programs. Dynamic logics [HTK00] also provide an axiomatic approach to reasoning about actions.

A prominent formalism with an operational semantics is the *Action Description Language ADL* [Ped94]. Its semantics is based on a so-called state-transition model of action, where states are represented meta-theoretically as first-order relational structures. Furthermore, it uses an action-centric approach: for each action the domain designer provides a list of preconditions and a list of effect descriptions. The semantics of an action is defined in terms of a transition relation on states respecting the frame assumption. Intuitively, executing an action corresponds to updating the respective first-order structure, that completely describes the current state of the world. The language is often used as a planning formalism with a fixed finite domain and complete information about the initial state. Later, the formalism evolved into the standard planning language *PDDL* [FL03] with additional extensions. The relative expressiveness of ADL and PDDL compared to Reiter's basic action theories has been investigated in detail in [CHL07; Roe14]. STRIPS [FN71] is a prominent purely propositional formalism with an operational semantics. It is a fragment of ADL.

In order to obtain formalisms that are considerably more expressive than the propositional ones, but still offer decidable and practical reasoning services, various action formalisms based on description logics have been introduced (e.g. [Baa+05a; Liu+06; BLL10; GS10; Ahm+14]). In this thesis, we consider classes of action theories with an operational semantics and with description logics as base logics (Section 2.1 and Section 2.2).

The next section is an informal introduction to action formalisms based on description logics.

1.2 Action Formalisms based on Description Logics

In this section, we briefly introduce the syntax of description logic knowledge bases using a simple example. We then use this example to discuss problems that arise when one wants to define action effects and preconditions in presence of such a knowledge base. Afterwards, we briefly review some of the existing solutions available in the literature.

Description Logic Knowledge Bases

Description Logics (DLs) [Baa+10] are a family of logic-based knowledge representation formalisms, that can be viewed as decidable fragments of first-order logic. The spectrum of DLs includes inexpressive members such as $\mathcal{EL}$, where certain reasoning tasks can be decided in polynomial time [BBL05], and also expressive member such as $\mathcal{ALC}$ [SS91] and $\mathcal{ALCQIO}$, where deciding consistency of a knowledge base is ExpTime-complete and NExpTime-complete, respectively. Nevertheless, practical and efficient reasoning tools, that are successfully used, exist also for those expressive DLs (e.g. [MSH09; SLG14]).

The primitive ingredients of the *variable-free DL syntax* are

- *concept names* (e.g. *Device*, *PowerSupply*, *On*) representing sets of objects,
- *role names* (e.g. *ConnectedTo*) describing binary relations of objects, and
- *object names* (e.g. laptop, battery) referring to concrete objects.

Several concept constructors are available to build *complex concepts* describing sets of objects. For example,

$$\mathit{Device} \sqcap \mathit{On} \sqcap \forall \mathit{ConnectedTo}.\{\mathsf{battery}\} \tag{1.3}$$

represents the set of all those devices that are turned on and can be only connected to the object battery. It uses the concept constructors $\sqcap$ (conjunction), $\forall$ (value restriction) and $\{o\}$ (nominal). A nominal describes a singleton set.

In first-order syntax the concept (1.3) can be written as the following formula with one free variable x:

$$\mathit{Device}(x) \wedge \mathit{On}(x) \wedge \forall y.\, (\mathit{ConnectedTo}(x,y) \rightarrow y \approx \mathsf{battery})\,.$$

Different DLs offer different sets of concept constructors and role constructors. An overview can be found in [Baa+10].

A *DL knowledge base* consists of an *ABox* and a *TBox*. The ABox is a set of ABox assertions describing properties of concrete objects and their role relationships. For example, the following ABox

$$\begin{aligned} \{\, &(\mathsf{laptop} \in \mathit{Device} \sqcap \mathit{On})\,, \\ &(\mathsf{battery} \in \mathit{PowerSupply})\,, \\ &((\mathsf{laptop}, \mathsf{battery}) \in \mathsf{ConnectedTo})\,\} \end{aligned} \tag{1.4}$$

incompletely describes the current state of world by stating that laptop is a device that is turned on and is connected to battery, which is a power supply. An ABox is interpreted under the *open-world assumption*, i.e. facts that are not a consequence of an ABox are neither assumed to be false nor assumed to be true. Thus, we can say that an ABox provides an *incomplete* description of the state of the world. A TBox is a set of terminological axioms and consists of so-called *concept inclusions* such as the following one:

$$\mathit{Device} \sqcap \mathit{On} \sqsubseteq \exists \mathit{ConnectedTo}.\mathit{PowerSupply}. \tag{1.5}$$

It states that devices that are turned on are always connected to some power supply. In first-order syntax this axiom can be written as follows:

$$\forall x.\, (\, (\mathit{Device}(x) \wedge \mathit{On}(x)) \rightarrow \exists y.\, (\mathit{ConnectedTo}(x,y) \wedge \mathit{PowerSupply}(y))\,)\,.$$

DL axioms such as the ones in (1.4) and (1.5) only describe static knowledge.

Representing Actions in Presence of a TBox

DLs are too inexpressive to axiomatize Reiter's basic action theories. To obtain an expressive action formalism based on DLs, a first ADL-like action formalism has been introduced by Baader et al. in [Baa+05a]. It was shown to be a fragment of Reiter's variant of the Situation Calculus with a decidable projection and executability problem. The only difference is that in [Baa+05a] also non-deterministic effects can be defined. Besides this exception, all underlying assumptions mentioned in Section 1.1 also carry over to this formalism.

Extensions have been introduced in [Liu+06] and [BLL10]. The formalisms mainly differ in the way the TBox is incorporated. In presence of global domain constraints provided in the TBox one has to be aware of the *ramification problem*, which is the problem of determining indirect action effects, and the *qualification problem*, which is the problem of determining implicit preconditions of actions. Both are well-known representational problems. Lin and Reiter [LR94] distinguished two kinds of state constraints: *ramification constraints* and *qualification constraints*. The first type gives rise to indirect action effects and the second one to implicit preconditions.

Ramification constraint. Consider the problem of defining the effects of an action named

$$\alpha := \texttt{disconnect}(\mathsf{laptop}, \mathsf{battery}),$$

in a domain, where (1.4) incompletely describes the initial situation and (1.5) is treated as a global state constraint. Consider the set $\mathsf{eff}(\alpha)$ describing the set of all effects of action α:

$$\mathsf{eff}(\alpha) := \big\{\langle \mathit{ConnectedTo}, \{(\mathsf{laptop}, \mathsf{battery})\}\rangle^-\big\}. \tag{1.6}$$

The semantics of α is given in terms of a transition relation on first-order interpretations, i.e. the execution of α updates an interpretation. Intuitively, we can interpret this description under the frame assumption as follows: the only specified effect in $\mathsf{eff}(\alpha)$ is a delete-effect on the role name *ConnectedTo*. The execution of α in a model of (1.4) and (1.5) deletes the pair of objects $(\mathsf{laptop}, \mathsf{battery})$ from the interpretation of *ConnectedTo* and changes nothing else in the model because of the frame assumption. As expected, the ABox assertion

$$(\mathsf{laptop}, \mathsf{battery}) \sqsubseteq \neg \mathit{ConnectedTo}$$

is true after doing α.

Note that the ABox (1.4) also has a model in which battery is the only power supply of laptop. After executing α in such a model we obtain an interpretation, where laptop is still turned on but is not connected to any power supply. This violates the concept inclusion (1.5). Therefore, under a semantics that assumes that $\mathsf{eff}(\alpha)$ is a complete description of *all* effects of α, the domain description is inconsistent. An action semantics that properly respects (1.5) as a global state constraint has to make sure that every model of (1.5) is transformed again into a model of (1.5). An intuitive semantics for the present example should treat (1.5) as a ramification constraint. It should turn off laptop after doing α by inferring the indirect delete-effect $\langle \mathit{On}, \{\mathsf{laptop}\}\rangle^-$ if battery is the only power supply of laptop before doing α.

Qualification constraint. There can also be dependencies between all three components of the domain: the preconditions, the effects and the TBox. Now, we consider the following ABox describing the initial situation

$$\{(\mathsf{laptop} \sqsubseteq \mathit{Device} \sqcap \neg \mathit{On})\} \tag{1.7}$$

and the action $\texttt{turn-on}(\mathsf{laptop})$ with the following preconditions and effects

$$\begin{aligned} \mathsf{pre}(\texttt{turn-on}(\mathsf{laptop})) &:= \{\mathsf{laptop} \sqsubseteq \neg \mathit{On}\} \\ \mathsf{eff}(\texttt{turn-on}(\mathsf{laptop})) &:= \big\{\langle \mathit{On}, \{\mathsf{laptop}\}\rangle^+\big\}. \end{aligned} \tag{1.8}$$

We assume that both sets describe all preconditions and all effects, respectively. The only precondition is that laptop is turned off. The execution of $\texttt{turn-on}(\mathsf{laptop})$ adds the object laptop to the interpretation of *On*. Obviously, the ABox (1.7) and the axiom (1.5) have models where laptop is not connected to any power supply. In those models, the execution leads to a state where laptop is turned on without being connected to any power supply which is in contradiction with (1.5). Again, the domain model is inconsistent if we assume that the sets of preconditions and effects given in (1.8) are complete. Now, it would be more intuitive to consider (1.5) as a qualification constraint that yields

$$(\mathsf{laptop} \sqsubseteq \exists \mathit{ConnectedTo}.\mathit{PowerSupply})$$

as an implicit precondition of $\mathtt{turn\text{-}on}(\mathsf{laptop})$ saying that laptop is required to be connected to at least one power supply. Another solution would be to add an additional effect that connects laptop with some power supply if no separate action for a connection is represented.

Usually, whether a state constraint should better be treated as a ramification or qualification constraint depends on the particular domain and the particular action under consideration. The concrete decision is often deferred to the user.

In [Baa+05a] both problems are avoided by carefully restricting the syntax of the TBox (only so-called *acyclic TBoxes* are allowed) and the syntax of effect descriptions (called *post-conditions* in [Baa+05a]). In [Liu+06] an unrestricted TBox is treated as a set of ramification constraints and actions are extended with expressive non-deterministic features. Unfortunately, the reasoning task of checking whether actions preserve the TBox (called *consistency*) is undecidable. A simpler deterministic solution for treating concept inclusions as ramification constraints is provided in [BLL10]. So-called (action-independent) *causal relationships* can be specified by the domain designer. They are then used to derive indirect action effects. Basic reasoning tasks such as projection and consistency with the TBox remain decidable and have the same complexity as deciding consistency of DL knowledge bases for most of the considered expressive DLs [BLL10].

Other DL-based action formalisms have been, for instance, introduced in [GS10] and [Ahm+14]. In [GS10] acyclic TBoxes are integrated in the same way as in [Baa+05a]. The reasoning tasks studied in [Ahm+14] involve general TBoxes, but indirect effects and implicit preconditions are not considered. A notable difference is that the formalisms in [Baa+05a; Liu+06; BLL10] only consider actions with *local effects*, whereas in the formalisms in [GS10; Ahm+14] also *non-local effects* can be specified. A *local effect* only affects the objects that are explicitly named in the action term such as laptop and $\mathsf{battery}$ in the example above. A non-local effect can affect also unnamed objects such as the action $\mathtt{drop}(\mathsf{box})$ considered in Section 1.1, that breaks all possibly unnamed fragile objects inside the box as well.

In Chapter 2, we consider an action language that allows us to abstract from a concrete syntax of effect descriptions and generalizes some of the formalisms mentioned above.

Representing actions in presence of complex state constraints such as concept inclusions requires also a complex and fine-grained modeling of actions in order to be able to integrate both components. Furthermore, it is desirable to equip the user with an efficient reasoning service for checking consistency of actions w.r.t. the state constraints.

In this thesis, we consider DL-based fragments of the action programming language Golog for describing complex actions. Checking whether all executions of a program always respect the TBox is then viewed as an instance of the verification problem (defined in Section 2.4). Note that this view integrates state constraints as part of the correctness specification for a concrete program and not as part of the underlying description of the primitive actions. A brief general introduction to Golog is given in the following section.

1.3 The Golog Family of Action Programming Languages

The *Golog* ("Al**go**l in **log**ic") family of action programming languages (e.g. [Lev+97; DLL00; Bou+00; Sar+04]) has already been extensively studied as a language for *high-level control* of autonomous agents. It is part of a knowledge representation and reasoning approach to *cognitive robotics* [LR98; LL08] based on the Situation Calculus. According to Levesque and Reiter cognitive robotics is concerned with "*the study of the knowledge representation and reasoning problems faced by an autonomous robot (or agent) in a dynamic and incompletely known world*". In their position paper on high-level robot control [LR98] they raised, among other questions, the following one:

> "*When should the inner workings of an action be available to the robot for reasoning and when should the action be considered primitive or atomic?*"

In Golog both representations are available. The (user-definable) primitives of the language are the actions defined in an underlying basic action theory, which provides the declarative part of the program. To describe a complex task, actions can be combined with imperative and non-deterministic program constructs and with tests, which are first-order formulas referring to the current state. For example, consider an agent, whose task is to repair the faults of a certain device named dev (object name). Faults are reified as objects related to dev via a binary fluent *HasFault*. An excerpt of a simple control program for such an agent could be the following one:

$$\begin{aligned}&\textbf{while } \exists x.\mathit{HasFault}(\mathsf{dev},x) \textbf{ do}\\ &\quad \mathsf{pick}(x) \to \mathit{HasFault}(\mathsf{dev},x)?; \mathtt{repair}(\mathsf{dev},x);\\ &\textbf{end}.\end{aligned} \tag{1.9}$$

As long as dev has some fault, the agent non-deterministically picks one of them and repairs it. With this highly non-deterministic program the faults are repaired one by one, but the programmer does not fix a certain order in advance. It is the task of the interpreter of the program to evaluate the tests and to choose a course of actions. Since the initial state is only incompletely represented in the action theory, reasoning about the effects of the primitive actions is required for this task. For instance, for executing the program above the successor state axiom for *HasFault* and the precondition axiom of `repair` become relevant.

In [DLL00] an extension of Golog called *ConGolog* is introduced, which, among other things, extends basic Golog [Lev+97] with (interleaved) concurrency. For example, in ConGolog one can also model exogenous actions occurring in the environment and describe the interleaved execution with a reactive control program of an agent. Thus, ConGolog programs are often non-terminating. The transition semantics of ConGolog is defined axiomatically on top of the Situation Calculus. This is done by treating program expressions as first-order terms in the logic as well and by using quantification over programs.

In the so-called *offline execution* mode the interpreter first analyses the program as a whole and computes a finite executable ground action sequence that represents a successful terminating execution of the program. The resulting sequence is then sent to the actual executor. This is similar to classical planning with the difference that in Golog the modeler is equipped with a fully-fledged imperative language for providing additional control information. Using the axiomatic semantics of Golog and ConGolog, the decision variant of the offline execution task can be expressed as a deduction problem in second-order logic. However, for fragments of the language, where the initial state is given as a closed-world database, practical implementations of offline execution systems written in Prolog are provided in [DLL00; Rei01a].

A different view on a Golog program is obtained, if *online execution* is considered. In this variant of Golog the tests in the program refer to what the executing agent itself knows or does not know about the world at run-time. In case of incomplete information about the world, acquiring additional information at run-time is required in order to decide on the next action to execute. For example, to the program (1.9) one could add an online program that describes how the agent has to use its sensors in order to identify the faults before repairing them. A Golog variant based on the epistemic Situation Calculus [SL03], that also accounts for online execution and sensing has been introduced in [Rei01b].

IndiGolog (incremental deterministic Golog) [Sar+04] is another member of the Golog family that supports concurrency, online execution and sensing. Furthermore, it includes a search operator, which allows for an integration of planning and lookahead for resolving the non-determinism. A detailed presentation of the language with a focus on implementation and applications can be found in [De +09].

In this thesis, we consider a ConGolog-like programming language (Chapter 2, Section 2.4). In Chapter 7, also knowledge and sensing is considered.

1.4 Formal Verification

Formal verification is concerned with the problem of checking whether all possible executions of a given program satisfy a given formal specification. For a Turing complete programming language and non-trivial specifications the verification problem is in general undecidable due to Rice's theorem. Despite this negative theoretical result, systems for formal verification have become valuable tools that are successfully used in practice. In this section, we first review several general approaches that are concerned with solving the verification problem in practice. Second, we give a brief overview of existing verification methods for Golog programs and Golog-like action languages.

General Approaches

In a recent survey paper on formal verification [BH14] three different general approaches to formal verification are distinguished: *deductive verification*, *model checking* and *abstract interpretation*. All of them are successfully used in practice [BH14].

For using *deductive verification* the correctness of a given program w.r.t. a specification is formalized in a suitable logic. The logic must be expressive enough such that the correctness of the program can be stated as a theorem in that logic. Then, the user tries to find a proof for that theorem by using appropriate deduction methods. Due to the high expressiveness of the logic often only semi-automated deduction methods are applicable. One example is Hoare logic [Hoa69], where it is possible to express preconditions and postconditions of programs. For proving such statements a proof calculus is available that requires the user to provide loop invariants. Other methods express program correctness in higher-order logic and employ a proof assistants such as *Isabelle* [NPW02], or a higher-order theorem prover like, for example, *PVS* [Owr+96]. An advantage of deductive verification is, that it is quite generally applicable. A disadvantage is, that it often requires a huge amount of user interaction and expert knowledge.

Model checking [CGP01; BK08], in contrast, is a fully automated approach to formal verification. It requires that the program (or an abstraction thereof) is given in terms of a transition system with finitely many states labeled with atomic propositions. Desirable properties of the program are then expressed in a temporal logic. Typical categories of such properties are *safety properties* ("something bad never happens") or *liveness properties* ("something good eventually happens") [Lam77]. The verification problem amounts to the check whether the transition system is a model of the temporal formula. Prominent approaches are based on *propositional linear-time logic LTL* [Pnu77] for specifying properties of infinite executions of a non-terminating concurrent program. The LTL model checking problem can be efficiently solved using an automata-based approach [VW86]. For example, one can express a property like "something good happens infinitely often". Other popular specification languages are the *propositional branching-time temporal logics* CTL and the more expressive extension CTL* [CE81]. Instead of a single path, a computation is viewed as an infinite tree where the branching comes from the non-determinism in the system model. The logics CTL and CTL* offer quantification over the different outgoing paths of a state. CTL* can be viewed as an extension of LTL with path quantifiers and it subsumes both logics LTL and CTL. *Symbolic model checking* [Bur+90; McM93] is an efficient approach for CTL model checking. It is based on compact encodings of the state space using variants of binary decision diagrams. However, a disadvantage of model checking in general is, that it is only applicable if a finite-state model of the system can be obtained.

Another verification method is *abstract interpretation* [CC77]. Instead of analyzing the possibly infinite state space of the program itself, it works on finite and sound approximations of the program executions. If the abstraction is proven correct, then this carries over to the actual program. However, an error detected in the abstraction is not necessarily also an error in the actual program. The method is incomplete and not fully automated, but it is successfully applied in practice (e.g. [Ber+11]).

Verification of Golog Programs

As discussed in Section 1.3, the members of the Golog family are very flexible programming languages. The programmer does not need to foresee all eventualities and can rely on the interpreter of the program, that resolves the non-determinism in the program and evaluates the tests using the action theory. However, especially in case of online execution a lot of uncertainty is involved. The initial state is only incompletely known and relevant information needs to be acquired at runtime by means of sensing. Moreover, in presence of exogenous actions and concurrency writing a program is an error-prone task and makes testing very difficult. For example, if a variant of Golog is used for the high-level control of a mobile robot, it is highly desirable to have some guarantee that the program works as expected before actually executing it on the robot.

Therefore, research on action programming languages such as Golog has not only been focused on efficient interpreters but also on formal verification. The axiomatic semantics of Golog and ConGolog in second-order logic is directly amenable to deductive verification: a first approach in [DTR97] considers verification of temporal properties of non-terminating

Golog programs. Program correctness is formulated as a deduction problem in second-order logic extended with fixpoint constructors.

In [Liu02] a Hoare logic calculus for terminating Golog programs has been introduced. Verification of ConGolog programs using the higher-order theorem prover PVS has been investigated in [SLL02].

A first fully automated verification method for a core fragment of ConGolog has been introduced in [CL08]. Program properties of non-terminating programs can be expressed in a first-order extension of CTL. The verification algorithm in [CL08] combines deductive verification and model checking. The overall algorithm is inspired by a standard CTL model checking algorithm. But in order to deal with the first-order representations, it makes calls to a solver of deduction problems in first-order logic during its runs, that are not guaranteed to be terminating. The algorithm has been also extended to temporal properties written in a first-order variant of CTL* [CL10]. In this extension the algorithm needs to solve deduction problems in second-order logic.

In this work, we are interested in achieving decidability of the verification problem of Golog programs by restricting the expressiveness of the language. One source of undecidability is the use of full first-order as base logic. To overcome this problem we consider action formalism based on description logics such as the ones introduced in Section 1.2. First decidability results for the verification problem based on the action formalism in [Baa+05a] can be found in [BLM10]. However, instead of an actual Golog program over DL actions, this approach considers infinite sequences of actions accepted by a given Büchi automaton. As the logic for specifying properties of infinite sequences of DL actions, the approach uses the temporalized DL $\mathcal{ALCO}$-LTL [BGL12], which extends propositional LTL by allowing for the use of DL axioms in place of propositional letters. For example, one can express that the object laptop is infinitely often connected to some power supply:

$$\mathsf{GF}(\mathsf{laptop} \in \exists \mathit{ConnectedTo}.\mathit{PowerSupply}),$$

where the letters G ("globally") and F ("eventually") are temporal modalities. In [Lip14] the decidability result has been extended to a setting, where the action formalism in [BLL10] instead of the one in [Baa+05a] is used to describe the primitive actions. The exact complexities of the problems studied in [BLM10] and [Lip14] are still open.

In this thesis, we consider actual Golog-like programs instead of Büchi automata. Moreover, we extend the decidability results obtained in [BLM10; Lip14] towards more expressive action languages and specification languages. The goal is to further explore the boundary between decidable and undecidable fragments of Golog programs over DL actions in order to widen the applicability of the verification methods.

We develop an abstraction technique that allows us to reduce the verification problem for restricted classes of Golog programs and temporal properties to a decidable propositional model checking problem. For *pick-free ConGolog programs over DL actions with only local effects* and CTL* properties we obtain tight complexity results. One main limitation of the action formalisms in [Baa+05a; BLL10] is that only local effects can be described. We observe that an extension towards non-local effects leads to undecidability of the verification even for rather inexpressive DLs. However, we are able to identify syntactical restrictions on action descriptions that allow us to regain decidability in presence of non-local effects.

In order to extend our decidability results towards online executions of Golog programs, we introduce a new action formalism based on the DL $\mathcal{ALCO}$, where also sensing actions can be represented. We study the complexity of the verification problem for *knowledge-based ConGolog programs* over such actions.

1.5 Outline and Contributions of the Thesis

In this section, we present an outline of the structure of the thesis and summarize the main contributions. Many of the results in this thesis have already been published in [BZ13; ZC14; ZC15b; ZC15a; ZC16].

- In *Chapter 2*, we define the basic notions we use to formalize the verification problem for a general class of ConGolog programs. The term *$\mathcal{L}$-definability* of a finite set of ground actions, where $\mathcal{L}$ is a fragment of first-order logic, is defined. To be able to instantiate $\mathcal{L}$ with a decidable logic, an expressive description logic is defined that has the same expressiveness as the *two-variable fragment of first-order logic with counting* C^2 [GOR97]. This DL serves as an "umbrella logic" for the different DLs considered in the thesis. The semantics of a ConGolog program is defined in terms of a possibly infinite transition system, where each state is labeled with a first-order interpretation representing the current state of the world. Finally, the *verification problem* is formalized as an infinite-state model checking problem for *CTL* properties over $\mathcal{L}$-axioms*, where $\mathcal{L}$ denotes the base logic under consideration.

- In *Chapter 3*, we start investigating the computational properties of the verification problem. After showing some properties of the program semantics defined in Chapter 2, we prove that the verification problem for Golog programs over non-ground actions defined in a simple $\mathcal{ALC}$-action theory is already undecidable. The unbounded pick-operator for non-deterministically choosing action arguments is identified as one of the main sources of undecidability. Therefore, in the following only pick-free ConGolog programs over finitely many ground actions are considered. To prepare the abstraction technique for pick-free programs used in the subsequent chapters, the notion of a *reachable subprogram expression* is defined, and we show that there are at most exponentially many of them measured w.r.t. the size of the program. Moreover, the notion of a *context-bisimilar abstraction* of a possibly infinite transition system is defined. The abstraction is a propositional transition system. Intuitively, context-bisimilarity ensures that the concrete transition system and the abstract one satisfy the same temporal properties.

  The properties of the program semantics, which handles terminating, non-terminating and failing runs of a program in a uniform way, and the results regarding the number of reachable subprograms have already been published in

  [BZ13] Franz Baader and Benjamin Zarrieß. "Verification of Golog Programs over Description Logic Actions". In: *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*. Ed. by Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt. Vol. 8152. Lecture Notes in Computer Science. Springer, 2013, pp. 181–196. URL: https://doi.org/10.1007/978-3-642-40885-4_12.

- In *Chapter 4*, we consider the verification problem for
  - pick-free *$\mathcal{L}$-ConGolog programs over $\mathcal{L}$-definable actions with only local effects*, and
  - CTL* properties over $\mathcal{L}$-axioms,

  where $\mathcal{L}$ is a DL between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$. We prove decidability of the verification problem by showing that a finite bisimilar abstraction of the transition system induced by the program is effectively computable. This allows us to reduce the problem to a finite-state model checking problem for propositional CTL* properties. The abstraction technique is based on the newly introduced notion of a *dynamic type* of an interpretation. Intuitively, executing the program in interpretations of the same dynamic type leads to the same behavior. There are only finitely many dynamic types, and realizability of a given set of type elements can be reduced to a consistency check in the underlying DL $\mathcal{L}$. The idea behind this abstraction technique was first presented in

  [BZ13] Franz Baader and Benjamin Zarrieß. "Verification of Golog Programs over Description Logic Actions". In: *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*. Ed. by Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt. Vol. 8152. Lecture Notes in Computer Science. Springer, 2013, pp. 181–196. URL: `https://doi.org/10.1007/978-3-642-40885-4_12`

  and further developed in

  [ZC14] Benjamin Zarrieß and Jens Claßen. "Verifying CTL* Properties of GOLOG Programs over Local-Effect Actions". In: *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*. Ed. by Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan. Vol. 263. Frontiers in Artificial Intelligence and Applications. IOS Press, 2014, pp. 939–944. URL: `https://doi.org/10.3233/978-1-61499-419-0-939`.

  Furthermore, we show that the verification problem is 2EXPTIME-complete if

  $$\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}, \text{ and}$$

  CO-N2EXPTIME-complete if $\mathcal{L} = \mathcal{ALCQIO}$.

- The class of programs over actions with only local effects is rather inexpressive, because only a fixed finite number of named objects is affected by an action execution. In *Chapter 5*, we push the decidability border further towards pick-free $\mathcal{L}$-ConGolog programs over $\mathcal{L}$-definable actions with possibly non-local effects. We prove that non-local action effects lead to undecidability of verification already for the rather inexpressive base logic $\mathcal{ELI}_\bot$. Decidability can be regained by imposing syntactical restrictions on the effect definitions of the actions. One decidable class is obtained by disallowing cyclic dependencies between fluents and another one by allowing only quantifier-free formulas for describing the effects. Decidability is shown by generalizing the abstraction technique from Chapter 4. The base logic $\mathcal{L}$ in this chapter is the DL that covers full C^2. Again, we consider temporal specifications formulated in CTL* over $\mathcal{L}$-axioms. The results are based on the publication

[ZC16] Benjamin Zarrieß and Jens Claßen. “Decidable Verification of Golog Programs over Non-Local Effect Actions”. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*. Ed. by Dale Schuurmans and Michael P. Wellman. AAAI Press, 2016, pp. 1109–1115. URL: `http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12283`.

- In *Chapter 6*, we extend $\mathcal{ALCO}$-action theories over local effect actions with a simple notion of sensing. $\mathcal{ALCO}$ is chosen as a simple prototypical DL. We embed this language into the epistemic Situation Calculus $\mathcal{ES}$ [LL04; LL11] to justify our semantics. The epistemic projection problem with projection queries formulated in the epistemic DL $\mathcal{ALCOK}$ [Don+98] is shown to be ExpTime-complete. Thus, the problem is not harder than standard reasoning in $\mathcal{ALCO}$. The results in this chapter are published in

  [ZC15b] Benjamin Zarrieß and Jens Claßen. “Verification of Knowledge-Based Programs over Description Logic Actions”. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 3278–3284.

- In *Chapter 7*, we consider *knowledge-based ConGolog programs* over actions defined in $\mathcal{ALCO}$-action theories with sensing that we have introduced in the previous chapter. Tests in the program are formulated in $\mathcal{ALCOK}$, and we consider CTL* properties over $\mathcal{ALCOK}$-axioms. A decidability result for the corresponding verification problem is obtained. For some fragments we are able to show tight complexity results. Among those fragments are also fragments with an ExpTime-complete verification problem. Thus, under suitable restrictions the verification problem can be simpler than in the non-epistemic case. Moreover, we are able to reintroduce a restricted variant of the pick operator. We show that for programs with the restricted pick operator the verification problem reduces to the ground action case. Some of the results in this chapter are published in

  [ZC15b] Benjamin Zarrieß and Jens Claßen. “Verification of Knowledge-Based Programs over Description Logic Actions”. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 3278–3284.

- In *Chapter 8*, we conclude and give some directions for future work.

Chapter 2

Preliminaries

In this chapter, the basic definitions used in this thesis are provided. We define first-order dynamical systems and our notion of definability of actions (Section 2.1), basic notions of description logics and action theories based on them (Section 2.2), transition systems and temporal logic (Section 2.3) and ConGolog programs and the verification problem (Section 2.4).

2.1 First-Order Dynamical Systems

In this section we introduce a general model of first-order dynamical systems. It is inspired by Pednault's state-transition model of action introduced in [Ped94]. In our setting states are given as first-order relational structures (FO interpretations) and the meaning of an action is described as a transition relation on interpretations.

We consider standard *first-order logic with equality* (*FO* for short) over a fixed vocabulary.

Definition 2.1. The vocabulary consists of the following countably infinite and pairwise disjoint sets of:

- *predicate names* $\mathsf{N_F} = \{F, F_0, F_1, \ldots\}$ and each predicate name $F \in \mathsf{N_F}$ is associated with an *arity* $\mathsf{ar}(F) \in \mathbb{N}$ and $\mathsf{ar}(F) > 0$;
- *variable names* $\mathsf{N_V} = \{x, y, x_0, x_1, \ldots\}$ and
- *object names* $\mathsf{N_O} = \{o, o_1, o_2, \ldots\}$.

We assume that for each $n \in \mathbb{N}$, $n > 0$ countably infinitely many predicate names $F \in \mathsf{N_F}$ exist with $\mathsf{ar}(F) = n$. And we assume that the set of all variable names is linearly ordered.

The set of all *object terms* is given by $\mathsf{N_V} \cup \mathsf{N_O}$.

We use the notation $\bar{x}$, $\bar{o}$ and $\bar{t}$ to denote tuples of variable names, object names and object terms, respectively. ▲

The syntax of FO formulas over the vocabulary $\mathsf{N_F}, \mathsf{N_V}, \mathsf{N_O}$ is defined in the usual way.

Definition 2.2. An *FO formula* ϕ is built according the following syntax rule

$$\phi ::= F(t_1, \ldots, t_n) \mid t_1 \approx t_2 \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \exists x.\phi \mid \forall x.\phi,$$

where $F \in \mathsf{N_F}$ stands for a predicate name with arity $\mathsf{ar}(F) = n$, $t_1, t_2, \ldots, t_n$ are object terms and $x \in \mathsf{N_V}$ is a variable name. The Boolean connectives "$\rightarrow$" (implication) and "$\equiv$" (equivalence) are defined as the usual abbreviations.

An FO formula ψ without any free occurrences of variables is called *sentence*. ▲

The semantics is given in terms of *first-order interpretations* (*interpretations* for short).

Definition 2.3. A first-order interpretation $\mathcal{I}$ is a pair of the form $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta_\mathcal{I}$ is a non-empty domain and $\cdot^\mathcal{I}$ a function that maps each predicate name $F \in \mathsf{N_F}$ to an $\mathsf{ar}(F)$-ary relation over $\Delta_\mathcal{I}$, denoted by $F^\mathcal{I} \subseteq (\Delta_\mathcal{I})^{\mathsf{ar}(F)}$, and each object name $o \in \mathsf{N_O}$ to an element $o^\mathcal{I} \in \Delta_\mathcal{I}$. The set $F^\mathcal{I}$ is called the *extension of F under $\mathcal{I}$*.

A *variable assignment μ for $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$* is a total function of the form $\mu : \mathsf{N_V} \to \Delta_\mathcal{I}$. Let $t \in \mathsf{N_V} \cup \mathsf{N_O}$. We define

$$t^{\mathcal{I},\mu} := \begin{cases} \mu(t), & \text{for } t \in \mathsf{N_V}; \\ t^\mathcal{I}, & \text{for } t \in \mathsf{N_O}. \end{cases}$$

Let $x \in \mathsf{N_V}$ and $d \in \Delta_\mathcal{I}$ and μ a variable assignment for $\mathcal{I}$. $\mu[x \mapsto d]$ is a variable assignment for $\mathcal{I}$ obtained from μ by mapping x to d. Let ϕ be an FO formula. *Satisfaction of ϕ in $\mathcal{I}, \mu$*, denoted by $\mathcal{I}, \mu \models \phi$, is defined by induction on the structure of ϕ as follows

$$\begin{array}{ll}
\mathcal{I}, \mu \models F(t_1, \dots, t_n) & \text{iff } (t_1^{\mathcal{I},\mu}, \dots, t_n^{\mathcal{I},\mu}) \in F^\mathcal{I}; \\
\mathcal{I}, \mu \models t_1 \approx t_2 & \text{iff } t_1^{\mathcal{I},\mu} = t_2^{\mathcal{I},\mu}; \\
\mathcal{I}, \mu \models \neg\phi_1 & \text{iff } \mathcal{I}, \mu \not\models \phi_1; \\
\mathcal{I}, \mu \models \phi_1 \wedge \phi_2 & \text{iff } \mathcal{I}, \mu \models \phi_1 \text{ and } \mathcal{I}, \mu \models \phi_2; \\
\mathcal{I}, \mu \models \phi_1 \vee \phi_2 & \text{iff } \mathcal{I}, \mu \models \phi_1 \text{ or } \mathcal{I}, \mu \models \phi_2; \\
\mathcal{I}, \mu \models \exists x.\phi_1 & \text{iff } \mathcal{I}, \mu[x \mapsto d] \models \phi_1 \text{ for some } d \in \Delta_\mathcal{I}; \\
\mathcal{I}, \mu \models \forall x.\phi_1 & \text{iff } \mathcal{I}, \mu[x \mapsto d] \models \phi_1 \text{ for all } d \in \Delta_\mathcal{I}.
\end{array}$$

For an FO sentence ψ we write $\mathcal{I} \models \psi$ to denote that ψ is satisfied in $\mathcal{I}$. A *knowledge base* is a finite set of FO sentences. An interpretation $\mathcal{I}$ is *a model of a knowledge base KB*, written as $\mathcal{I} \models \mathit{KB}$, iff each element of *KB* is satisfied in $\mathcal{I}$. $\mathcal{M}(\mathit{KB})$ denotes the *set of all models of a knowledge base KB*.

Let ϕ be an FO formula and $x_1, \dots, x_n$ exactly the variable names that occur free in ϕ with $x_i < x_{i+1}$ for all $i \in \{1, \dots, n-1\}$. The *extension of ϕ under $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$*, denoted by $\phi^\mathcal{I}$, is defined as follows

$$\phi^\mathcal{I} := \{(d_1, \dots, d_n) \in (\Delta_\mathcal{I})^n \mid \mathcal{I}, \mu[x_1 \mapsto d_1] \cdots [x_n \mapsto d_n] \models \phi \text{ for some } \mu \text{ for } \mathcal{I}\}.$$

▲

We distinguish classes of interpretations satisfying certain assumptions regarding the interpretation of object names.

Definition 2.4. An interpretation $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ satisfies the *unique name assumption* (UNA) for object names iff for any two distinct individuals $o, o' \in \mathsf{N_O}$ it holds that $o^\mathcal{I} \neq o'^\mathcal{I}$. $\mathcal{I}$ satisfies the *standard name assumption* (SNA) iff $\Delta_\mathcal{I} = \mathsf{N_O}$ and $o^\mathcal{I} = o$ for all $o \in \mathsf{N_O}$. ▲

Next, we define the syntax of *actions*. Syntactically, actions are defined as terms built from action names with an arity that possibly have object terms as arguments.

Definition 2.5. Let $\mathsf{N_A}$ be a countable infinite set of *action names* disjoint with $\mathsf{N_F} \cup \mathsf{N_V} \cup \mathsf{N_O}$. Each action name $\alpha \in \mathsf{N_A}$ has an arity, denoted by $\mathsf{ar}(\alpha) \in \mathbb{N}$, such that for each arity $k = 0, 1, 2, \ldots$ there are infinitely many action names in $\mathsf{N_A}$ with arity k.

The *set of all action terms* over $\mathsf{N_A}$, $\mathsf{N_V}$ and $\mathsf{N_O}$, denoted by $\mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$, is defined as the smallest set satisfying the following conditions:

- $\alpha \in \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$ for any $\alpha \in \mathsf{N_A}$with $\mathsf{ar}(\alpha) = 0$, and
- if $\alpha \in \mathsf{N_A}$ with $\mathsf{ar}(\alpha) > 0$ and $\bar{t} \in (\mathsf{N_V} \cup \mathsf{N_O})^{\mathsf{ar}(\alpha)}$, then $\alpha(\bar{t}) \in \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$.

Action terms that contain variable names can be instantiated. A *variable mapping* ν is a total function of the form $\nu : \mathsf{N_V} \cup \mathsf{N_O} \to \mathsf{N_O}$ that maps object terms to object names such that $\nu(o) = o$ for all $o \in \mathsf{N_O}$.

Let $\alpha(t_1, \ldots, t_k)$ be an action term built using the action name $\alpha \in \mathsf{N_A}$ with $\mathsf{ar}(\alpha) = k$, and ν a variable mapping. The ground action term $\alpha(\nu(t_1), \ldots, \nu(t_k))$ is called the *ground instantiation* of $\alpha(t_1, \ldots, t_k)$ w.r.t. ν.

For a set of action terms $\mathsf{Act} \subseteq \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$, the set $\mathsf{ground}(\mathsf{Act})$ denotes the *set of all possible ground instantiations* of action terms in Act. To simplify the notation we will sometimes omit the tuple of arguments when writing an action term. The symbol α will also stand for an action term. It will be clear from the context whether an action name or action term is meant. We will often use the bold symbols $\boldsymbol{\alpha}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1, \ldots, \boldsymbol{\beta}, \boldsymbol{\beta}_0, \ldots$ to denote ground action terms. ▲

Semantically, a ground action term is understood as a name of an operator that updates first-order interpretations. The semantics of actions will be given in terms of a transition relation between interpretations. For this purpose, the notion of a *first-order dynamical system (FO-DS for short)* is introduced. In our view of a dynamical system an interpretation completely describes the current state of the world. An FO-DS defines executability of a ground action in an interpretation and how an interpretation is changing as the result of an action execution. The state space of an FO-DS is a set of interpretations. We assume that there are only finitely many relevant (possibly non-ground) action terms and finitely many relevant predicate names called *fluents* whose extensions are possibly subject to changes.

Definition 2.6. A *first-order dynamical system* (FO-DS) $\mathfrak{D}$ is a tuple

$$\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle, \mathsf{Pre})$$

that consists of the following components:

- $\mathbb{I}$ is a (possibly uncountable) set of interpretations called the *state space* of $\mathfrak{D}$;
- $\mathbb{I}_{\mathsf{ini}} \subseteq \mathbb{I}$ is the set of initial states;
- $\mathcal{F} \subset \mathsf{N_F}$ is a finite set of relevant predicate names (also called *fluents*);
- $\mathsf{Act} \subset \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$ is a finite set of *relevant action terms*;
- $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$ is a pair of two functions, where add (called *positive effect function*) and del (called *negative effect function*) are total functions that map each interpretation $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}}) \in \mathbb{I}$, ground action $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ and fluent $F \in \mathcal{F}$ to sets

$$\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F) \subseteq (\Delta_{\mathcal{I}})^{\mathsf{ar}(F)} \text{ and } \mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F) \subseteq (\Delta_{\mathcal{I}})^{\mathsf{ar}(F)};$$

- Pre is a binary *precondition relation* with $\mathsf{Pre} \subseteq \mathbb{I} \times \mathsf{ground}(\mathsf{Act})$.

The sets $\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F)$ and $\mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F)$ are called *add-sets* and *delete-sets*, respectively. ▲

In the following we often write $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ to denote an FO-DS and do not name add and del explicitly.

The effect functions in $\mathcal{E}$ *completely* describe the effect of executing a ground action in an interpretation. Given an interpretation, a ground action and an n-ary fluent, add comprises the set of n-tuples from the domain that are added to the extension of the respective fluent predicate and del the set of n-tuples that are deleted from it. Before the transition relation on interpretations induced by $\mathcal{E}$ is defined an example for an FO-DS is given.

Example 2.7. An example domain with devices and power supplies is considered. In the following, we describe a first-order dynamical system for this domain. We use a countably infinite set of object names

$$\mathsf{N_O} := \{\mathsf{d}_0, \mathsf{d}_1, \mathsf{d}_2, \ldots\} \cup \{\mathsf{p}_0, \mathsf{p}_1, \mathsf{p}_2, \ldots\}.$$

As state space $\mathbb{I}$ we choose the set of all first-order structures satisfying the SNA, i.e. the set of all interpretations having $\mathsf{N_O}$ as their domain.

A set $\mathcal{F}$ of four fluent predicates is chosen as follows: *ConnectedTo* (*ConTo* for short) is binary and interpreted as a subset of $\mathsf{N_O} \times \mathsf{N_O}$ and *PowerSupply* (*PowerS* for short) *Device* (*Dev* for short) and *On* are unary fluents and interpreted as subsets of $\mathsf{N_O}$.

The initial state space $\mathbb{I}_{\mathsf{ini}}$ consists of those interpretations $\mathcal{I} = (\mathsf{N_O}, \cdot^{\mathcal{I}})$ satisfying the following constraints

$$\mathsf{d}_0 \in \mathit{Dev}^{\mathcal{I}},\ \mathsf{p}_0 \in \mathit{PowerS}^{\mathcal{I}},\ \mathsf{d}_0 \notin \mathit{On}^{\mathcal{I}},\ (\mathsf{d}_0, \mathsf{p}_0) \notin \mathit{ConTo}^{\mathcal{I}}.$$

The set of action terms Act in this example consists of the ground terms

$$\texttt{disable}(\mathsf{p}_0), \texttt{turn-on}(\mathsf{d}_0) \text{ and } \texttt{connect}(\mathsf{d}_0, \mathsf{p}_0).$$

We represent the effect functions $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$ by providing explicit definitions of the corresponding add- and delete-sets. The action $\texttt{turn-on}(\mathsf{d}_0)$ turns d_0 on, if it is connected to a power supply, and changes nothing else. For all $\mathcal{I} \in \mathbb{I}$ we have

$$\mathsf{add}(\mathcal{I}, \texttt{turn-on}(\mathsf{d}_0), \mathit{On}) := \begin{cases} \{\mathsf{d}_0\} & \text{if there is a } y \in \mathit{PowerS}^{\mathcal{I}} \text{ with } (\mathsf{d}_0, y) \in \mathit{ConTo}^{\mathcal{I}}, \\ \emptyset & \text{otherwise.} \end{cases}$$

All delete-sets and the other remaining add-sets involving $\texttt{turn-on}(\mathsf{d}_0)$ are empty. The action $\texttt{connect}(\mathsf{d}_0, \mathsf{p}_0)$ simply connects d_0 and p_0 and changes nothing else. We have for all $\mathcal{I} \in \mathbb{I}$

$$\mathsf{add}(\mathcal{I}, \texttt{connect}(\mathsf{d}_0, \mathsf{p}_0), \mathit{ConTo}) := \{(\mathsf{d}_0, \mathsf{p}_0)\}$$

and in the remaining cases the add- and delete-sets are empty. After $\texttt{disable}(\mathsf{p}_0)$, p_0 is no longer a power supply

$$\mathsf{del}(\mathcal{I}, \texttt{disable}(\mathsf{p}_0), \mathit{PowerS}) := \{\mathsf{p}_0\} \text{ for all } \mathcal{I} \in \mathbb{I},$$

and all devices with p_0 as their only connected power supply are no longer members of the set *On*. Thus, we have the following delete-set for all $\mathcal{I} \in \mathbb{I}$:

$$\begin{aligned}\mathsf{del}(\mathcal{I}, \texttt{disable}(\mathsf{p}_0), \mathit{On}) := \{x \in \mathit{Dev}^{\mathcal{I}} \mid {} & (x, \mathsf{p}_0) \in \mathit{ConTo}^{\mathcal{I}} \text{ and} \\ & \text{there is no } y \in \mathit{PowerS}^{\mathcal{I}} \text{ with} \\ & y \neq \mathsf{p}_0 \text{ and } (x, y) \in \mathit{ConTo}^{\mathcal{I}}\}.\end{aligned}$$

No other fluent predicate is affected by $\texttt{disable}(\mathsf{p}_0)$. The remaining sets involving the action $\texttt{disable}(\mathsf{p}_0)$ are empty.

Preconditions are given as follows: for all $\mathcal{I} \in \mathbb{I}$ we have

$$\begin{aligned}(\mathcal{I}, \texttt{turn-on}(\mathsf{d}_0)) \in \mathsf{Pre} &\text{ iff } \mathsf{d}_0 \in \mathit{Dev}^{\mathcal{I}};\\ (\mathcal{I}, \texttt{connect}(\mathsf{d}_0, \mathsf{p}_0)) \in \mathsf{Pre} &\text{ iff } \mathsf{d}_0 \in \mathit{Dev}^{\mathcal{I}};\\ (\mathcal{I}, \texttt{disable}(\mathsf{p}_0)) \in \mathsf{Pre} &\text{ iff } \mathsf{p}_0 \in \mathit{PowerS}^{\mathcal{I}}.\end{aligned}$$

▲

The effect functions $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$ of an FO-DS uniquely determine a *transition relation* on the state space that is defined as follows.

Definition 2.8. Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS as above with the pair of effect functions $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$, and let $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ be a ground action, and $\mathcal{I}, \mathcal{I}' \in \mathbb{I}$ two interpretations. We say that $\boldsymbol{\alpha}$ *transforms* $\mathcal{I}$ *into* $\mathcal{I}'$, written as

$$\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{I}',$$

iff the following conditions are satisfied:

- $\Delta_{\mathcal{I}} = \Delta_{\mathcal{I}'}$;
- $F^{\mathcal{I}'} = (F^{\mathcal{I}} \setminus \mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F)) \cup \mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F)$ for all fluents $F \in \mathcal{F}$;
- $X^{\mathcal{I}'} = X^{\mathcal{I}}$ for all $X \in \mathsf{N_O} \cup (\mathsf{N_F} \setminus \mathcal{F})$.

We also use the notation $\Rightarrow_{\mathfrak{D}}$ for transformations caused by executing a sequence of ground actions. Let $\sigma = \boldsymbol{\alpha}_0 \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \mathsf{ground}(\mathsf{Act})^*$ for some $n > 0$ be a ground action sequence, and $\mathcal{J}, \mathcal{Y} \in \mathbb{I}$ two interpretations. We write

$$\mathcal{J} \Rightarrow^{\sigma}_{\mathfrak{D}} \mathcal{Y}$$

iff there exists a sequence of interpretations $\mathcal{I}_0, \ldots, \mathcal{I}_{n+1}$ with $\mathcal{I}_j \in \mathbb{I}$ for all $j \in \{0, \ldots, n+1\}$; $\mathcal{I}_i \Rightarrow^{\boldsymbol{\alpha}_i}_{\mathfrak{D}} \mathcal{I}_{i+1}$ for all $i \in \{0, \ldots, n\}$; and $\mathcal{I}_0 = \mathcal{J}$ and $\mathcal{I}_{n+1} = \mathcal{Y}$. ▲

For any ground action $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ and interpretation $\mathcal{I} \in \mathbb{I}$ the successor interpretation $\mathcal{I}' \in \mathbb{I}$ with $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{I}'$ is unique, if it exists. It is possible that an interpretation $\mathcal{I}'$ satisfying the three conditions of $\Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}}$ does not exist in $\mathbb{I}$. In the following we only consider FO-DSs where for each $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ the transition relation $\Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}}$ is right-total, i.e. a successor interpretation always exists in the state space $\mathbb{I}$.

Example 2.9 (2.7 continued). An interpretation $\mathcal{I}_0 = (\mathsf{N_O}, \cdot^{\mathcal{I}_0})$ (satisfying the SNA) from the initial state space of the FO-DS described in Example 2.7 interprets the relevant fluent predicates as follows:

$$\begin{gathered} \mathit{Dev}^{\mathcal{I}_0} := \{\mathsf{d}_0, \mathsf{d}_1, \mathsf{d}_2, \ldots\}, \quad \mathit{PowerS}^{\mathcal{I}_0} := \{\mathsf{p}_0, \mathsf{p}_1\}, \\ \mathit{ConTo}^{\mathcal{I}_0} := \{(\mathsf{d}_1, \mathsf{p}_0), (\mathsf{d}_1, \mathsf{p}_1), (\mathsf{d}_2, \mathsf{p}_0)\}, \quad \mathit{On}^{\mathcal{I}_0} := \{\mathsf{d}_1, \mathsf{d}_2\}. \end{gathered}$$

Furthermore, an interpretation $\mathcal{J}_0 = (\mathsf{N_O}, \cdot^{\mathcal{J}_0})$ is given by the following sets

$$\mathit{Dev}^{\mathcal{J}_0} := \{\mathsf{d}_0, \mathsf{d}_1, \mathsf{d}_2, \ldots\},\ \mathit{PowerS}^{\mathcal{J}_0} := \{\mathsf{p}_0, \mathsf{p}_1\},\ \mathit{ConTo}^{\mathcal{J}_0} := \{(\mathsf{d}_0, \mathsf{p}_1)\},\ \mathit{On}^{\mathcal{J}_0} := \emptyset.$$

We consider the execution of the action sequence

$$\sigma = \texttt{connect}(\mathsf{d}_0, \mathsf{p}_0)\ \texttt{turn-on}(\mathsf{d}_0)\ \texttt{disable}(\mathsf{p}_0).$$

in $\mathcal{I}_0$ and $\mathcal{J}_0$. It leads to interpretations $\mathcal{I}_1$ and $\mathcal{J}_1$ with $\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\sigma} \mathcal{I}_1$ and $\mathcal{J}_0 \Rightarrow_{\mathfrak{D}}^{\sigma} \mathcal{J}_1$, where $\mathfrak{D}$ is the FO-DS described in Example 2.7 and we have

$$\mathit{PowerS}^{\mathcal{I}_1} := \{\mathsf{p}_1\},\ \mathit{ConTo}^{\mathcal{I}_1} := \{(\mathsf{d}_1, \mathsf{p}_0), (\mathsf{d}_1, \mathsf{p}_1), (\mathsf{d}_2, \mathsf{p}_0), (\mathsf{d}_0, \mathsf{p}_0)\},\ \mathit{On}^{\mathcal{I}_1} := \{\mathsf{d}_1\}$$

and

$$\mathit{PowerS}^{\mathcal{J}_1} := \{\mathsf{p}_1\},\ \mathit{ConTo}^{\mathcal{J}_1} := \{(\mathsf{d}_0, \mathsf{p}_1), (\mathsf{d}_0, \mathsf{p}_0)\},\ \mathit{On}^{\mathcal{J}_1} := \{\mathsf{d}_0\}.$$

The interpretation of *Dev* remains unchanged in both cases. Note that in $\mathcal{I}_0$ the device d_1 is also connected to the power supply p_1. Thus, the device is not affected by disabling p_0 and it stays on in $\mathcal{I}_1$. The same applies to the device d_0 in $\mathcal{J}_0$ and $\mathcal{J}_1$. ▲

To simplify the technical treatment we have defined the effect functions add and del as total functions that are independent of the possibility relation. Another choice would have been to consider them as partial functions that are only defined if the ground action is possible in the interpretation.

It is possible that the two sets $\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F)$ and $\mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F)$ overlap. In this case the chosen "add-after-delete semantics" of the transition relation gives precedence to the corresponding add-set. For example, in PDDL this is handled in the same way. However, an unambiguous consistent representation of an FO-DS should ensure that add-sets and delete-sets are always disjoint.

We assume that $\mathcal{F}$ consists of all the predicate names that are relevant for the domain including also those predicates that are assumed to be rigid. In this case the functions add and del always yield the empty set for a rigid predicate name.

Although our model of a first-order dynamical system is quite general, there are several simplifying assumptions we have adopted (see Section 1.1):

- Actions are deterministic. There always exists a unique successor interpretation.

- We consider only instantaneous actions. Actions do not have any duration.
- In each step only one action can be executed. There is no simultaneous execution of more than one action.
- The possibility of a ground action and its effects only depend on the *current* state.

Next, we define projection and executability.

Definition 2.10. Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be as above, φ a FO sentence (that mentions only predicate names from $\mathcal{F}$), and $\sigma = \boldsymbol{\alpha}_0\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \mathsf{ground}(\mathsf{Act})^*$ a ground action sequence.

We say that φ *is true in* $\mathfrak{D}$ *after executing* σ iff for all interpretations $\mathcal{I} \in \mathbb{I}_{\mathsf{ini}}$ the transformed interpretation $\mathcal{I}' \in \mathbb{I}$ with $\mathcal{I} \Rightarrow^{\sigma}_{\mathfrak{D}} \mathcal{I}'$ satisfies $\mathcal{I}' \models \varphi$.

We say that σ *is executable in an interpretation* $\mathcal{I} \in \mathbb{I}$ iff there exists a sequence of interpretations $\mathcal{I}_0, \ldots, \mathcal{I}_n$ with $\mathcal{I}_j \in \mathbb{I}$ for all $j \in \{0, \ldots, n\}$, $\mathcal{I}_i \Rightarrow^{\boldsymbol{\alpha}_i}_{\mathfrak{D}} \mathcal{I}_{i+1}$ and $(\mathcal{I}_i, \boldsymbol{\alpha}_i) \in \mathsf{Pre}$ for all $i \in \{0, \ldots, n-1\}$ and $\mathcal{I}_0 = \mathcal{I}$.

We say that σ *is executable in* $\mathfrak{D}$ iff σ is executable in all interpretations $\mathcal{I} \in \mathbb{I}_{\mathsf{ini}}$. ▲

So far we haven't made any assumptions about a particular representation of the pair of effect functions $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$ and the precondition relation Pre. In Example 2.7 we have described $\mathcal{E}$ und Pre using standard set notation. We now define a representation where it is ensured that effects and preconditions of actions have a certain structure that is definable in first-order logic. The goal of such a representation is to enable first-order reasoning about actions. One basic assumption we make is that preconditions and action effects are context-dependent w.r.t. a *context* that is definable as a finite set of FO sentences. If a finite set of ground actions is considered, then a corresponding context consists of all relevant properties (formulated as FO sentences) that are needed to characterize preconditions and effects of actions in a certain state. (Recall that states are first-order interpretations in our view.) Furthermore, the *static type* (w.r.t. a context) of a state is the subset of the context consisting of exactly those properties that are indeed true in the state.

First, the notion of a context and the static type of an interpretation w.r.t. a context is defined formally.

Definition 2.11. A *context*, denoted by $\mathcal{C}$, is a finite set of FO sentences that is closed under negation, i.e. for each $\psi \in \mathcal{C}$ we have also $\neg\psi \in \mathcal{C}$ (modulo elimination of double negation).

Let $\mathcal{C}$ be a context and $\mathcal{F} \subset \mathsf{N_F}$ a finite set of relevant predicate names. We say that $\mathcal{C}$ is *a context over* $\mathcal{F}$ iff the sentences in $\mathcal{C}$ mention only predicate names from $\mathcal{F}$.

A *static type w.r.t.* $\mathcal{C}$ is a maximal (w.r.t. $\subseteq$) subset of $\mathcal{C}$ that has a model. The *set of all static types w.r.t.* $\mathcal{C}$ is denoted by $\mathfrak{S}_{\mathcal{C}}$. Let $\mathcal{I}$ be an interpretation. The *static type of* $\mathcal{I}$ *w.r.t.* $\mathcal{C}$, denoted by $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I})$, is given by

$$\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) := \{\psi \in \mathcal{C} \mid \mathcal{I} \models \psi\}.$$

▲

Obviously, it holds that $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) \in \mathfrak{S}_{\mathcal{C}}$ for all interpretations $\mathcal{I}$ and for all types $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ we have $\mathfrak{s} = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I})$ for some $\mathcal{I}$.

Example 2.12 (2.7 and 2.9 continued)**.** For the FO-DS in Example 2.7 the following sentences are relevant

$$\begin{aligned}
\varphi_1 &:= \mathit{Dev}(\mathsf{d}_0),\\
\varphi_2 &:= \mathit{PowerS}(\mathsf{p}_0),\\
\varphi_3 &:= \exists x.\,(\mathit{ConTo}(\mathsf{d}_0, x) \wedge \mathit{PowerS}(x)),\\
\varphi_4 &:= \mathit{On}(\mathsf{d}_0), \text{ and}\\
\varphi_5 &:= \mathit{ConTo}(\mathsf{d}_0, \mathsf{p}_0).
\end{aligned}$$

φ_1 is relevant for the precondition of `turn-on`(d_0) and `connect`$(\mathsf{d}_0, \mathsf{p}_0)$, φ_2 is the precondition of `disable`(p_0), satisfaction of φ_3 is relevant for the outcome of `turn-on`(d_0) and φ_4 and φ_5 are needed to describe the initial situation. A context $\mathcal{C}$ is given by

$$\mathcal{C} := \{\varphi_1, \neg\varphi_1, \varphi_2, \neg\varphi_2, \varphi_3, \neg\varphi_3, \varphi_4, \neg\varphi_4, \varphi_5, \neg\varphi_5\}.$$

Now consider the interpretations $\mathcal{I}_0$, $\mathcal{I}_1$, $\mathcal{J}_0$ and $\mathcal{J}_1$ from Example 2.9. The static types of them w.r.t. $\mathcal{C}$ are as follows:

$$\begin{aligned}
\text{s-type}_{\mathcal{C}}(\mathcal{I}_0) = \text{s-type}_{\mathcal{C}}(\mathcal{J}_0) &= \{\varphi_1, \varphi_2, \neg\varphi_3, \neg\varphi_4, \neg\varphi_5\};\\
\text{s-type}_{\mathcal{C}}(\mathcal{I}_1) &= \{\varphi_1, \neg\varphi_2, \neg\varphi_3, \neg\varphi_4, \varphi_5\};\\
\text{s-type}_{\mathcal{C}}(\mathcal{J}_1) &= \{\varphi_1, \neg\varphi_2, \varphi_3, \varphi_4, \varphi_5\}.
\end{aligned}$$

▲

Based on the notion of static types we define several conditions for an appropriate representation of action effects and preconditions.

Definition 2.13. Let $\boldsymbol{A}$ be a *finite* set of ground action terms. An *FO-admissible representation of $\boldsymbol{A}$* is given as a tuple of the form

$$\Sigma_{\boldsymbol{A}} = (\mathit{KB}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}}),$$

that consists of the following components:

- a finite set of relevant fluents $\mathcal{F}$,
- an FO knowledge base KB over $\mathcal{F}$ as an incomplete description of the initial situation,
- a context $\mathcal{C}$ over $\mathcal{F}$ with $\varphi \in \mathcal{C}$ for all sentences $\varphi \in \mathit{KB}$, and
- a pair of effectively computable functions $(\mathsf{E}^+, \mathsf{E}^-)$, that map each tuple

  $$(\mathfrak{s}, \boldsymbol{\alpha}, F) \in \mathfrak{S}_{\mathcal{C}} \times \boldsymbol{A} \times \mathcal{F}$$

  to an FO formula

  $$\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F] \text{ and } \mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F], \text{ respectively,}$$

  that is formulated over $\mathcal{F}$ and has exactly $\mathsf{ar}(F)$ many free variables, and
- a decidable relation $\mathsf{Pre}_{\mathcal{C}} \subseteq \mathfrak{S}_{\mathcal{C}} \times \boldsymbol{A}$.

▲

An FO-admissible representation induces a first-order dynamical system.

Definition 2.14. Let $\boldsymbol{A}$ be a finite set of ground action terms and

$$\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$$

an FO-admissible representation of $\boldsymbol{A}$. The *FO-DS induced by* $\Sigma_{\boldsymbol{A}}$, denoted by $\mathfrak{D}(\Sigma_{\boldsymbol{A}})$, is an FO-DS the form

$$\mathfrak{D}(\Sigma_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(KB), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}),$$

where the state space is given by

$$\mathbb{I} := \left\{ \mathcal{J} \;\middle|\; \text{there exists } \mathcal{I} \in \mathcal{M}(KB) \text{ with } \Delta_{\mathcal{J}} = \Delta_{\mathcal{I}} \text{ and } o^{\mathcal{J}} = o^{\mathcal{I}} \text{ for all } o \in \mathsf{N_O} \right\};$$

and where $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$ and Pre are defined as follows

- for all $\mathcal{I} \in \mathbb{I}$ and all $(\boldsymbol{\alpha}, F) \in \boldsymbol{A} \times \mathcal{F}$ we have

$$\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F) := \left(\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]\right)^{\mathcal{I}} \text{ and } \mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F) := \left(\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]\right)^{\mathcal{I}},$$

  with $\mathfrak{s} = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I})$, and

- for all $\mathcal{I} \in \mathbb{I}$ and all $\boldsymbol{\alpha} \in \boldsymbol{A}$ it holds that

$$(\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}), \boldsymbol{\alpha}) \in \mathsf{Pre}_{\mathcal{C}} \text{ iff } (\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}.$$

▲

In an FO-admissible representation of a finite set of ground actions we require that the add-sets and delete-sets are first-order definable and context-dependent w.r.t. a context that is defined as a set of FO sentences. The definitions of the add-sets and delete-sets in form of FO formulas are effectively computable given the static type, the ground action and the fluent. Note that the restriction on E^+ and E^- does not guarantee a solution to the frame problem. For example, if both functions are explicitly given in terms of a complete table with formulas for each tuple $(\mathfrak{s}, \boldsymbol{\alpha}, F) \in \mathfrak{S}_{\mathcal{C}} \times \boldsymbol{A} \times \mathcal{F}$, then this representation does not count as a solution to the frame problem.

Moreover, it is decidable whether a ground action is executable in an interpretation that is abstracted in terms of its static type. The initial states are exactly the models of *KB* and the state space of $\mathfrak{D}(\Sigma_{\boldsymbol{A}})$ is the set of all interpretations for which a model of the initial KB exists that has the same domain and agrees with the interpretation on the mapping of object names. Since actions do not affect the domain and interpretation of object names, it is guaranteed that the induced transition relation $\Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})}$ on the state space is right-total for all $\boldsymbol{\alpha} \in \boldsymbol{A}$.

The notion of an FO-admissible representation can be refined in case a syntactical fragment of first-oder logic is considered as a base logic.

Definition 2.15. Let $\mathcal{L}$ be a syntactical fragment of FO and $\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ an FO-admissible representation for a finite set of ground actions $\boldsymbol{A}$.

We say that $\Sigma_{\boldsymbol{A}}$ is an *$\mathcal{L}$-admissible representation* of $\boldsymbol{A}$ iff the following conditions are satisfied

- the fluents in $\mathcal{F}$ satisfy the requirements of $\mathcal{L}$ regarding the arity,
- $\mathcal{C}$ and *KB* consist of $\mathcal{L}$-axioms, and
- the computable functions E^+ and E^- map each tuple $(\mathfrak{s}, \boldsymbol{\alpha}, F) \in \mathfrak{S}_\mathcal{C} \times \boldsymbol{A} \times \mathcal{F}$ to formulas $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$ and $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$ that are formulated in $\mathcal{L}$.

▲

Next, we define *$\mathcal{L}$-definability* of a finite set of ground actions in an FO-DS.

Definition 2.16. Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS and $\boldsymbol{A} \subseteq \mathsf{ground}(\mathsf{Act})$ a finite set of ground actions. The *restriction of $\mathfrak{D}$ to $\boldsymbol{A}$* is the FO-DS of the form

$$\mathfrak{D}|_{\boldsymbol{A}} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \boldsymbol{A}, \mathcal{E}|_{\boldsymbol{A}}, \mathsf{Pre}_{\boldsymbol{A}}),$$

where $\mathbb{I}$, $\mathbb{I}_{\mathsf{ini}}$, and $\mathcal{F}$ are the same as in $\mathfrak{D}$ and $\mathcal{E}|_{\boldsymbol{A}}$ and $\mathsf{Pre}_{\boldsymbol{A}}$ denote the restrictions of $\mathcal{E}$ and Pre, respectively, to the actions in $\boldsymbol{A}$.

Let $\mathcal{L}$ be a syntactical fragment of FO and $\mathcal{C}$ an $\mathcal{L}$-context over $\mathcal{F}$. We say that $\boldsymbol{A}$ is *$\mathcal{L}$-definable in $\mathcal{D}$ w.r.t. $\mathcal{C}$* iff there exists an $\mathcal{L}$-admissible representation of the form

$$\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_\mathcal{C}) \text{ such that } \mathfrak{D}|_{\boldsymbol{A}} = \mathfrak{D}(\Sigma_{\boldsymbol{A}})$$

And, $\boldsymbol{A}$ is *$\mathcal{L}$-definable in $\mathfrak{D}$* iff there exists an $\mathcal{L}$-context $\mathcal{C}$ over $\mathcal{F}$ such that $\boldsymbol{A}$ is $\mathcal{L}$-definable in $\mathfrak{D}$ w.r.t. $\mathcal{C}$. ▲

Note that $\mathcal{L}$-definability of a set of ground actions in an FO-DS also implies that the set of initial states in the FO-DS is equal to the set of all models of some $\mathcal{L}$-KB.

Example 2.17 (2.7 and 2.12 continued)**.** The ground actions

$$\boldsymbol{A} := \{\texttt{connect}(\mathsf{d}_0, \mathsf{p}_0),\ \texttt{turn-on}(\mathsf{d}_0),\ \texttt{disable}(\mathsf{p}_0)\}$$

are FO-definable in the FO-DS described in Example 2.7 w.r.t. the context $\mathcal{C}$ given in Example 2.12. We briefly sketch (some parts of) a possible FO-admissible representation for $\boldsymbol{A}$. The initial state space corresponds to the set of all models of the sentence

$$\mathit{Dev}(\mathsf{d}_0) \land \mathit{PowerS}(\mathsf{p}_0) \land \neg \mathit{On}(\mathsf{d}_0) \land \neg \mathit{ConTo}(\mathsf{d}_0, \mathsf{p}_0).$$

Obviously, the add-sets and delete-sets of the actions are first-order definable. Functions E^+, E^- providing first-order formulas can be explicitly described. For instance, consider the action $\texttt{turn-on}(\mathsf{d}_0)$ and the fluent *On*. For all static types $\mathfrak{s} \in \mathfrak{S}_\mathcal{C}$ the add-set is defined by

$$\mathsf{E}^+[\mathfrak{s}, \texttt{turn-on}(\mathsf{d}_0), \mathit{On}] := \begin{cases} x \approx \mathsf{d}_0 & \text{if } \exists x. (\mathit{ConTo}(\mathsf{d}_0, x) \land \mathit{PowerS}(x)) \in \mathfrak{s}, \\ x \not\approx x & \text{otherwise.} \end{cases}$$

Next, consider the action $\texttt{disable}(\mathsf{p}_0)$ and the unary fluent predicate *On*. The delete-set formula $\mathsf{E}^-[\mathfrak{s}, \texttt{disable}(\mathsf{p}_0), \mathit{On}]$ is defined for all $\mathfrak{s} \in \mathfrak{S}_\mathcal{C}$ by

$$\mathit{Dev}(x) \land \mathit{ConTo}(x, \mathsf{p}_0) \land \neg \exists y. (\mathit{PowerS}(y) \land y \not\approx \mathsf{p}_0 \land \mathit{ConTo}(x, y)). \qquad (2.1)$$

It describes devices that are not connected to any power supply different from p_0. ▲

2.2 Description Logics and Action Languages

In this section, we consider first-order dynamical systems based on decidable *Description Logics* [Baa+10]. We first recall basic notion of DLs and then consider dynamical systems with a DL as its base logic.

2.2.1 Basic Notions of Description Logics

In this section we briefly recall basic notions of DLs that are relevant for this thesis.

Syntax and Semantics

We consider a decidable DL named $\mathcal{DL}$. It is as expressive as the *two-variable fragment of first-order logic with counting and equality*. The name "$\mathcal{DL}$" is just a short cut for an expressive umbrella DL. $\mathcal{DL}$ is very similar to the Description Logic introduced in [Bor96].

The usual naming scheme for the sub-logics is introduced later.

The two main ingredients in DLs are *concepts* and *roles*. Intuitively, a concept describes a set of objects and a role a binary relation between objects.

Definition 2.18 (syntax of concepts and roles)**.** To describe sets of objects and binary relations between them, the following countably infinite sets of symbols are available:

- *concept names* $\mathsf{N_C} = \{A, B, \ldots\}$,
- *role names* $\mathsf{N_R} = \{P, Q, \ldots\}$ and

in addition also object names $\mathsf{N_O}$ and variable names $\mathsf{N_V}$ are available. All the sets are pairwise disjoint. From these names complex concept and role descriptions can be formed using several *concept constructors* and *role constructors*. The set of all *$\mathcal{DL}$-concept descriptions* (set of all *concepts* for short) and the set of all *$\mathcal{DL}$-role descriptions* (set of all *roles* for short) are defined by mutual induction as the smallest sets satisfying the following conditions:

1. Every concept name $A \in \mathsf{N_C}$ is a concept and the symbols $\top$ (*top concept*) and $\bot$ (*bottom concept*) are concepts.
2. Every role name $P \in \mathsf{N_R}$ is a role and the symbol id (*identity role*) is a role.
3. If $t \in \mathsf{N_O} \cup \mathsf{N_V}$ is an object term, C and D concepts, R a role and n a natural number, the also the following expressions are concepts: $\{t\}$ (*nominal*), $\neg C$ (*negation*), $C \sqcap D$ (*conjunction*) and $\geq n R.C$ (*at least restriction*).
4. If R and S are roles and C and D concepts, then also the following descriptions are roles: $\mathsf{inv}(R)$ (*inverse role*), $\neg R$ (*role negation*), $R \sqcap S$ (*role conjunction*) and $C \times D$ (*concept product*).

Corresponding syntax rules for concepts C and roles R are given as follows

$$C ::= A \mid \top \mid \bot \mid \{t\} \mid \neg C \mid C \sqcap C \mid \geq n R.C$$

$$R ::= P \mid \mathsf{id} \mid \mathsf{inv}(R) \mid \neg R \mid R \sqcap R \mid C \times C.$$

▲

Concepts and roles without variable names are called *ground*. For an example of a ground concept see (1.3) in Section 1.2. The variable names occurring in nominals are placeholders for object names and are handled using syntactical substitutions.

Definition 2.19 (grounding of concepts and roles)**.** A *variable mapping* ν is a total function of the form $\nu : \mathsf{N_V} \cup \mathsf{N_O} \rightarrow \mathsf{N_O}$ such that $\nu(o) = o$ for all $o \in \mathsf{N_O}$. Given a concept C, role R and a variable mapping ν, the corresponding *ground concept and ground role*, denoted by C^ν and R^ν, respectively, are defined by mutual induction as follows:

- $A^\nu := A$, $P^\nu := P$, $\top^\nu := \top$, $\bot^\nu := \bot$ and $\mathsf{id}^\nu := \mathsf{id}$, where $A \in \mathsf{N_C}$ and $P \in \mathsf{N_R}$;
- $\{t\}^\nu := \{\nu(t)\}$, $(\neg C)^\nu := \neg C^\nu$, $(C \sqcap D)^\nu := C^\nu \sqcap D^\nu$ and $(\geq nR.C)^\nu := \geq nR^\nu.C^\nu$, where $t \in \mathsf{N_V} \cup \mathsf{N_O}$, C and D are concept, R a role and n a natural number;
- $(\mathsf{inv}(R))^\nu := \mathsf{inv}(R^\nu)$, $(\neg R)^\nu := \neg R^\nu$, $(R \sqcap S)^\nu := R^\nu \sqcap S^\nu$ and $(C \times D)^\nu := C^\nu \times D^\nu$, where R and S are roles and C and D concepts.

▲

The semantics of ground concepts and ground roles is defined in terms of first-order interpretations $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ such that concept names are interpreted as unary predicate names from $\mathsf{N_F}$ and role names as binary predicate names from $\mathsf{N_F}$.

Definition 2.20 (semantics of ground concepts and ground roles)**.** Let $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation. The function $\cdot^\mathcal{I}$ maps each concept name $A \in \mathsf{N_C}$ to a set $A^\mathcal{I} \subseteq \Delta_\mathcal{I}$, each role name P to a binary relation $P^\mathcal{I} \subseteq \Delta_\mathcal{I} \times \Delta_\mathcal{I}$ and each object name $o \in \mathsf{N_O}$ to an element $o^\mathcal{I} \in \Delta_\mathcal{I}$. The mapping $\cdot^\mathcal{I}$ is extended to complex ground concepts C and ground roles R. The image sets $C^\mathcal{I}$ and $R^\mathcal{I}$ are called *extension of C and R, respectively, under $\mathcal{I}$* and are defined inductively as given in the third column of Table 2.1, where C and D denote ground concepts, R and S ground roles and o and o' object names. ▲

Note that $\top$ and $\bot$ can be defined as abbreviations using a concept name, negation and conjunction. The list given in Table 2.1 contains concept constructors and role constructors not mentioned in Definition 2.18. In the table C, D and R, S stand for possibly complex *ground* concepts and roles, respectively, but we sometimes also use the constructors to built concepts with nominals that contain variable names. U is a fixed symbol denoting the universal role (or "top role"), it is the role-counterpart of the top concept. We will later use the additional constructors to define sub-logics of $\mathcal{DL}$.

The additional concept constructors $C \sqcup D$ (*disjunction*), $\exists R.C$ (*existential restriction*), $\forall R.C$ (*value restriction*) and $\leq nR.C$ (*at most restriction*) in the table do not increase the expressiveness of $\mathcal{DL}$. They can be equivalently expressed using only the constructors mentioned in Definition 2.18:

$$\begin{aligned} C \sqcup D &\equiv \neg(\neg C \sqcap \neg D), & \exists R.C &\equiv \geq 1R.C, \\ \forall R.C &\equiv \neg(\geq 1R.\neg C), & \leq nR.C &\equiv \neg(\geq(n+1)R.C). \end{aligned}$$

Concepts

constructor	syntax	extension under $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$	FO mapping $\mathsf{tr}^x(\cdot)$
concept name	$A \in \mathsf{N_C}$	$A^\mathcal{I} \subseteq \Delta_\mathcal{I}$	$A(x)$
top concept	$\top$	$\Delta_\mathcal{I}$	$x = x$
bottom concept	$\bot$	$\emptyset$	$x \neq x$
negation	$\neg C$	$\Delta_\mathcal{I} \setminus C^\mathcal{I}$	$\neg \mathsf{tr}^x(C)$
conjunction	$C \sqcap D$	$C^\mathcal{I} \cap D^\mathcal{I}$	$\mathsf{tr}^x(C) \land \mathsf{tr}^x(D)$
disjunction	$C \sqcup D$	$C^\mathcal{I} \cup D^\mathcal{I}$	$\mathsf{tr}^x(C) \lor \mathsf{tr}^x(D)$
existential restriction	$\exists R.C$	$\{d \mid \exists e \in \Delta_\mathcal{I}.(d,e) \in R^\mathcal{I} \land e \in C^\mathcal{I}\}$	$\exists y.(\mathsf{tr}^{x,y}(R) \land \mathsf{tr}^y(C))$
value restriction	$\forall R.C$	$\{d \mid \forall e \in \Delta_\mathcal{I}.(d,e) \in R^\mathcal{I} \to e \in C^\mathcal{I}\}$	$\forall y.(\mathsf{tr}^{x,y}(R) \to \mathsf{tr}^y(C))$
nominal	$\{o\}$	$\{o^\mathcal{I}\}$	$x = o$
at most restriction	$\leq n R.C$	$\{d \mid \lvert\{e \mid (d,e) \in R^\mathcal{I} \land e \in C^\mathcal{I}\}\rvert \leq n\}$	$\exists^{\leq n} y.(\mathsf{tr}^{x,y}(R) \land \mathsf{tr}^y(C))$
at least restriction	$\geq n R.C$	$\{d \mid \lvert\{e \mid (d,e) \in R^\mathcal{I} \land e \in C^\mathcal{I}\}\rvert \geq n\}$	$\exists^{\geq n} y.(\mathsf{tr}^{x,y}(R) \land \mathsf{tr}^y(C))$

Roles

constructor	syntax	extension under $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$	FO mapping $\mathsf{tr}^{x,y}(\cdot)$
role name	$P \in \mathsf{N_R}$	$P^\mathcal{I} \subseteq \Delta_\mathcal{I} \times \Delta_\mathcal{I}$	$P(x,y)$
inverse role	$\mathsf{inv}(R)$	$\{(e,d) \mid (d,e) \in R^\mathcal{I}\}$	$\mathsf{tr}^{y,x}(R)$
identity role	id	$\{(d,d) \mid d \in \Delta_\mathcal{I}\}$	$x = y$
universal role	U	$\Delta_\mathcal{I} \times \Delta_\mathcal{I}$	$x = x \land y = y$
role negation	$\neg R$	$(\Delta_\mathcal{I} \times \Delta_\mathcal{I}) \setminus R^\mathcal{I}$	$\neg \mathsf{tr}^{x,y}(R)$
role conjunction	$R \sqcap S$	$R^\mathcal{I} \cap S^\mathcal{I}$	$\mathsf{tr}^{x,y}(R) \land \mathsf{tr}^{x,y}(S)$
role disjunction	$R \sqcup S$	$R^\mathcal{I} \cup S^\mathcal{I}$	$\mathsf{tr}^{x,y}(R) \lor \mathsf{tr}^{x,y}(S)$
role difference	$R \setminus S$	$R^\mathcal{I} \setminus S^\mathcal{I}$	$\mathsf{tr}^{x,y}(R) \land \neg \mathsf{tr}^{x,y}(S)$
nominal role	$\{(o,o')\}$	$\{(o^\mathcal{I}, o'^\mathcal{I})\}$	$x = o \land y = o'$
concept product	$C \times D$	$C^\mathcal{I} \times D^\mathcal{I}$	$\mathsf{tr}^x(C) \land \mathsf{tr}^y(D)$

Table 2.1: List of concept and role constructors

The same holds for the additional role constructors U (*universal role*), $R \sqcup S$ (*role disjunction*), $R \setminus S$ (*role difference*) and $\{(o, o')\}$ (*nominal role*):

$$\begin{aligned} R \sqcup S &\equiv \neg(\neg R \sqcap \neg S), & R \setminus S &\equiv R \sqcap \neg S, \\ U &\equiv \top \times \top, & \{(o, o')\} &\equiv \{o\} \times \{o'\}. \end{aligned}$$

In a $\mathcal{DL}$-knowledge base ($\mathcal{DL}$-KB for short) axioms stating facts about certain objects, so called *ABox assertions*, and axioms representing general domain knowledge, called *concept inclusions* (CIs), are distinguished.

Definition 2.21 (syntax of axioms and KBs)**.** Let C and D be possibly non-ground concept descriptions, R a possibly non-ground role description and $t, t' \in \mathsf{N_V} \cup \mathsf{N_O}$ object terms. *ABox assertions* are of the form

- $t \in C$ (*concept assertion*) or
- $(t, t') \in R$ (*positive role assertion*) or $(t, t') \in \neg R$ (*negative role assertion*) or
- $t \approx t'$ or $t \not\approx t'$ (equality and inequality assertion, respectively).

A *concept inclusion* (CI) is of the form
$$C \sqsubseteq D.$$
An ABox assertion or CI is called *ground*, if no variable names occur in it. An *ABox* $\mathcal{A}$ is a finite set of ground ABox assertions and a *TBox* $\mathcal{T}$ is a finite set of ground CIs. A *knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$.

We also consider Boolean combinations of ABox assertions and concept inclusions. A *Boolean KB* ψ is built according to the following syntax rule

$$\psi ::= t \in C \mid (t, t') \in R \mid (t, t') \in \neg R \mid t \approx t' \mid t \not\approx t' \mid C \sqsubseteq D \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi.$$

A *Boolean ABox* is a Boolean KB with only ABox assertions as propositions. The *set of variable names occurring in a Boolean KB* ψ is denoted by $\mathsf{FVar}(\psi)$. A *ground Boolean KB (ABox)* is a Boolean KB (ABox) without variable names.

Concept and role assertions of the form $t \in A$, $t \in \neg A$, $(t, t') \in P$ or $(t, t') \in \neg P$, where A is a concept name, P a role name and t, t' object terms, are called *ABox literals* (*literals* for short). ▲

Definition 2.22. Let $\nu : \mathsf{N_V} \cup \mathsf{N_O} \to \mathsf{N_O}$ be a variable mapping and ψ a Boolean KB. The *ground Boolean KB* ψ^ν is defined by induction on the structure of ψ as follows:

- $(t \in C)^\nu := \nu(t) \in C^\nu$, where $t \in C$ is a concept assertion,
- $((t, t') \in R)^\nu := (\nu(t), \nu(t')) \in R^\nu$, $((t, t') \in \neg R)^\nu := (\nu(t), \nu(t')) \in \neg R^\nu$, where $(t, t') \in R$ and $(t, t') \in \neg R$ are role assertions and
- $(t \approx t')^\nu := \nu(t) \approx \nu(t')$, $(t \not\approx t')^\nu := \nu(t) \not\approx \nu(t')$, where $t \approx t'$ and $t \not\approx t'$ are equality and inequality assertions, respectively, and
- $(C \sqsubseteq D)^\nu := C^\nu \sqsubseteq D^\nu$, where $C \sqsubseteq D$ is a concept inclusion and

- $(\neg\psi_1)^\nu := \neg\psi_1^\nu$, $(\psi_0 \wedge \psi_1)^\nu := \psi_0^\nu \wedge \psi_1^\nu$ and $(\psi_0 \vee \psi_1)^\nu := \psi_0^\nu \vee \psi_1^\nu$, where ψ_0 and ψ_1 are Boolean KBs.

▲

If not stated otherwise, we implicitly assume in the following that all concepts, roles, ABox assertions, CIs, ABoxes, TBoxes and Boolean KBs are ground.

Definition 2.23 (semantics of axioms and KBs)**.** Let $o, o' \in \mathsf{N_O}$, C a concept, R a role, ψ_1 and ψ_2 Boolean KBs and $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ an interpretation. *Satisfaction of a Boolean KB ψ in $\mathcal{I}$*, denoted by $\mathcal{I} \models \psi$, is defined inductively as follows:

$$\begin{array}{ll}
\mathcal{I} \models o \sqsubseteq C & \text{iff } o^\mathcal{I} \in C^\mathcal{I} \\
\mathcal{I} \models (o, o') \sqsubseteq R & \text{iff } (o^\mathcal{I}, o'^\mathcal{I}) \in R^\mathcal{I} \\
\mathcal{I} \models (o, o') \sqsubseteq \neg R & \text{iff } (o^\mathcal{I}, o'^\mathcal{I}) \notin R^\mathcal{I} \\
\mathcal{I} \models o \approx o' & \text{iff } o^\mathcal{I} = o'^\mathcal{I} \\
\mathcal{I} \models o \not\approx o' & \text{iff } o^\mathcal{I} \neq o'^\mathcal{I} \\
\mathcal{I} \models C \sqsubseteq D & \text{iff } C^\mathcal{I} \subseteq D^\mathcal{I} \\
\mathcal{I} \models \neg\psi_1 & \text{iff } \mathcal{I} \not\models \psi_1 \\
\mathcal{I} \models \psi_1 \wedge \psi_2 & \text{iff } \mathcal{I} \models \psi_1 \text{ and } \mathcal{I} \models \psi_2 \\
\mathcal{I} \models \psi_1 \vee \psi_2 & \text{iff } \mathcal{I} \models \psi_1 \text{ or } \mathcal{I} \models \psi_2.
\end{array}$$

We say that the *interpretation $\mathcal{I}$ is a model of the Boolean KB ψ* iff $\mathcal{I} \models \psi$.

An *interpretation $\mathcal{I}$ is a model of a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$*, denoted by $\mathcal{I} \models \mathcal{K}$, iff all axioms in $\mathcal{A}$ and $\mathcal{T}$ are satisfied in $\mathcal{I}$.

We say that $\mathcal{K}$ *is consistent* iff $\mathcal{K}$ has a model. A Boolean KB ψ is entailed by $\mathcal{K}$, denoted by $\mathcal{K} \models \psi$, iff all models of $\mathcal{K}$ are also models of ψ. ▲

Note that ABox assertions can be equivalently formulated as concept inclusions and vice versa. However, this is not the case for all sub-logics of $\mathcal{DL}$ we consider later.

Other kinds of DL axioms are defined as abbreviations as follows:

- *role inclusion*: $R \sqsubseteq S := \exists(R \setminus S).\top \sqsubseteq \bot$,
- *concept and role equivalence*: $C \equiv D := C \sqsubseteq D \wedge D \sqsubseteq C, \quad R \equiv S := R \sqsubseteq S \wedge S \sqsubseteq R.$

Furthermore, the following abbreviations are used:

- $\textsc{True} := o \sqsubseteq \top$, and
- $\textsc{False} := o \sqsubseteq \bot$,

where o is an arbitrary but fixed object name.

A concept equivalence with a concept name on the left-hand side can be viewed as an abbreviation of a complex concept. A set of such non-circular abbreviations is called *acyclic TBox* and is formally defined as follows.

Definition 2.24. A concept equivalence of the form $A \equiv C$, where A is a concept name, is called *concept definition*. An *acyclic TBox* $\mathcal{T}$ is a finite set of concept definitions with unique left-hand sides that satisfies the following restriction: there is *no* sequence of the form

$$A_1 \equiv C_1 \in \mathcal{T}, \ldots A_k \equiv C_k \in \mathcal{T} \text{ for some } k \geq 1$$

such that A_{i+1} is mentioned in C_i for all $i \in \{1, \ldots, k-1\}$, and $A_1 = A_k$ and $C_1 = C_k$.

A concept name A that occurs on the left-hand side of some definition in $\mathcal{T}$ is called *defined name in* $\mathcal{T}$. ▲

Reasoning in presence of an acyclic TBox can be reduced to reasoning w.r.t. the empty TBox by exhaustively replacing all defined names by their corresponding right-hand side.

Let X be a concept, role, ABox, TBox, KB or Boolean KB. The *size* of X, denoted by $|X|$, is defined in the usual way as the number of symbols needed to write X. Numbers in at most and at least restrictions are assumed to be encoded in binary.

Different DLs and Relation to First-Order Logic

In order to find an appropriate trade-off between expressiveness and complexity of reasoning, in the literature on DLs often only sub-logics of full $\mathcal{DL}$ are considered. Specific restricted DLs differ in the set of concept and role constructors that are allowed. A basic DL that is a sub-logic of $\mathcal{DL}$ is called $\mathcal{ALC}$. An $\mathcal{ALC}$-concept C is built according to the following syntax rule:

$$C ::= A \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \exists P.C \mid \forall P.C,$$

where $A \in \mathsf{N_C}$ and $P \in \mathsf{N_R}$. $\mathcal{ALC}$-roles are role names. Complex roles are not allowed. Other DLs between $\mathcal{ALC}$ and $\mathcal{DL}$ are obtained by allowing further constructors for concepts and roles.

The name of a DL that extends $\mathcal{ALC}$ with some constructors is obtained by appending corresponding letters or symbols given in Table 2.2 to the string $\mathcal{ALC}$. For instance, the DL $\mathcal{ALCQIO}$ extends $\mathcal{ALC}$ with at least and at most restrictions, inverse roles and nominals. Thus, in $\mathcal{ALCQIO}$ the full set of concept constructors is available but roles are restricted to be either role names or inverses of role names. Symbols for additional role constructors are added in parentheses as a superscript of the name. For example the DL $\mathcal{ALCQIO}^{(U)}$ extends $\mathcal{ALCQIO}$ with the universal role and the DL named by $\mathcal{ALCQIO}^{(\mathsf{id},\neg,\sqcap,\times)}$ corresponds to full $\mathcal{DL}$.

Let $\mathcal{L}$ be the name of a particular DL. $\mathcal{L}$-ABox assertions, $\mathcal{L}$-ABoxes, $\mathcal{L}$-CIs, $\mathcal{L}$-TBoxes or Boolean $\mathcal{L}$-KBs are constructed by using only $\mathcal{L}$-concepts and $\mathcal{L}$-roles. Note that if R is an $\mathcal{L}$-role, then $(t, t') \in \neg R$ is also an $\mathcal{L}$-ABox assertion even if $\mathcal{L}$ does not support role negation. If no particular name of a DL is mentioned, then we assume that $\mathcal{DL}$ is used.

DL concepts, roles and axioms can be translated to first-order logic using standard methods [Bor96].

Definition 2.25 (translation to FO syntax). For the translation concept names are viewed as elements of $\mathsf{N_F}$ with arity one and role names as elements of $\mathsf{N_F}$ with arity two. Let $x, y \in \mathsf{N_V}$ be variable names. The functions $\mathsf{tr}^x(\cdot)$ and $\mathsf{tr}^y(\cdot)$ map ground concepts to FO formulas with one free variable x and y, respectively, and the functions $\mathsf{tr}^{x,y}(\cdot)$ and $\mathsf{tr}^{y,x}(\cdot)$ map ground roles to FO formulas with free variables x and y. $\mathsf{tr}^x(\cdot)$ and $\mathsf{tr}^{x,y}(\cdot)$ are defined as given in

Concept		Role	
DL name	constructor	DL name	constructor
	top concept	$\mathcal{I}$	inverse role
	bottom concept	$\cdot^{(,\mathsf{id})}$	identity role
$\mathcal{ALC}$	negation	$\cdot^{(,U)}$	universal role
	conjunction	$\cdot^{(,\neg)}$	role negation
	disjunction	$\cdot^{(,\sqcap)}$	role conjunction
	existential restriction	$\cdot^{(,\sqcup)}$	role disjunction
	value restriction	$\cdot^{(,\mathsf{diff})}$	role difference
$\mathcal{Q}$	at most restriction	$\cdot^{(,\mathsf{o})}$	nominal role
	at least restriction	$\cdot^{(,\times)}$	concept product
$\mathcal{O}$	nominal		

Table 2.2: Names of different DLs

fourth column of Table 2.1 and $\mathsf{tr}^y(\cdot)$ and $\mathsf{tr}^{y,x}(\cdot)$ are defined as $\mathsf{tr}^x(\cdot)$ and $\mathsf{tr}^{x,y}(\cdot)$ but with x and y swapped. For axioms a function $\mathsf{tr}(\cdot)$ is defined as follows:

$$\begin{aligned}
\mathsf{tr}(o \sqsubseteq C) &:= \exists x.(x \approx o \land \mathsf{tr}^x(C)),\\
\mathsf{tr}((o,o') \sqsubseteq R) &:= \exists x,y.(x \approx o \land y \approx o' \land \mathsf{tr}^{x,y}(R)),\\
\mathsf{tr}((o,o') \sqsubseteq \neg R) &:= \neg \mathsf{tr}((o,o') \sqsubseteq R),\\
\mathsf{tr}(C \sqsubseteq D) &:= \forall x.(\mathsf{tr}^x(C) \rightarrow \mathsf{tr}^x(D)),\\
\mathsf{tr}(o \approx o') &:= o \approx o',\\
\mathsf{tr}(o \not\approx o') &:= o \not\approx o'.
\end{aligned}$$

▲

2.2.2 Integrating DL Knowledge Bases and Actions

We introduce the notion of a DL-action theory that can be viewed as a specific class of DL-admissible representations.

Note that in context with DLs we consider FO-DSs of the form $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$, where $\mathcal{F}$ is the finite set of all relevant concept names and role names.

According to Definition 2.15 an $\mathcal{L}$-admissible representation for a finite set of ground actions $\boldsymbol{A}$ and a DL $\mathcal{L}$ is of the form $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$, where

- the context $\mathcal{C}$ consists of Boolean $\mathcal{L}$-KBs and
- the computable functions $\mathsf{E}^+[\cdot]$ and $\mathsf{E}^-[\cdot]$ provide possibly complex $\mathcal{L}$-concepts and

$\mathcal{L}$-roles defining the add-sets and delete-sets for the relevant concept names and role names, respectively.

Our notion of a DL-admissible representation for a set of ground actions seems to be general enough to capture the meaning of *deterministic* ground actions definable in the action languages considered in [Baa+05a; BLL10; GS10; Ahm+14]. However, we do not provide a formal proof for this claim.

We introduce *DL-action theories* as a formalism where the domain modeler explicitly provides a (complete) list of preconditions and so-called effect descriptions for each action. Even though we use the word "theory" a DL-action theory is not a set of axioms formulated in a logic that allows us to talk about actions. Instead, we introduce a special syntax with a semantics directly given in terms of an FO-DS. A DL-action theory is a special case of a DL-admissible representation. For such theories we will then briefly discuss several ways to integrate a TBox as a set of global state constraints.

DL-Action Theories

The notion of a general effect description provides a syntactical schema for describing the changes caused by an action execution.

Definition 2.26. *Effect descriptions* (*effects* for short) are expressions of the form

- $\psi \rhd \langle A, C\rangle^+$ or $\psi \rhd \langle P, R\rangle^+$ (called *positive effect on A or P, respectively*) or
- $\psi \rhd \langle A, C\rangle^-$ or $\psi \rhd \langle P, R\rangle^-$ (called *negative effect on A or P, respectively*),

where ψ stands for a possibly non-ground Boolean KB, A for a concept name, C for a possibly non-ground concept, P for a role name and R for a possibly non-ground role. ψ is called *effect condition* and C and R *effect descriptors*.

The expression $\psi \rhd \langle F, X\rangle^\pm$ stands for a *positive or negative effect description* on a role name or concept name F.

An *unconditional effect* is an effect, where the effect condition ψ is a tautology. In this case we omit the effect condition and write just

$$\langle F, X\rangle^\pm \text{ instead of } \textsc{True} \rhd \langle F, X\rangle^\pm.$$

The instantiation of an effect description given a variable mapping is defined in the obvious way. Let ν be a variable mapping and $\mathsf{e} = \psi \rhd \langle F, X\rangle^\pm$ an effect. With e^ν we denote the ground instantiated effect given by $\psi^\nu \rhd \langle F, X^\nu\rangle^\pm$.

Effects of the form

$$\psi \rhd \langle A, \{t\}\rangle^\pm \text{ or } \psi \rhd \big\langle P, \{(t, t')\}\big\rangle^\pm,$$

where t and t' are object terms are called *local effects*.

Let $\mathcal{L}$ be a DL. An effect $\psi \rhd \langle F, X\rangle^\pm$ is called an *$\mathcal{L}$-effect* iff ψ is a Boolean $\mathcal{L}$-KB and X is an $\mathcal{L}$-concept or $\mathcal{L}$-role. A local effect is called *local $\mathcal{L}$-effect* iff the effect condition is a Boolean $\mathcal{L}$-KB. ▲

For instance, a positive and ground effect like $\psi \rhd \langle A, C\rangle^+$ changes the interpretation of the concept name A by adding all instances of the concept C to A. These changes are triggered only if the effect condition ψ is true. In case of a local effect the effect descriptor C is just a nominal of the form $\{o\}$. It means that only the single object the name o refers to is added to A. To define local effects nominals and nominal roles are needed, because an $\mathcal{L}$-effect is required to have an effect descriptor formulated in $\mathcal{L}$. However, when we call an effect description a "local $\mathcal{L}$-effect" for some specific DL $\mathcal{L}$ we only require that the effect condition has to be a Boolean $\mathcal{L}$-KB but $\mathcal{L}$ need not necessarily offer nominals or nominal roles. Let $\mathcal{LO}^{(\mathsf{o})}$ be the extension of $\mathcal{L}$ with nominal concepts and nominal roles. According to our definition every local $\mathcal{L}$-effect is an $\mathcal{LO}^{(\mathsf{o})}$-effect.

Example 2.27 (2.7 continued)**.** Consider the action `disable`(p_0) from Example 2.7. The effects of this action can be described using $\mathcal{ALCO}$-effects. *Dev*, *PowerS* and *On* are viewed as concept names and *ConTo* as a role name. We have the following delete effect on *PowerS*:

$$\langle \mathit{PowerS}, \{\mathsf{p}_0\}\rangle^-.$$

p_0 is deleted from *PowerS*. It is an unconditional local effect. Only the object p_0 is affected. Furthermore, the action for disabling p_0 turns off all devices without any other power supply in reserve. This is represented using the following (non-local) effect:

$$\langle \mathit{On}, \mathit{Dev} \sqcap \exists \mathit{ConTo}.\{\mathsf{p}_0\} \sqcap \forall \mathit{ConTo}.(\neg \mathit{PowerS} \sqcup \{\mathsf{p}_0\})\rangle^-.$$

All instances of the concept

$$\mathit{Dev} \sqcap \exists \mathit{ConTo}.\{\mathsf{p}_0\} \sqcap \forall \mathit{ConTo}.(\neg \mathit{PowerS} \sqcup \{\mathsf{p}_0\})$$

describing the set of devices with p_0 as the only connected power supply are deleted from *On*. ▲

As an auxiliary notion we define the *update of an interpretation given a set of unconditional effects*.

Definition 2.28. Let $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ be an interpretation and E a finite set of unconditional ground effect descriptions. The *update of $\mathcal{I}$ w.r.t.* E is an interpretation, denoted by $\mathcal{I}^\mathsf{E}$, that is defined as follows:

$$\begin{aligned}
\Delta_{\mathcal{I}^\mathsf{E}} &:= \Delta_\mathcal{I};\\
F^{\mathcal{I}^\mathsf{E}} &:= \big(F^\mathcal{I} \setminus \big(\bigcup_{\langle F,X\rangle^- \in \mathsf{E}} X^\mathcal{I}\big)\big) \cup \bigcup_{\langle F,X\rangle^+ \in \mathsf{E}} X^\mathcal{I} \text{ for all } F \in \mathsf{N_C} \cup \mathsf{N_R};\\
o^{\mathcal{I}^\mathsf{E}} &:= o^\mathcal{I} \text{ for all } o \in \mathsf{N_O}.
\end{aligned}$$

▲

Next, we introduce the syntax of DL-action theories.

Definition 2.29. A *DL-action theory* is a tuple $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$, where

- $\mathcal{K}$ is a DL KB,
- Act is a finite set of action terms with pairwise different action names,

- pre associates each action term $\alpha(\bar{t}) \in \mathsf{Act}$ with a set of (possibly non-ground) Boolean DL-KBs, denoted by $\mathsf{pre}(\alpha(\bar{t}))$, and
- eff associates each action term $\alpha(\bar{t}) \in \mathsf{Act}$ with a set of effects, denoted by $\mathsf{eff}(\alpha(\bar{t}))$.

We assume that for all $\alpha(\bar{t}) \in \mathsf{Act}$ the sets $\mathsf{pre}(\alpha(\bar{t}))$ and $\mathsf{eff}(\alpha(\bar{t}))$ are explicitly given and the expressions therein mention only object terms that are arguments of $\alpha(\bar{t})$.

Since the action terms in Act have pairwise different action names, we can uniquely extend eff and pre to the set of all ground actions ground(Act) as follows: For any $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ such that $\boldsymbol{\alpha}$ is the ground instantiation of $\alpha(\bar{t}) \in \mathsf{Act}$ with ν we define

$$\mathsf{pre}(\boldsymbol{\alpha}) := \{\psi_1{}^{\nu}, \dots, \psi_n{}^{\nu}\} \text{ and } \mathsf{eff}(\boldsymbol{\alpha}) := \{\mathsf{e}_1{}^{\nu}, \dots, \mathsf{e}_m{}^{\nu}\}$$

where $\mathsf{pre}(\alpha(\bar{t})) = \{\psi_1, \dots, \psi_n\}$ and $\mathsf{eff}(\alpha(\bar{t})) = \{\mathsf{e}_1, \dots, \mathsf{e}_m\}$ for some $n, m \geq 0$.

The DL-action theory Σ is called *local effect action theory* iff for all $\alpha(\bar{t}) \in \mathsf{Act}$ the set $\mathsf{eff}(\alpha(\bar{t}))$ contains only local effects. Let $\mathcal{L} \subseteq \mathcal{DL}$ be a DL.

- Σ is an *$\mathcal{L}$-action theory* iff $\mathcal{K}$ is an $\mathcal{L}$-KB, and for all $\alpha(\bar{t}) \in \mathsf{Act}$ the set $\mathsf{pre}(\alpha(\bar{t}))$ consists of Boolean $\mathcal{L}$-KBs and the effects $\mathsf{eff}(\alpha)$ are $\mathcal{L}$-effects.
- And Σ is called a *local effect $\mathcal{L}$-action theory* iff $\mathcal{K}$ is an $\mathcal{L}$-KB, and for all $\alpha(\bar{t}) \in \mathsf{Act}$ the set $\mathsf{pre}(\alpha(\bar{t}))$ consists of Boolean $\mathcal{L}$-KBs and the effects $\mathsf{eff}(\alpha(\bar{t}))$ are local $\mathcal{L}$-effects.

▲

Next, we define the semantics of a DL-action theory in terms of an FO-DS.

Definition 2.30. Let $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$ be a DL-action theory with

$$\mathsf{Act} = \{\alpha_1(\bar{t}_1), \dots, \alpha_n(\bar{t}_n)\}$$

for some $n > 0$. *The FO-DS induced by* Σ is an FO-DS

$$\mathfrak{D}(\Sigma) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$$

over the state space

$$\mathbb{I} := \{\mathcal{J} \mid \text{ there exists } \mathcal{I} \in \mathcal{M}(\mathcal{K}) \text{ with } \Delta_{\mathcal{J}} = \Delta_{\mathcal{I}} \text{ and } o^{\mathcal{J}} = o^{\mathcal{I}} \text{ for all } o \in \mathsf{N_O}\};$$

and with $\mathcal{F}$, $\mathcal{E}$ and Pre defined as follows:

1. $\mathcal{F}$ consists of all those concept names and role names that are mentioned in $\mathcal{K}$, in the effect descriptions in the sets $\mathsf{eff}(\alpha_1(\bar{t}_1)), \dots, \mathsf{eff}(\alpha_n(\bar{t}_n))$ and in the preconditions in $\mathsf{pre}(\alpha_1(\bar{t}_1)), \dots, \mathsf{pre}(\alpha_n(\bar{t}_n))$.
2. For all $(\mathcal{I}, \boldsymbol{\alpha}, F) \in \mathbb{I} \times \mathsf{ground}(\mathsf{Act}) \times \mathcal{F}$ we define

$$\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F) := \bigcup_{\substack{\psi \rhd \langle F, X\rangle^{+} \in \mathsf{eff}(\boldsymbol{\alpha}),\\ \mathcal{I} \models \psi}} X^{\mathcal{I}} \text{ and } \mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F) := \bigcup_{\substack{\psi \rhd \langle F, X\rangle^{-} \in \mathsf{eff}(\boldsymbol{\alpha}),\\ \mathcal{I} \models \psi}} X^{\mathcal{I}}.$$

action term α	$\mathsf{pre}(\alpha)$	$\mathsf{eff}(\alpha)$
$\texttt{turn-on}(x)$	$\{(x \sqsubseteq Dev)\}$	$\{\langle On, \{x\}\rangle^+\}$
$\texttt{connect}(x,y)$	$\{(x \sqsubseteq Dev), (y \sqsubseteq PowerS)\}$	$\{\langle ConTo, \{(x,y)\}\rangle^+\}$
$\texttt{disconnect}(x,y)$	$\{(x \sqsubseteq Dev), (y \sqsubseteq PowerS)\}$	$\{\langle ConTo, \{(x,y)\}\rangle^-\}$

Table 2.3: Example action descriptions

3. For all $\mathcal{I} \in \mathbb{I}$ and all $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ we define

$$(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre} \text{ iff } (\mathcal{I} \models \varphi \text{ for all } \varphi \in \mathsf{pre}(\boldsymbol{\alpha})).$$

▲

The semantics ensures that all effects in the set $\mathsf{eff}(\boldsymbol{\alpha})$ of a ground action $\boldsymbol{\alpha}$ take place simultaneously. The set $\mathsf{eff}(\boldsymbol{\alpha})$ completely represents *all* changes caused by an execution of $\boldsymbol{\alpha}$, i.e. only the effects in $\mathsf{eff}(\boldsymbol{\alpha})$ are executed in case the effect condition is satisfied and nothing else is changed. Consequently, the transition semantics of $\mathfrak{D}(\Sigma)$ can be also characterized in terms of interpretation updates.

Lemma 2.31. *Let $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$ be a DL-action theory and $\mathcal{I}$ an interpretation from the state space of the induced FO-DS $\mathfrak{D}(\Sigma)$ and $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ a ground action. It holds that*

$$\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma)} \mathcal{I}^{\mathsf{E}} \textit{ with } \mathsf{E} = \{\langle F,X\rangle^{\pm} \mid \psi \rhd \langle F,X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}) \textit{ and } \mathcal{I} \models \psi\}.$$

Proof. We omit the proof. The lemma directly follows from the definitions. □

Example 2.32. We model a simple domain about electrical devices (represented by the concept name *Dev*), that can be on, i.e. can be instances of the concept name *On*, and can be connected to power supplies (*PowerS*) via the role name *ConTo*. We use the individual names dev for a particular device and main-socket for a particular power supply.

For now, assume that the TBox $\mathcal{T}$ of the initial knowledge base $\mathcal{K}$ is empty. The ABox $\mathcal{A}$ of $\mathcal{K}$ describing the initial situation consists of the facts:

$$(\mathsf{dev} \sqsubseteq Dev \sqcap \neg On), (\mathsf{main\text{-}socket} \sqsubseteq PowerS). \tag{2.2}$$

Consider the following action terms that are part of Act:

$$\texttt{turn-on}(x), \texttt{connect}(x,y), \texttt{disconnect}(x,y) \tag{2.3}$$

The preconditions and effects for these non-ground actions are given in Table 2.3. The actions only have local unconditional effects. For example the ground instance turn-on(dev) is possible in any model of the assertions in (2.2) because dev is a device. Executing turn-on(dev) in an interpretation adds the object dev to the extension of *On* and changes nothing else. ▲

A DL-action theory $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$ yields a DL-admissible representation of any finite set of ground actions $\boldsymbol{A} \subseteq \mathsf{ground}(\mathsf{Act})$. It is easy to see that all ground actions in an FO-DS induced by a DL-action theory are DL-definable w.r.t. a DL-context.

Lemma 2.33. *Let* $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$ *be an* $\mathcal{L}$*-action theory for some DL* $\mathcal{L}$ *with* $\mathcal{ALC} \subseteq \mathcal{L} \subseteq \mathcal{DL}$ *and* $\boldsymbol{A} \subseteq \mathsf{ground}(\mathsf{Act})$ *a finite set of ground actions. It holds that the actions in* $\boldsymbol{A}$ *are* $\mathcal{L}^{(\mathsf{diff},\sqcup)}$*-definable in the FO-DS* $\mathfrak{D}(\Sigma)$ *induced by* Σ *w.r.t. the context*

$$\mathcal{C}_{\boldsymbol{A}} = \bigcup_{\boldsymbol{\alpha}\in\boldsymbol{A}} \big(\{\psi, \neg\psi \mid \psi \in \mathsf{pre}(\boldsymbol{\alpha})\} \cup \{\psi, \neg\psi \mid \psi \rhd \langle F, X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha})\}\big).$$

Proof. An $\mathcal{L}^{(\mathsf{diff},\sqcup)}$-admissible representation for $\boldsymbol{A}$ of the form $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}_{\boldsymbol{A}}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}_{\boldsymbol{A}}})$, where $\mathcal{F}$ consists of all the names mentioned in Σ, can be defined by

$$\begin{aligned}\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F] &:= \bigsqcup\{X \mid \psi \rhd \langle F, X\rangle^+ \in \mathsf{eff}(\boldsymbol{\alpha}), \psi \in \mathfrak{s}\} \text{ and}\\ \mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F] &:= \bigsqcup\{X \mid \psi \rhd \langle F, X\rangle^- \in \mathsf{eff}(\boldsymbol{\alpha}), \psi \in \mathfrak{s}\}\end{aligned}$$

for all $(\mathfrak{s}, \boldsymbol{\alpha}, F) \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{A}}} \times \boldsymbol{A} \times \mathcal{F}$ and

$$(\mathfrak{s}, \boldsymbol{\alpha}) \in \mathsf{Pre}_{\mathcal{C}_{\boldsymbol{A}}} \text{ iff } \mathsf{pre}(\boldsymbol{\alpha}) \subseteq \mathfrak{s}, \text{ for all } (\mathfrak{s}, \boldsymbol{\alpha}) \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{A}}} \times \boldsymbol{A}.$$

For a finite set of concepts/roles M, $\bigsqcup M$ denotes the disjunction of all concepts/roles contained in M. We assume $\bigsqcup\emptyset := \bot$ in case of concepts and $\bigsqcup\emptyset := P \setminus P$ for some $P \in \mathsf{N_R}$ in case of roles. It follows that $\mathfrak{D}(\Sigma)|_{\boldsymbol{A}} = \mathfrak{D}(\Sigma_{\boldsymbol{A}})$ □

For a DL-admissible representation an equivalent DL-action theory is computable.

Lemma 2.34. *Let* $\mathcal{L}$ *be a DL,* $\boldsymbol{A}$ *a finite set of ground action terms and*

$$\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$$

an $\mathcal{L}$*-admissible representation of* $\boldsymbol{A}$*. It holds that an* $\mathcal{L}$*-action theory* $\Sigma = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff})$ *is computable such that* $\mathfrak{D}(\Sigma_{\boldsymbol{A}}) = \mathfrak{D}(\Sigma)$*.*

Proof. $\mathcal{K}$ and $\boldsymbol{A}$ are already given and it remains to define pre and eff. A Boolean $\mathcal{L}$-KB describing the precondition of an action $\boldsymbol{\alpha} \in \boldsymbol{A}$ is given as follows:

$$\psi_{\boldsymbol{\alpha}} := \bigvee_{\substack{\mathfrak{s}\in\mathfrak{S}_{\mathcal{C}},\\ (\mathfrak{s},\boldsymbol{\alpha})\in\mathsf{Pre}_{\mathcal{C}}}} \Big(\bigwedge_{\varphi\in\mathfrak{s}} \varphi\Big).$$

Since $\mathsf{Pre}_{\mathcal{C}}$ is decidable, $\psi_{\boldsymbol{\alpha}}$ is computable. We define $\mathsf{pre}(\boldsymbol{\alpha}) := \{\psi_{\boldsymbol{\alpha}}\}$ for all $\boldsymbol{\alpha} \in \boldsymbol{A}$. The corresponding effect descriptions for each ground action $\boldsymbol{\alpha} \in \boldsymbol{A}$ are obtained as follows:

$$\begin{aligned}\mathsf{eff}(\boldsymbol{\alpha}) := &\left\{\Big(\bigwedge_{\varphi\in\mathfrak{s}} \varphi\Big) \rhd \big\langle F, \mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]\big\rangle^+ \;\middle|\; \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}, \boldsymbol{\alpha} \in \boldsymbol{A}, F \in \mathcal{F}\right\} \cup\\ &\left\{\Big(\bigwedge_{\varphi\in\mathfrak{s}} \varphi\Big) \rhd \big\langle F, \mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]\big\rangle^- \;\middle|\; \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}, \boldsymbol{\alpha} \in \boldsymbol{A}, F \in \mathcal{F}\right\}.\end{aligned}$$

This completes the definition of the general $\mathcal{L}$-action theory Σ. It is straightforward to show that $\mathfrak{D}(\Sigma_A) = \mathfrak{D}(\Sigma)$. $\square$

Thus, we view DL-action theories as a *canonical DL-admissible representations* of dynamical systems. In the next subsection, the role of the TBox as a set of global state constraints is discussed. We review the example from Section 1.2 more formally.

Qualification and Ramification Problem

With the semantics given in Definition 2.30 the TBox neither affects the executability of an action nor its effects. It might happen that a ground action $\boldsymbol{\alpha}$ is executed in a model $\mathcal{I}$ of a global TBox $\mathcal{T}$ but the updated interpretation $\mathcal{I}'$ with $\mathcal{I} \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}} \mathcal{I}'$ violates a CI in $\mathcal{T}$. According to Lin and Reiter's view on state constraints [LR94] there are two possible ways to interpret this violation:

1. The violated CI is a qualification constraint. Consequently, a not explicitly stated precondition of $\boldsymbol{\alpha}$ was violated in $\mathcal{I}$ and it is not possible to execute $\boldsymbol{\alpha}$ in $\mathcal{I}$.

2. The violated CI is a ramification constraint and triggers indirect effects. The set $\mathsf{eff}(\boldsymbol{\alpha})$ is incomplete and additional changes are needed in order to keep the CI satisfied.

We now adapt the semantics of DL-action theories and incorporate the TBox as a set of qualification constraints. Let $\Sigma = (\mathcal{K} = (\mathcal{T}, \mathcal{A}), \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$ be a DL-action theory and $\mathfrak{D}(\Sigma) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ the induced FO-DS. To obtain a semantics that respects the TBox we replace Pre with a relation denoted by $\mathsf{Pre}_{\mathcal{T}}$ that is defined as follows

$$(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}_{\mathcal{T}} \text{ iff } (\mathcal{I}' \models \mathcal{T} \text{ with } \mathcal{I} \Rightarrow_{\mathfrak{D}(\Sigma)}^{\boldsymbol{\alpha}} \mathcal{I}' \text{ and } \mathcal{I} \models \varphi \text{ for all } \varphi \in \mathsf{pre}(\boldsymbol{\alpha})). \tag{2.4}$$

for all $(\mathcal{I}, \boldsymbol{\alpha}) \in \mathbb{I} \times \mathsf{ground}(\mathsf{Act})$.

We consider an example similar to the one in Section 1.2.

Example 2.35. We continue Example 2.32 about devices and their power supplies. Now the initial KB consists of a single concept inclusion:

$$\begin{aligned} \mathcal{K} = (\mathcal{T} &= \{\mathit{Dev} \sqcap \mathit{On} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}\}, \\ \mathcal{A} &= \{(\mathsf{dev} \sqsubseteq \mathit{Dev}), (\mathsf{dev} \sqsubseteq \neg \mathit{On}), (\mathsf{main\text{-}socket} \sqsubseteq \mathit{PowerS}\})). \end{aligned} \tag{2.5}$$

We consider the following sets of all relevant action terms and concept/role names.

$$\begin{aligned} \mathsf{Act} &= \{\texttt{turn-on}(x), \texttt{connect}(x,y), \texttt{disconnect}(x,y)\}; \\ \mathcal{F} &= \{\mathit{PowerS}, \mathit{Dev}, \mathit{On}, \mathit{ConTo}\}. \end{aligned}$$

The preconditions and effects of actions are defined as in Table 2.3. Consider an interpretation

$\mathcal{I}$ that is given as follows:

$$\begin{aligned}
\Delta_{\mathcal{I}} &:= \{\mathsf{dev}, d_1, d_2, \ldots\} \cup \{\mathsf{main\text{-}socket}, s_1, s_2, \ldots\},\\
\mathsf{dev}^{\mathcal{I}} &:= \mathsf{dev},\\
\mathsf{main\text{-}socket}^{\mathcal{I}} &:= \mathsf{main\text{-}socket},\\
\mathit{Dev}^{\mathcal{I}} &:= \{\mathsf{dev}, d_1, d_2, \ldots\},\\
\mathit{PowerS}^{\mathcal{I}} &:= \{\mathsf{main\text{-}socket}, s_1, s_2, \ldots\},\\
\mathit{On}^{\mathcal{I}} &:= \emptyset,\\
\mathit{ConTo}^{\mathcal{I}} &:= \emptyset.
\end{aligned}$$

Obviously, $\mathcal{I}$ is a model of $\mathcal{T}$ and $\mathcal{A}$. Now, consider the ground instance $\mathtt{turn\text{-}on}(\mathsf{dev})$ of $\mathtt{turn\text{-}on}(x)$ with

$$\mathsf{pre}(\mathtt{turn\text{-}on}(\mathsf{dev})) = \{(\mathsf{dev} \sqsubseteq \mathit{Dev})\} \quad \mathsf{eff}(\mathtt{turn\text{-}on}(\mathsf{dev})) = \{\langle \mathit{On}, \{\mathsf{dev}\}\rangle^+\}.$$

The execution of $\mathtt{turn\text{-}on}(\mathsf{dev})$ in $\mathcal{I}$ leads to an interpretation $\mathcal{I}'$, where $\mathit{On}^{\mathcal{I}'} = \{\mathsf{dev}\}$ but the extension of *ConTo* remains empty, because this fluent is not affected by $\mathtt{turn\text{-}on}(\mathsf{dev})$. Consequently, $\mathcal{I}'$ violates the global TBox

$$\mathcal{I}' \not\models \mathit{Dev} \sqcap \mathit{On} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}.$$

According to (2.4) $\mathcal{T}$ prevents the execution:

$$(\mathcal{I}, \mathtt{turn\text{-}on}(\mathsf{dev})) \notin \mathsf{Pre}_{\mathcal{T}}.$$

$\mathtt{turn\text{-}on}(\mathsf{dev})$ is not executable in $\mathcal{I}$ because dev is not connected to any power supply. In presence of the TBox preserving possibility relation there is no need to explicitly formulate

$$(x \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS})$$

as a precondition of $\mathtt{turn\text{-}on}(x)$. It is knowledge already modeled in the $\mathcal{T}$. In this case, the semantics of the possibility relation preventing $\mathtt{turn\text{-}on}(\mathsf{dev})$ leads to an intuitive behavior. The action of connecting a device to a power supply and the action of turning the device on are modeled as separate actions. The explicitly specified preconditions of both actions however do not allow a conclusion about which of the two should be done first, but the TBox does it in this case.

Next, we consider the execution of the sequence

$$\mathtt{connect}(\mathsf{dev}, \mathsf{main\text{-}socket}), \mathtt{turn\text{-}on}(\mathsf{dev}), \mathtt{disconnect}(\mathsf{dev}, \mathsf{main\text{-}socket})$$

in the interpretation $\mathcal{I}$. First, dev is connected to main-socket. This action goes through and $\mathtt{turn\text{-}on}(\mathsf{dev})$ becomes possible and its execution leads to an interpretation $\mathcal{J}$ with

$$\mathit{ConTo}^{\mathcal{J}} = \{(\mathsf{dev}, \mathsf{main\text{-}socket})\} \text{ and } \mathit{On}^{\mathcal{J}} = \{\mathsf{dev}\}.$$

However, the action $\texttt{disconnect}(\mathsf{dev}, \mathsf{main\text{-}socket})$ with the effect

$$\mathsf{eff}(\texttt{disconnect}(\mathsf{dev}, \mathsf{main\text{-}socket})) = \{\langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{main\text{-}socket})\}\rangle^-\}$$

(according to Table 2.3) is *not* possible in $\mathcal{J}$. With an execution in $\mathcal{J}$ the device dev would loose its only power supply, but would be still turned on, because there are no effects on *On* specified. This would lead again to a violation of the TBox. Hence, the disconnect action is not possible according to the semantics with qualification constraints (2.4). However, in this particular case it would be more desirable to have a semantics that allows us to use the concept inclusion as a ramification constraint to infer $\langle \mathit{On}, \{\mathsf{dev}\}\rangle^-$ as an indirect effect of the disconnection. The problem can be fixed by directly specifying the effects as follows:

$$\begin{aligned}\mathsf{eff}(\texttt{disconnect}(x, y)) = \{\, & \langle \mathit{ConTo}, \{(x, y)\}\rangle^-, \\ & (x \sqsubseteq \forall \mathit{ConTo}.(\neg \mathit{PowerS} \sqcup \{y\})) \rhd \langle \mathit{On}, \{x\}\rangle^-\end{aligned}$$

The conditional effect ensures that if y is the only power supply of x before the disconnection, then after y is disconnected from x, the device x will be off. Now, the action sequence is executable and has the expected effects. ▲

In the following we do *not* fix the TBox to be a set of qualification constraints as given in (2.4). Instead, we consider temporal specifications where the preservation of a TBox can be expressed.

2.3 Transition Systems and Temporal Logic

In this section, *transition systems* are introduced as basic models for completely describing the behavior of a dynamical systems. As a logic for specifying desired properties of such systems a first-order extension of the *propositional branching-time temporal logic* CTL* [CE81] is considered.

Transition Systems and FO-CTL*

A transition system is a directed graph where the nodes represent the states of the system and the edges are transitions between states. We distinguish different kinds of transition systems depending on the labeling of the states.

Definition 2.36. Let $\mathfrak{L}_{\mathfrak{T}}$ be a set of state labels. A *transition system* over $\mathfrak{L}_{\mathfrak{T}}$ is a tuple

$$\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}}),$$

where

- $Q_{\mathfrak{T}}$ is a *set of states* and $I_{\mathfrak{T}} \subseteq Q_{\mathfrak{T}}$ a *set of initial states*;
- $\hookrightarrow_{\mathfrak{T}} \subseteq Q_{\mathfrak{T}} \times Q_{\mathfrak{T}}$ is the *transition relation* such that for each state $q \in Q_{\mathfrak{T}}$ there is at least one state $q' \in Q_{\mathfrak{T}}$ such that $(q, q') \in \hookrightarrow_{\mathfrak{T}}$;
- $\lambda_{\mathfrak{T}} : Q_{\mathfrak{T}} \to \mathfrak{L}_{\mathfrak{T}}$ is a total *labeling function* that maps each state to an element of the *label set* $\mathfrak{L}_{\mathfrak{T}}$.

$\mathfrak{T}$ is called a *first-order transition system* iff $\mathfrak{L}_\mathfrak{T}$ is a set of first-order interpretations and the labeling function $\lambda_\mathfrak{T} : q \mapsto \mathcal{I}_q$ maps each state $q \in Q_\mathfrak{T}$ to a first-order interpretation $\mathcal{I}_q$.

Let AP be a finite set of atomic propositions (propositional letters). $\mathfrak{T}$ is called a *propositional transition system over* AP iff $\mathfrak{L}_\mathfrak{T} = 2^{\mathsf{AP}}$, i.e. the labeling function $\lambda_\mathfrak{T}$ maps each state to a set of atomic propositions from AP.

Instead of

$$(q,q') \in \hookrightarrow_\mathfrak{T} \text{ we often write } q \hookrightarrow_\mathfrak{T} q'.$$

A *path* π in $\mathfrak{T}$ is an infinite sequence of the form $\pi = q_0q_1q_2\cdots$, where $q_i \in Q_\mathfrak{T}$ and $q_i \hookrightarrow_\mathfrak{T} q_{i+1}$ holds for all $i \geq 0$. Given a path $\pi = q_0q_1q_2\cdots$ and an index $j \in \{0,1,2,\ldots\}$ the path $q_jq_{j+1}q_{j+2}\cdots$ is denoted by $\pi[j..]$ and the jth state in π is denoted by $\pi[j]$. The set of all paths in $\mathfrak{T}$ starting in a state $q \in Q_\mathfrak{T}$ is denoted by $\mathsf{paths}(\mathfrak{T},q)$. ▲

The *trace of a path* is the infinite sequence of the corresponding state labels. In [Pnu77] Pnueli introduced the propositional linear-time logic LTL for specifying and reasoning about temporal properties of infinite program executions. Propositional LTL formulas are interpreted over traces of paths representing on-going executions. Given that a program is abstracted in terms of a propositional transition system *program correctness w.r.t. a specification* that is formulated in LTL is formalized as a so-called *model checking problem*. It asks whether all the traces of all paths in a given transition system satisfy a given LTL specification.

Other popular specification languages for verification are the *propositional branching-time temporal logics* CTL and CTL* [CE81] introduced by Clarke and Emerson. Instead of a single path, a computation is viewed as an infinite tree where the branching comes from the non-determinism in the system model. In addition to LTL the logics CTL and CTL* offer explicit quantification over the different outgoing paths of a state. In CTL the use of path quantifiers is restricted and the logic is incomparable to LTL regarding expressiveness. CTL* can be viewed as an extension of LTL with path quantifiers and it subsumes both logics LTL and CTL.

In a first-order transition system such as the one induced by an FO-DS the trace of a path is an infinite sequence of FO interpretations. To talk about properties of a particular state represented by an FO interpretation one would like to use FO sentences instead of just propositional letters. For specifying desired behavior in a first-order transition system we define an extension of (propositional) CTL* called *FO-CTL**, where FO sentences are used in place of propositions. Also the corresponding DL-based fragments are defined.

Definition 2.37 (syntax of FO-CTL* formulas)**.** Formulas describing properties of a state and of a path are distinguished. *FO-CTL* state formulas* Φ and *FO-CTL* path formulas* Ψ are built according to the following syntax rules:

$$\begin{aligned} \Phi &::= \varrho \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathsf{E}\Psi \mid \mathsf{A}\Psi; \\ \Psi &::= \Phi \mid \neg\Psi \mid \Psi \wedge \Psi \mid \mathsf{X}\Psi \mid \Psi\,\mathsf{U}\,\Psi, \end{aligned} \tag{2.6}$$

where ϱ stands for an FO sentence.

Let $\mathcal{L} \subseteq \mathcal{DL}$ be a DL. We define the logic $\mathcal{L}$-CTL* as a syntactical fragment of FO-CTL*. State and path formulas are built using the rules as given above but the atomic state formula

ϱ is now restricted to be an $\mathcal{L}$-axiom according to the following rule

$$\varrho ::= C \sqsubseteq D \mid o \in C \mid (o,o') \in R \mid (o,o') \in \neg R, \tag{2.7}$$

where C and D stand for $\mathcal{L}$-concepts, o, o' for object names and R for an $\mathcal{L}$-role. The family of logics, where $\mathcal{L}$ can be instantiated with any DL is called *DL-CTL***.

The propositional fragment of FO-CTL* is obtained by restricting ϱ to be an atomic proposition. ▲

Any FO sentence is an FO-CTL* state formula. The symbols E and A are the path quantifiers. $\mathsf{E}\Psi$ is true in a state iff *there is some path* starting in this state that satisfies the FO-CTL* path formula Ψ and $\mathsf{A}\Psi$ is true iff *all paths* from this state respect Ψ. To formulate properties of paths the temporal modalities X and U are available. $\mathsf{X}\Psi$ says that Ψ holds from the next state on and $\Psi_1 \,\mathsf{U}\, \Psi_2$ expresses that Ψ_1 should hold until Ψ_2 becomes true. Additional temporal modalities for formulating path formulas are defined as abbreviations as follows

- $\mathsf{F}\Psi$ (read as "eventually Ψ") abbreviates $\textsc{True}\,\mathsf{U}\,\Psi$ and
- $\mathsf{G}\Psi$ ("globally Ψ") abbreviates $\neg\mathsf{F}\neg\Psi$,

where $\textsc{True}$ stands for a tautology.

FO-CTL* formulas are interpreted over first-order transition systems.

Definition 2.38. Let Φ be an FO-CTL* state formula, $\mathfrak{I} = (Q_\mathfrak{I}, I_\mathfrak{I}, \hookrightarrow_\mathfrak{I}, \lambda_\mathfrak{I})$ a first-order transition system over some label set consisting of interpretations and $q \in Q$ a state. *Satisfaction of* Φ *in* $\mathfrak{I}, q$, denoted by $\mathfrak{I}, q \models \Phi$, is defined inductively as follows:

$$\begin{array}{ll}
\mathfrak{I}, q \models \varrho & \text{iff } \mathcal{I}_q \models \varrho, \text{ where } \varrho \text{ is an FO sentence and } \mathcal{I}_q = \lambda_\mathfrak{I}(q),\\
\mathfrak{I}, q \models \neg\Phi' & \text{iff } \mathfrak{I}, q \not\models \Phi'\\
\mathfrak{I}, q \models \Phi_1 \wedge \Phi_2 & \text{iff } \mathfrak{I}, q \models \Phi_1 \text{ and } \mathfrak{I}, q \models \Phi_2\\
\mathfrak{I}, q \models \Phi_1 \vee \Phi_2 & \text{iff } \mathfrak{I}, q \models \Phi_1 \text{ or } \mathfrak{I}, q \models \Phi_2\\
\mathfrak{I}, q \models \mathsf{E}\Psi & \text{iff } \mathfrak{I}, \pi \models \Psi \text{ for some } \pi \in \mathsf{paths}(\mathfrak{I}, q),\\
\mathfrak{I}, q \models \mathsf{A}\Psi & \text{iff } \mathfrak{I}, \pi \models \Psi \text{ for all } \pi \in \mathsf{paths}(\mathfrak{I}, q),
\end{array}$$

satisfaction of an FO-CTL path formula* Ψ *in a path* π *in* $\mathfrak{I}$, denoted by $\mathfrak{I}, \pi \models \Psi$, is defined as follows:

$$\begin{array}{ll}
\mathfrak{I}, \pi \models \Phi & \text{iff } \mathfrak{I}, \pi[0] \models \Phi\\
\mathfrak{I}, \pi \models \neg\Psi' & \text{iff } \mathfrak{I}, \pi \not\models \Psi'\\
\mathfrak{I}, \pi \models \Psi_1 \wedge \Psi_2 & \text{iff } \mathfrak{I}, \pi \models \Psi_1 \text{ and } \mathfrak{I}, \pi \models \Psi_2\\
\mathfrak{I}, \pi \models \Psi_1 \vee \Psi_2 & \text{iff } \mathfrak{I}, \pi \models \Psi_1 \text{ or } \mathfrak{I}, \pi \models \Phi_2\\
\mathfrak{I}, \pi \models \mathsf{X}\Psi' & \text{iff } \mathfrak{I}, \pi[1..] \models \Psi'\\
\mathfrak{I}, \pi \models \Psi_1 \,\mathsf{U}\, \Psi_2 & \text{iff there exists a } j \text{ with } j \geq 0 \text{ such that } \mathfrak{I}, \pi[j..] \models \Psi_2 \text{ and}\\
& \qquad \text{for all } k \text{ with } 0 \leq k < j \text{ we have } \mathfrak{I}, \pi[k..] \models \Psi_1.
\end{array}$$

Let $\mathfrak{T} = (Q_\mathfrak{T}, I_\mathfrak{T}, \hookrightarrow_\mathfrak{T}, \lambda_\mathfrak{T})$ be a propositional transition system with the labeling function $\lambda_\mathfrak{T} : Q_\mathfrak{T} \to 2^{\mathsf{AP}}$, where AP is a finite set of atomic propositions. Furthermore, let Φ be a propositional CTL* state formula over AP and Ψ a propositional CTL* path formula over AP. Satisfaction of Φ in $\mathfrak{T}, q$ with $q \in Q_{\mathfrak{T}}$ is defined as above except for the atomic case with $\Phi = \varrho$ for some $\varrho \in \mathsf{AP}$. In this case we define

$$\mathfrak{T}, q \models \varrho \text{ iff } \varrho \in \lambda_{\mathfrak{T}}(q).$$

Satisfaction of Ψ in $\mathfrak{T}, \pi$, where π is a path in $\mathfrak{T}$, is defined as in the FO case. ▲

The definition also covers the semantics of DL-CTL*, because every DL axiom can be written as an FO sentence.

The temporal logic FO-LTL can be defined as syntactical fragment of FO-CTL*. FO-LTL formulas correspond to FO-CTL* *path* formulas without path quantifiers.

Propositional Model Checking

Next, we focus on the propositional case. The *finite-state model checking problem* takes as input a *finite* propositional transition system and a propositional CTL* state formula over a fixed set of atomic propositions. A propositional transition system is finite if the set of states is finite.

Definition 2.39 (propositional CTL* model checking)**.** Let $\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$ be a finite propositional transition system over the label set 2^{AP} and Φ a propositional CTL* state formula over AP.

We say that $\mathfrak{T}$ *models* Φ, denoted by $\mathfrak{T} \models \Phi$, iff $\mathfrak{T}, q \models \Phi$ for all initial states $q \in I_{\mathfrak{T}}$. ▲

The complexity of the model checking problem is measured w.r.t. the size of the finite propositional transition system (sum of the number of states and the number of transitions) and the length of the temporal formula.

Theorem 2.40 (complexity of model checking)**.** *Let n be the size of the finite propositional transition system and m the length of the CTL* state formula.*

1. *The propositional CTL* model checking problem is decidable with a time bound*

$$p(n) \cdot 2^{p(m)}$$

   for some polynomial p.

2. *The propositional CTL* model checking problem is decidable in* PSPACE.

Proofs and more details can be found in [BK08].

In the following we will deal with infinite first-order transition systems where the transitions between states are caused by actions. Finite propositional transition systems will serve as abstractions that allow to apply model checking algorithms.

2.4 ConGolog Programs over FO Dynamical Systems

Golog as first introduced by Levesque et al. [Lev+97] is a programming language that allows to combine actions, defined in an action theory, and tests using a set of imperative and non-deterministic programming constructs. ConGolog [DLL00] is an extension of Golog with an interleaving construct for modeling concurrency. In [DLL00] ConGolog is defined within the Situation Calculus. Here, we consider programs over actions whose meaning is defined in terms of a first-order dynamical system. The semantics of ConGolog programs is given in terms of a first-order transition system. This allows us to define the problem whether a given program satisfies a given temporal specification as a (first-order) model checking problem.

2.4.1 Syntax and Semantics of ConGolog

We focus only on a core fragment of ConGolog similar to the fragment considered in [CL08]. For instance, features like (possibly recursive) procedures or prioritized interleaving are omitted.

Definition 2.41 (syntax of ConGolog)**.** Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS. The *set of all program expressions over* $\mathfrak{D}$ is defined as the smallest set satisfying the following conditions:

1. The *empty program*, denoted by $\langle\rangle$, is a program expression over $\mathfrak{D}$.
2. Every action $\alpha(\bar{t}) \in \mathsf{Act}$ is a program expression over $\mathfrak{D}$.
3. If ψ is an FO formula over $\mathcal{F}$, then the *test* $\psi?$ is a program expression over $\mathfrak{D}$.
4. If δ is a program expression over $\mathfrak{D}$, then the *non-deterministic iteration* (δ^*) is a program expression over $\mathfrak{D}$.
5. If δ_1 and δ_2 are program expressions over $\mathfrak{D}$, then the *sequence* $(\delta_1; \delta_2)$, the *non-deterministic choice between programs* $(\delta_1 | \delta_2)$ and the *interleaving of programs* $(\delta_1 \| \delta_2)$ are also program expressions over $\mathfrak{D}$.
6. If $\bar{x}$ is a tuple of variable names, ψ an FO formula with only free occurrences of variables from $\bar{x}$ and δ a program expression over $\mathfrak{D}$, then

   $$(\mathsf{pick}(\bar{x}) \to \psi?; \delta) \text{ (\textit{non-deterministic choice of arguments})}$$

   is also a program expression over $\mathfrak{D}$. The expression $\mathsf{pick}(\bar{x}) \to \psi?$ is called *guarded pick*.

A *variable name occurs free* in a program expression over $\mathfrak{D}$ iff it is not bound by a guarded pick operator. And a program expression is called *closed* iff it has no free variables, and it is called *pick-free* iff no guarded pick expressions occur.

A *ConGolog program over* $\mathfrak{D}$ is of the form $\mathcal{P} = (\mathfrak{D}, \delta)$, where δ is a closed program expression over the FO-DS $\mathfrak{D}$. ▲

A ConGolog program consists of an FO-DS and a program expression. In this thesis, we consider FO-DSs induced by a DL-action theory or, more general, by a DL-admissible representation. However, the following definitions are independent of the concrete representation of the FO-DS. The meaning of the programming constructs can be explained as follows.

- Every atomic action is a program.

- A question mark turns an FO formula into a test. Intuitively, a program consisting of a test $\psi?$ terminates if ψ is satisfied and fails otherwise.
- The construct δ^* means executing the program δ zero or more times.
- To do $(\delta_1;\delta_2)$ one has to start with executing δ_1 and after the termination of δ_1 the execution continues with δ_2.
- The program $(\delta_1|\delta_2)$ offers a choice between program δ_1 or δ_2.
- To execute the interleaved program $(\delta_1\|\delta_2)$ one can choose among the next possible actions from δ_1 or from δ_2.
- The pick operator $(\mathsf{pick}(\bar{x}) \rightarrow \psi?;\delta)$ is used to instantiate free variables in δ. The interpreter of the program is supposed to non-deterministically choose some objects for the variables $\bar{x}$ that pass the test ψ and then the program δ is instantiated and executed with the chosen objects.

For the definition of the program semantics some further notions are needed. The meaning of a guarded pick expression that binds variables in tests and action terms is defined in terms of substitutions that replace variable names by object names.

Example 2.42. We consider a program in the device domain from Example 2.35. It one by one picks the power supplies connected to dev and disconnects them until the device is powered off. The program expression is given as follows:

$$\big(\,\mathsf{dev} \sqsubseteq \mathit{On}?;\big(\mathsf{pick}(x) \rightarrow (\mathsf{dev},x) \sqsubseteq \mathit{ConTo}?;\mathtt{disconnect}(\mathsf{dev},x)\big)\,\big)^*;\mathsf{dev} \sqsubseteq \neg\mathit{On}?$$

▲

Let δ be a program expression with free variables and ν a variable mapping, the closed program expression δ^ν is obtained from δ by simultaneously replacing each occurrence of a free variable x in δ by the object name $\nu(x)$. Next, we define the notion of a program state and a transition relation among them.

Definition 2.43. Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be a FO-DS. A *program state over* $\mathfrak{D}$ (*program state* for short) is a tuple of the form

$$\langle \mathcal{I}, \sigma, \xi \rangle,$$

where

- $\mathcal{I} \in \mathbb{I}$ represents the current state of the world,
- $\sigma \in \mathsf{ground}(\mathsf{Act})^*$ is the history of actions occurred so far and
- ξ is a closed program expression over $\mathfrak{D}$ denoting the program that remains to be executed.

The set of all program states over $\mathfrak{D}$ is denoted by $\mathsf{States}(\mathfrak{D})$. The set $\mathsf{Final}(\mathfrak{D})$ denotes the *set of all final program states over* $\mathfrak{D}$ and is defined by induction on the size of program expressions as the smallest set satisfying the following conditions:

1. $\langle \mathcal{I}, \sigma, \langle\rangle \rangle \in \mathsf{Final}(\mathfrak{D})$;
2. $\langle \mathcal{I}, \sigma, \psi? \rangle \in \mathsf{Final}(\mathfrak{D})$, if $\mathcal{I} \models \psi$;
3. $\langle \mathcal{I}, \sigma, \delta^* \rangle \in \mathsf{Final}(\mathfrak{D})$;
4. $\langle \mathcal{I}, \sigma, \delta_1; \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ and $\langle \mathcal{I}, \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$;
5. $\langle \mathcal{I}, \sigma, \delta_1 | \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ or $\langle \mathcal{I}, \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$;
6. $\langle \mathcal{I}, \sigma, \delta_1 \| \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ and $\langle \mathcal{I}, \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$;
7. $\langle \mathcal{I}, \sigma, \mathsf{pick}(\bar{x}) \to \psi?; \delta \rangle \in \mathsf{Final}(\mathfrak{D})$, if $\mathcal{I} \models \psi^{\nu}$ and $\langle \mathcal{I}, \sigma, \delta^{\nu} \rangle \in \mathsf{Final}(\mathfrak{D})$ for some ν.

A *transition relation*

$$\to_{\mathfrak{D}} \subseteq \mathsf{States}(\mathfrak{D}) \times \mathsf{States}(\mathfrak{D})$$

is defined by induction on the size of program expressions as the smallest set satisfying the following conditions:

1. $\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \langle\rangle \rangle$, if $\mathcal{I} \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}} \mathcal{I}'$ and $(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}$;
2. $\langle \mathcal{I}, \sigma, \delta^* \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta'; \delta^* \rangle$, if $\langle \mathcal{I}, \sigma, \delta \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$;
3. $\langle \mathcal{I}, \sigma, \delta_1; \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_1'; \delta_2 \rangle$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_1' \rangle$;
4. $\langle \mathcal{I}, \sigma, \delta_1; \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_2' \rangle$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ and
   $\langle \mathcal{I}, \sigma, \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_2' \rangle$;
5. $\langle \mathcal{I}, \sigma, \delta_1 | \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma, \delta' \rangle$ or $\langle \mathcal{I}, \sigma, \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma, \delta' \rangle$
6. $\langle \mathcal{I}, \sigma, \delta_1 \| \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_1' \| \delta_2 \rangle$, if $\langle \mathcal{I}, \sigma, \delta_1 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_1' \rangle$;
7. $\langle \mathcal{I}, \sigma, \delta_1 \| \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_1 \| \delta_2' \rangle$, if $\langle \mathcal{I}, \sigma, \delta_2 \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta_2' \rangle$;
8. $\langle \mathcal{I}, \sigma, \mathsf{pick}(\bar{x}) \to \psi?; \delta \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$, if $\mathcal{I} \models \psi^{\nu}$ and
   $\langle \mathcal{I}, \sigma, \delta^{\nu} \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$ for some ν.

The *set of all failure states over* $\mathfrak{D}$, denoted by $\mathsf{Fail}(\mathfrak{D})$, is defined as follows:

$$\mathsf{Fail}(\mathfrak{D}) := \{ \langle \mathcal{I}, \sigma, \delta \rangle \in \mathsf{States}(\mathfrak{D}) \mid \langle \mathcal{I}, \sigma, \delta \rangle \notin \mathsf{Final}(\mathfrak{D}) \text{ and there exists no } q \in \mathsf{States}(\mathfrak{D}) \text{ with } \langle \mathcal{I}, \sigma, \delta \rangle \to_{\mathfrak{D}} q \}.$$

▲

The transition rules and the definition of the final states are the same as the ones introduced by Claßen and Lakemeyer in their logic [CL08] for reasoning about ConGolog. They also discuss the relationship of their semantics with the classical semantics of Golog provided in terms of axioms formulated in second-order predicate logic [Lev+97; DLL00] within the Situation Calculus.

Consider the transition rule 8. dealing with a guarded pick expression. The pick non-deterministically instantiates variable names with object names. Thus, the pick only ranges over the named part of the interpretation domain. In Claßen's and Lakemeyer's semantics the pick ranges over the whole fixed domain of countably infinite standard names. Thus, under the SNA with $\mathsf{N_O}$ as the set of standard names, our semantics is compatible with Claßen's and Lakemeyer's definition. However, we don't provide a formal proof for this claim. Also note that since $\mathsf{pick}(\bar{x}) \to \psi?; \delta$ is a closed program expression, the free variables in ψ and δ are among $\bar{x}$.

The following lemma is a direct consequence of the definition. It states that a transition in "$\to_{\mathfrak{D}}$" is always caused by an executable ground action.

Lemma 2.44. *Let* $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ *be an FO-DS and let* $\langle \mathcal{I}, \sigma, \delta \rangle$ *and* $\langle \mathcal{I}', \sigma', \delta' \rangle$ *be program states over* $\mathfrak{D}$ *such that* $\langle \mathcal{I}, \sigma, \delta \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma', \delta' \rangle$. *There exists a ground action* $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ *that is a ground instance of an action term occurring in* δ *such that* $(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}$, $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{J}$ *and* $\sigma' = \sigma \cdot \boldsymbol{\alpha}$.

Proof. The proof is by induction on the structure of the closed program expression δ. For the base case where δ is a ground action the claim follows immediately from the definition of the transition relation. For complex expressions, the induction step is straightforward using the transition rules and the induction hypothesis. □

Note that in our semantics tests do not cause any transition step. Tests can be viewed as preconditions of the next primitive ground action to be executed. This is compatible with the semantics used in [CL08; SD09] but different from the original semantics in [DLL00].

Before the first-oder transition system induced by a program is defined we introduce a workaround to deal with dead-end program states that might occur in the transition relation defined above. Failure states do not have an outgoing transition and also final states do not necessarily have one. To handle non-terminating, terminating and failing executions in a uniform way some additional "dummy actions" that indicate termination and failure of a program, respectively, are defined. Note that *failure* in our case means that the program execution is stuck before reaching a final program state.

Definition 2.45 (termination and failure)**.** We use the following names from the vocabulary:

- two unary predicate names *Final* and *Fail* from $\mathsf{N_F}$;
- an object name $\mathsf{prog} \in \mathsf{N_O}$ and
- two action names $\epsilon, \mathfrak{f} \in \mathsf{N_A}$ with arity zero.

Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS, where the names *Final* and *Fail* are not contained in $\mathcal{F}$, the object name prog and the actions ϵ and $\mathfrak{f}$ are not contained in Act. The *extension of* $\mathfrak{D}$ *with* ϵ *and* $\mathfrak{f}$, denoted by $\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}$, is an FO-DS given as follows

$$\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\} = (\mathbb{I}, \mathbb{I}'_{\mathsf{ini}}, \mathcal{F} \cup \{\mathit{Final}, \mathit{Fail}\}, \mathsf{Act} \cup \{\epsilon, \mathfrak{f}\}, \mathcal{E}_{\epsilon,\mathfrak{f}},\ \mathsf{Pre}^{\epsilon,\mathfrak{f}}),$$

with

$$\mathbb{I}'_{\mathsf{ini}} = \{\mathcal{I} \in \mathbb{I}_{\mathsf{ini}} \mid \mathcal{I} \models \neg \mathit{Final}(\mathsf{prog}) \land \neg \mathit{Fail}(\mathsf{prog})\},$$

and the effect functions $\mathcal{E}_{\epsilon,\mathfrak{f}} = \langle \mathsf{add}_{\epsilon,\mathfrak{f}}, \mathsf{del}_{\epsilon,\mathfrak{f}} \rangle$ are obtained from $\mathcal{E} = \langle \mathsf{add}, \mathsf{del} \rangle$ as follows:

$$\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\alpha}, F) := \mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F) \text{ and } \mathsf{del}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\alpha}, F) := \mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F)$$

for all $\mathcal{I} \in \mathbb{I}$, $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$, $F \in \mathcal{F}$.

It remains to define the effects of $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ and the effects of the actions in $\mathsf{ground}(\mathsf{Act})$ on the names *Final* and *Fail*.

- The action $\boldsymbol{\epsilon}$ indicates termination and causes $\mathit{Final}(\mathsf{prog})$ to be true and changes nothing else. For all $\mathcal{I} \in \mathbb{I}$ we define
  - $\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\epsilon}, \mathit{Final}) := \{\mathsf{prog}^{\mathcal{I}}\}$ and $\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\epsilon}, F) := \emptyset$ for all $F \in \mathcal{F} \cup \{\mathit{Fail}\}$;
  - $\mathsf{del}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\epsilon}, F) := \emptyset$ for all $F \in \mathcal{F} \cup \{\mathit{Final}, \mathit{Fail}\}$.
- The action $\mathfrak{f}$ indicates a failure and causes $\mathit{Fail}(\mathsf{prog})$ to be true and changes nothing else. For all $\mathcal{I} \in \mathbb{I}$ we define
  - $\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \mathfrak{f}, \mathit{Fail}) := \{\mathsf{prog}^{\mathcal{I}}\}$ and $\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \mathfrak{f}, F) := \emptyset$ for all $F \in \mathcal{F} \cup \{\mathit{Final}\}$;
  - $\mathsf{del}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \mathfrak{f}, F) := \emptyset$ for all $F \in \mathcal{F} \cup \{\mathit{Final}, \mathit{Fail}\}$.
- The fluents *Final* and *Fail* are not affected by any other action different from $\boldsymbol{\epsilon}$ and $\mathfrak{f}$, respectively. For all $\mathcal{I} \in \mathbb{I}$ we define
  - $\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\alpha}, \mathit{Final}) := \mathsf{del}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\alpha}, \mathit{Final}) := \emptyset$ for all $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act}) \cup \{\mathfrak{f}\}$;
  - $\mathsf{add}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\alpha}, \mathit{Fail}) := \mathsf{del}_{\epsilon,\mathfrak{f}}(\mathcal{I}, \boldsymbol{\alpha}, \mathit{Fail}) := \emptyset$ for all $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act}) \cup \{\boldsymbol{\epsilon}\}$.

The actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ are always possible. We define

$$(\mathcal{I}, \boldsymbol{\epsilon}) \in \mathsf{Pre}^{\epsilon,\mathfrak{f}} \text{ and } (\mathcal{I}, \mathfrak{f}) \in \mathsf{Pre}^{\epsilon,\mathfrak{f}} \text{ for all } \mathcal{I} \in \mathbb{I}.$$

For all $\mathcal{I} \in \mathbb{I}$ and all $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ we define $(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}^{\epsilon,\mathfrak{f}}$ iff $(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}$. ▲

In the following we assume that all FO-DSs can be extended as defined above. We are now ready to define the first-order transition system induced by a program.

Definition 2.46. Let $\mathcal{P} = (\mathfrak{D}, \delta)$ be a ConGolog program over $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$. The *first-order transition system induced by* $\mathcal{P}$ is a transition system over the label set $\mathbb{I}$, denoted by

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}}),$$

and consists of

- the set of *states*

  $Q_{\mathcal{P}} := \{\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{States}(\mathfrak{D} \uplus \{\boldsymbol{\epsilon}, \mathfrak{f}\}) \mid$ the symbols $\boldsymbol{\epsilon}, \mathfrak{f}, \mathit{Final}, \mathit{Fail}$ do not occur in $\rho\}$;

- the set of initial states given by

  $$I_{\mathcal{P}} := \{\langle \mathcal{I}, \langle\rangle, \delta \rangle \mid \mathcal{I} \in \mathbb{I}_{\mathsf{ini}}, \mathcal{I} \models \neg \mathit{Final}(\mathsf{prog}) \land \neg \mathit{Fail}(\mathsf{prog})\}$$

  and

- the transition relation $\hookrightarrow_{\mathcal{P}} \subseteq Q_{\mathcal{P}} \times Q_{\mathcal{P}}$ that is defined as the smallest set satisfying the following conditions:
  - $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{J}, \sigma \cdot \boldsymbol{\alpha}, \xi \rangle$, if $\langle \mathcal{I}, \sigma, \rho \rangle \rightarrow_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \langle \mathcal{J}, \sigma \cdot \boldsymbol{\alpha}, \xi \rangle$;

- $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{J}, \sigma \cdot \boldsymbol{\epsilon}, \langle\rangle \rangle$, if $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D} \uplus \{\boldsymbol{\epsilon}, \mathfrak{f}\})$ and $\mathcal{I} \Rightarrow^{\boldsymbol{\epsilon}}_{\mathfrak{D}\uplus\{\boldsymbol{\epsilon},\mathfrak{f}\}} \mathcal{J}$;
- $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{J}, \sigma \cdot \mathfrak{f}, \rho \rangle$, if $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\boldsymbol{\epsilon}, \mathfrak{f}\})$ and $\mathcal{I} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D}\uplus\{\boldsymbol{\epsilon},\mathfrak{f}\}} \mathcal{J}$.

- the labeling function $\lambda_{\mathcal{P}} : \langle \mathcal{I}, \sigma, \rho \rangle \mapsto \mathcal{I}$ for each $\langle \mathcal{I}, \sigma, \rho \rangle \in Q_{\mathcal{P}}$.

▲

The program expression δ in the program $\mathcal{P} = (\mathfrak{D}, \delta)$ is a program expression over $\mathfrak{D}$. By our assumption on $\mathfrak{D}$ the names $\boldsymbol{\epsilon}$, $\mathfrak{f}$, *Final* and *Fail* are not mentioned in δ. Consequently, we also disallow these names in the program expressions of the states in the transition system $\mathfrak{T}_{\mathcal{P}}$. We use $\mathsf{States}(\mathfrak{D} \uplus \{\boldsymbol{\epsilon}, \mathfrak{f}\})$ instead of $\mathsf{States}(\mathfrak{D})$ to define $Q_{\mathcal{P}}$ because we allow that the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ occur in the action history of final and failure states. The execution of δ starts in an initial state, with the empty action history and we define that an initial state is labeled with the literals $\neg$*Final*(prog) and $\neg$*Fail*(prog). The program states are progressed according to the transition rules used to define the relation $\rightarrow_{\mathfrak{D}\uplus\{\boldsymbol{\epsilon},\mathfrak{f}\}}$. A path fragment that leads to a final state or failure state is extended to an infinite path using the actions $\boldsymbol{\epsilon}$ or $\mathfrak{f}$, respectively. $\mathfrak{T}_{\mathcal{P}}$ is a proper transition system where each state has at least one successor state (the proof is postponed to Section 3.1, Lemma 3.1).

The program semantics allows to distinguish different kinds of paths.

Definition 2.47. Let π be a path in the transition system $\mathfrak{T}_{\mathcal{P}}$ of a ConGolog program

$$\mathcal{P} = (\mathfrak{D}, \delta_0)$$

of the form

$$\pi = \langle \mathcal{I}_0, \langle\rangle, \delta_0 \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}_1, \boldsymbol{\alpha}_1, \delta_1 \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}_2, \boldsymbol{\alpha}_1\boldsymbol{\alpha}_2, \delta_2 \rangle \hookrightarrow_{\mathcal{P}} \cdots$$

with $\langle \mathcal{I}_0, \langle\rangle, \delta_0 \rangle \in I_{\mathcal{P}}$.

π is called a *non-terminating and non-failing execution* of δ_0 in $\mathcal{I}_0$, iff for all $i \geq 0$:

$$\langle \mathcal{I}_i, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_i, \delta_i \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}_{i+1}, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_i\boldsymbol{\alpha}_{i+1}, \delta_{i+1} \rangle.$$

π is a *terminating execution* iff there exists an index $j \geq 0$ such that

$\langle \mathcal{I}_j, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_j, \delta_j \rangle \in \mathsf{Final}(\mathfrak{D})$ and
$\langle \mathcal{I}_i, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_i, \delta_i \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}_{i+1}, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_i\boldsymbol{\alpha}_{i+1}, \delta_{i+1} \rangle$ for all $i = 0, \ldots, j-1$ and
$\delta_\ell = \langle\rangle, \boldsymbol{\alpha}_\ell = \boldsymbol{\epsilon}$ and $\mathcal{I}_{\ell-1} \Rightarrow^{\boldsymbol{\epsilon}}_{\mathfrak{D}\uplus\{\boldsymbol{\epsilon},\mathfrak{f}\}} \mathcal{I}_\ell$ for all $\ell > j$.

π is a *failing execution* iff there is an index $j > 0$ such that

$\langle \mathcal{I}_j, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_j, \delta_j \rangle \in \mathsf{Fail}(\mathfrak{D})$ and
$\langle \mathcal{I}_i, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_i, \delta_i \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}_{i+1}, \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_i\boldsymbol{\alpha}_{i+1}, \delta_{i+1} \rangle$ for all $i = 0, \ldots, j-1$ and
$\delta_\ell = \delta_j, \boldsymbol{\alpha}_\ell = \mathfrak{f}$ and $\mathcal{I}_{\ell-1} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D}\uplus\{\boldsymbol{\epsilon},\mathfrak{f}\}} \mathcal{I}_\ell$ for all $\ell > j$.

▲

The proof that this characterization indeed partitions the set of all paths starting in some initial state is postponed to Lemma 3.2 in Section 3.1.

With the available programming constructs while-loops and if-then-else statements can be expressed as follows:

$$\textbf{while } \psi \textbf{ do } \delta \textbf{ end} := (\psi?;\delta)^*;(\neg\psi)?;$$
$$\textbf{if } \psi \textbf{ then } \delta_1 \textbf{ else } \delta_2 \textbf{ end} := (\psi?;\delta_1) \mid ((\neg\psi)?;\delta_2).$$

For example, the program expression in Example 2.42 is a while-loop with non-deterministic choice in the body of the loop. The primitive actions definable in an FO-DS are instantaneous and not time consuming. However, simple (discrete) "durative actions" can be incorporated into a ConGolog program using interleaving. The basic idea described in [DLL00] is to decompose a complex ongoing action into a start-action initiating the process and an end-action terminating it, whereas in the meantime also other actions might occur. A similar approach is used in the PDDL standard to incorporate durative actions [FL03]. The following example describes an agent that is charging a battery.

Example 2.48. We extend the domain with devices and power supplies from Example 2.32 and 2.35. The following ABox assertions describe an initial situation

$$\mathsf{bat} \sqsubseteq (\mathit{Battery} \sqcap \mathit{PowerS} \sqcap \neg\mathit{Charging}), \quad \mathsf{dev} \sqsubseteq (\mathit{Dev} \sqcap \forall \mathit{ConTo}.\{\mathsf{bat}\} \sqcap \mathit{On}).$$

The battery bat is the only connected power supply for the device dev that is initially turned on. The agent is able to initiate the charging of bat in case the battery is not broken:

$$\mathsf{pre}(\mathtt{start\text{-}charge}(\mathsf{bat})) := \{\mathsf{bat} \sqsubseteq \mathit{Battery}\},$$
$$\mathsf{eff}(\mathtt{start\text{-}charge}(\mathsf{bat})) := \big\{\langle \mathit{Charging}, \{\mathsf{bat}\}\rangle^+\big\}.$$

There are exogenous events that stand for the end of the charging and the end of the discharging process of bat. They are modeled using the following actions:

$$\mathsf{pre}(\mathtt{recharged}(\mathsf{bat})) := \{\mathsf{bat} \sqsubseteq \mathit{Charging}\};$$
$$\mathsf{eff}(\mathtt{recharged}(\mathsf{bat})) := \big\{\langle \mathit{PowerS}, \{\mathsf{bat}\}\rangle^+, \langle \mathit{Charging}, \{\mathsf{bat}\}\rangle^-\big\};$$

$$\mathsf{pre}(\mathtt{discharged}(\mathsf{bat})) := \{\mathsf{bat} \sqsubseteq (\neg\mathit{Charging} \sqcap \mathit{PowerS})\};$$
$$\mathsf{eff}(\mathtt{discharged}(\mathsf{bat})) := \big\{\langle \mathit{PowerS}, \{\mathsf{bat}\}\rangle^-, \langle \mathit{On}, C\rangle^-\big\}, \text{ where}$$
$$C := \mathit{Dev} \sqcap \exists \mathit{ConTo}.\{\mathsf{bat}\} \sqcap \forall \mathit{ConTo}.(\neg\mathit{PowerS} \sqcup \{\mathsf{bat}\}).$$

After charging it, bat becomes a power supply in case it is not broken. The occurrence of the action discharged(bat) indicates that bat is fully discharged. Thus, it is no longer a power supply and we have the side effect that those devices that are connected to bat and are not connected to any other power supply are turned off.

The following program expression describes a non-terminating control loop of an agent that initiates the charging of the battery in case it is fully discharged and it turns the device on in case its battery is fully charged. While the battery is charging or in use the agent is waiting and executes the action `idle` with $\mathsf{pre}(\texttt{idle}) = \mathsf{eff}(\texttt{idle}) = \emptyset$.

```
δ_agent := while TRUE do
             if (bat ⊑ Charging) then
               idle;
             else
               (bat ⊑ ¬PowerS)?; start-charge(bat) |
               (bat ⊑ PowerS) ∧ (dev ⊑ ¬On)?; turn-on(dev) |
               (bat ⊑ PowerS) ∧ (dev ⊑ On)?; idle;
             end;
           end.
```

Another loop describes the exogenous actions in the environment affecting the charge of the battery:

$$\delta_{\mathsf{nature}} := \textbf{while } \textsc{True} \textbf{ do } \texttt{discharged}(\mathsf{bat}) \mid \texttt{recharged}(\mathsf{bat}) \textbf{ end}.$$

The overall behavior is described by the concurrent program $(\delta_{\mathsf{agent}} \parallel \delta_{\mathsf{nature}})$. The part of the program that describes the actions that are under the control of the agent is deterministic. Note that each action is guarded with tests that are mutually exclusive. However, the program δ_{nature} is non-deterministic which reflects the uncertainty about the length of the charging and discharging period. The capacity of the battery is unknown. For instance, the following infinite action sequence is a possible execution of $(\delta_{\mathsf{agent}} \parallel \delta_{\mathsf{nature}})$:

$$\texttt{discharged}(\mathsf{bat}) \big(\texttt{start-charge}(\mathsf{bat})\ \texttt{recharged}(\mathsf{bat})\ \texttt{discharged}(\mathsf{bat}) \big)^{\omega} \tag{2.8}$$

This particular extreme case of an execution describes the behavior in presence of an unusable battery with zero capacity. The agent is not able to turn the device on again.

Note that the example does not model the actual continuous increase and decrease of the energy level in the battery taking place between `start-charge`(bat) and `recharged`(bat) and `recharged`(bat) and `discharged`(bat), respectively. Of course, a simple discretization of these processes could be easily added within our framework using intermediate actions. Clearly, the model of the battery is rather high-level and covers only some aspects of the system. ▲

2.4.2 The Verification Problem

In this section we define the *verification problem* for ConGolog programs. The input consists of the program and a temporal property that specifies the desired behavior. The question is whether all executions of the program satisfy the temporal specification.

Definition 2.49 (verification problem). Let $\mathcal{P}$ be a ConGolog program and Φ an FO-CTL* state formula. We say that Φ *is valid in* $\mathcal{P}$ iff $\mathfrak{I}_{\mathcal{P}}, q_0 \models \Phi$ holds for all initial states $q_0 \in I_{\mathcal{P}}$ of the transition system $\mathfrak{I}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ induced by $\mathcal{P}$. Φ *is said to be satisfiable in* $\mathcal{P}$ iff $\mathfrak{I}_{\mathcal{P}}, q_0 \models \Phi$ for some initial state $q_0 \in I_{\mathcal{P}}$.

The *verification problem* is the problem of determining whether Φ is valid in $\mathcal{P}$. ▲

It holds that Φ is valid in $\mathcal{P}$ iff $\neg\Phi$ is not satisfiable in $\mathcal{P}$.

Example 2.50. We specify some properties of the program in the domain described in Example 2.48. The test guarding the turn-on action ensures that the concept inclusion in (2.5) globally holds on all paths which can be expressed with the following state formula:

$$\mathsf{AG}(\mathit{EDev} \sqcap \mathit{On} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}).$$

Thus, the problem whether all executions of a program respect a given $\mathcal{L}$-TBox constraint is expressible in $\mathcal{L}$-CTL*

The following state formula expresses that once the battery is now longer a power supply and not yet charging, the agent immediately initiates the charging process:

$$\mathsf{AG}\big(\mathsf{bat} \in (\neg \mathit{PowerS} \sqcap \neg \mathit{Charging}) \rightarrow \mathsf{AX}(\mathsf{bat} \in \mathit{Charging})\big).$$

The following path formula describes an execution where the battery is infinitely often a power supply and infinitely often not, and it supplies energy for at least one time point after charging is completed:

$$\begin{aligned}\Psi_{\mathsf{fair}} := \; & \mathsf{GF}(\mathsf{bat} \in \mathit{PowerS}) \wedge \mathsf{GF}(\mathsf{bat} \in \neg \mathit{PowerS}) \wedge \\ & \mathsf{G}\big((\mathsf{bat} \in \mathit{Charging}) \wedge \mathsf{X}(\mathsf{bat} \in \neg \mathit{Charging} \sqcap \mathit{PowerS}) \rightarrow \mathsf{XX}(\mathsf{bat} \in \mathit{PowerS})\big).\end{aligned}$$

The formula rules out executions like (2.8) and those where the battery from some time point on is charging forever, or never gets discharged. Under this assumption it holds that the device is infinitely often turned on:

$$\mathsf{A}(\Psi_{\mathsf{fair}} \rightarrow \mathsf{GF}(\mathsf{dev} \in \mathit{On}))$$

▲

Using the literals *Final*(prog) and *Fail*(prog) maintained by the termination and failure action, respectively, we can also express the verification of postconditions of programs that are supposed to be terminating. Let ψ be a Boolean KB characterizing a desired goal state. Validity of

$$\mathsf{E}(\neg \mathit{Fail}(\mathsf{prog}) \; \mathsf{U} \; (\mathit{Final}(\mathsf{prog}) \wedge \psi))$$

ensures that a non-failing and terminating execution to a goal state is possible from all initial states. To check *partial correctness* w.r.t. ψ one can verify the following formula

$$\mathsf{AG}(\mathit{Final}(\mathsf{prog}) \rightarrow \psi).$$

Total correctness of a program w.r.t. ψ corresponds to validity of $\mathsf{AF}(\mathit{Final}(\mathsf{prog}) \wedge \psi)$ in the program.

Chapter 3

Towards Decidable Fragments of ConGolog

This chapter has three parts. In the first part (Section 3.1), some properties of the program semantics are proven in order to justify the treatment of terminating and failing executions. In Section 3.2, we start our investigation of the computational properties of the verification problem. The main goal is to identify fragments of ConGolog and of the specification language for which verification is decidable. The first step is to restrict the base logic to a decidable DL. We obtain a simple DL-based setting

- by using a DL-action theory for representing the underlying dynamical system,
- by restricting the tests in the program to Boolean DL KBs and
- by formulating the temporal specification in DL-CTL*.

However, it is shown that even in a simple DL-based setting as describe above the verification problem is undecidable. Non-ground actions and the guarded pick expressions in programs are identified as one the main suspects for causing undecidability. Therefore, our focus in the subsequent sections 3.3 and 3.4 is on pick-free program expressions over a finite set of ground actions. Before concrete decidable fragments are discussed, we set up the overall framework for proving decidability. First, a finite representation of the set of all reachable subprogram expression is provided. Second, an appropriate notion of abstraction and bisimulation is established.

3.1 Termination and Failure

It remains to be shown that each state in the FO transition system induced by a ConGolog program has at least one successor state. Furthermore, we have to show that the set of all paths in the transition system (according to Definition 2.47) can be partitioned into three sets of

- non-terminating and non-failing executions,
- terminating executions, and
- failing executions.

Lemma 3.1. *Let $\mathcal{P} = (\mathfrak{D}, \delta)$ be a ConGolog program over $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$, and let $\mathfrak{T}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ be the transition system induced by $\mathcal{P}$. It holds that for every $q \in Q_{\mathcal{P}}$ there exists $q' \in Q_{\mathcal{P}}$ such that $q \hookrightarrow_{\mathcal{P}} q'$.*

Proof. Let $q \in Q_{\mathcal{P}}$ and assume $q = \langle \mathcal{I}, \sigma, \rho \rangle$. First, assume

$$q \notin \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ and } q \notin \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}).$$

By definition $q \notin \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \cup \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$ implies that there exists a successor state

$$q' \in \mathsf{States}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ with } q \rightarrow_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} q'.$$

By definition of $Q_{\mathcal{P}}$ the symbols ϵ, $\mathfrak{f}$, *Final* and *Fail* do not occur in ρ. By induction on the structure of ρ it can be shown that the same holds for the program expression in the successor state q', because the transition relation $\rightarrow_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}}$ does not introduce new symbols. It follows that $q' \in Q_{\mathcal{P}}$.

Next, assume $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$. There exists a program state

$$\langle \mathcal{I}', \sigma \cdot \epsilon, \langle\rangle \rangle \in \mathsf{States}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ with } \mathcal{I} \Rightarrow^{\epsilon}_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'.$$

Obviously, $\langle \mathcal{I}', \sigma \cdot \epsilon, \langle\rangle \rangle \in Q_{\mathcal{P}}$. By definition of $\hookrightarrow_{\mathcal{P}}$ we have

$$\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \epsilon, \langle\rangle \rangle.$$

Assume $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$. There exists a program state

$$\langle \mathcal{I}', \sigma \cdot \mathfrak{f}, \rho \rangle \in \mathsf{States}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ with } \mathcal{I} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'.$$

Obviously, $\langle \mathcal{I}', \sigma \cdot \mathfrak{f}, \rho \rangle \in Q_{\mathcal{P}}$. By definition of $\hookrightarrow_{\mathcal{P}}$ we have

$$\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \mathfrak{f}, \rho \rangle.$$

□

As a consequence we get that $\mathfrak{T}_{\mathcal{P}}$ is a well-defined FO transition system. Recall that for a program $\mathcal{P} = (\mathfrak{D}, \delta)$ the symbol "$\rightarrow_{\mathfrak{D}}$" stands for the transition relation on program states over $\mathfrak{D}$ and "$\rightarrow_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}}$" for transitions between states over $\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}$, i.e. states that are allowed to have the termination action ϵ and the action indicating failure $\mathfrak{f}$ in their action history. The actual transition relation "$\hookrightarrow_{\mathcal{P}}$" of the transition system induced by $\mathcal{P}$ extends "$\rightarrow_{\mathfrak{D}}$" by transitions in final and failure states.

The following lemma shows that the characterization in Definition 2.47 of the different paths covers all possible executions of a program.

Lemma 3.2. *Let $\mathcal{P} = (\mathfrak{D}, \delta)$ be a ConGolog program over $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$,*

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$$

the transition system induced by $\mathcal{P}$ and $\langle \mathcal{I}, \sigma, \rho \rangle \in Q_{\mathcal{P}}$ a state reachable from an initial state.

1. *There are action sequences σ' and σ'' with $\sigma = \sigma' \cdot \sigma''$ such that $\sigma' \in \mathsf{ground}(\mathsf{Act})^*$ and $\sigma'' \in \{\epsilon\}^*$ or $\sigma'' \in \{\mathfrak{f}\}^*$.*

2. *Let $\sigma \in \mathsf{ground}(\mathsf{Act})^*$. It holds that*

a) $\langle \mathcal{I}, \sigma, \rho \rangle \to_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} q'$ *iff* $\langle \mathcal{I}, \sigma, \rho \rangle \to_{\mathfrak{D}} q'$ *for all* $q' \in \mathsf{States}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$*;*

b) $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$ *iff* $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D})$*;*

c) $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$ *iff* $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D})$.

Proof. 1. The proof is by induction on the length n of the action sequence σ occurring in the reachable state $\langle \mathcal{I}, \sigma, \rho \rangle$. Obviously, the claim holds for $n = 0$.

Now, let $n > 0$ and let $\sigma = \widehat{\sigma} \cdot \boldsymbol{\alpha}$ such that

$$\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}, \sigma, \rho \rangle$$

and $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle$ is reachable from an initial state. By induction we assume that

$$\widehat{\sigma} = \widehat{\sigma}' \cdot \widehat{\sigma}'' \text{ with } \widehat{\sigma}' \in \mathsf{ground}(\mathsf{Act})^* \text{ and } \widehat{\sigma}'' \in \{\epsilon\}^* \text{ or } \widehat{\sigma}'' \in \{\mathfrak{f}\}^*.$$

It has to be shown that also σ can be written in that way. We distinguish three cases regarding $\widehat{\sigma}''$.

$\widehat{\sigma}'' = \langle\rangle$: Thus, we have $\widehat{\sigma} \in \mathsf{ground}(\mathsf{Act})^*$. According to the definition of "$\hookrightarrow_{\mathcal{P}}$" there are three possible cases. First, assume

$$\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \to_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \langle \mathcal{I}, \widehat{\sigma} \cdot \boldsymbol{\alpha}, \rho \rangle.$$

Lemma 2.44 implies that $\boldsymbol{\alpha}$ is the ground instance of an action term in ξ and $\mathcal{J} \succ^{\epsilon,\mathfrak{f}}_{\mathsf{poss}} \boldsymbol{\alpha}$ and $\mathcal{J} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}$. The action terms in ξ are contained in Act. It follows that $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$. Consequently, $\sigma = \widehat{\sigma} \cdot \boldsymbol{\alpha}$ satisfies the claim. The two remaining cases according to the definition of "$\hookrightarrow_{\mathcal{P}}$" are $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \in \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$ and $\boldsymbol{\alpha} = \boldsymbol{\epsilon}$ or $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$ and $\boldsymbol{\alpha} = \mathfrak{f}$. In both cases $\sigma = \widehat{\sigma} \cdot \boldsymbol{\alpha}$ has the desired form.

$\widehat{\sigma}'' \neq \langle\rangle \wedge \widehat{\sigma}'' \in \{\epsilon\}^*$: It follows that $\xi = \langle\rangle$ and $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \in \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$. With $\xi = \langle\rangle$ it is implied that $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle$ has no successor in "$\to_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}}$". According to the definition of $\hookrightarrow_{\mathcal{P}}$, we have $\sigma = \widehat{\sigma} \cdot \boldsymbol{\epsilon}$. Therefore, σ satisfies the claim.

$\widehat{\sigma}'' \neq \langle\rangle \wedge \widehat{\sigma}'' \in \{\mathfrak{f}\}^*$: According to the definition of $\hookrightarrow_{\mathcal{P}}$ there must be a reachable state

$$\langle \mathcal{Y}, \tilde{\sigma}, \xi \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$$

that is the predecessor of $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle$ with $\mathcal{Y} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{J}$ and $\widehat{\sigma} = \tilde{\sigma} \cdot \mathfrak{f}$. We show that $\langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$ is implied. Since $\mathcal{Y} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{J}$, the definition of $\mathcal{E}_{\epsilon,\mathfrak{f}}$ implies

$$F^{\mathcal{Y}} = F^{\mathcal{J}} \text{ for all } F \in \mathcal{F} \text{ and } o^{\mathcal{Y}} = o^{\mathcal{J}} \text{ for all } o \in \mathsf{N_O}.$$

It follows that for all $\boldsymbol{\beta} \in \mathsf{ground}(\mathsf{Act})$:

$$\mathcal{Y} \succ^{\epsilon,\mathfrak{f}}_{\mathsf{poss}} \boldsymbol{\beta} \text{ iff } \mathcal{Y} \succ_{\mathsf{poss}} \boldsymbol{\beta} \text{ iff } \mathcal{J} \succ_{\mathsf{poss}} \boldsymbol{\beta} \text{ iff } \mathcal{J} \succ^{\epsilon,\mathfrak{f}}_{\mathsf{poss}} \boldsymbol{\beta}.$$

Furthermore, all tests in ξ mention only predicates from $\mathcal{F}$. Therefore,

$$\langle \mathcal{Y}, \tilde{\sigma}, \xi \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ implies } \langle \mathcal{J}, \widehat{\sigma}, \xi \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$$

According to the definition it follows that $\sigma = \widehat{\sigma} \cdot \mathfrak{f}$ and σ is of the desired form.

2. $\langle \mathcal{I}, \sigma, \rho \rangle \in Q_{\mathcal{P}}$ with $\sigma \in \mathsf{ground}(\mathsf{Act})^*$ implies $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{States}(\mathfrak{D})$. Using a simple induction proof on the structure of ρ

$$\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ iff } \langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D}) \tag{3.1}$$

can be shown directly. Let $q' \in \mathsf{States}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\})$. The claim

$$\langle \mathcal{I}, \sigma, \rho \rangle \to_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} q' \text{ iff } \langle \mathcal{I}, \sigma, \rho \rangle \to_{\mathfrak{D}} q' \tag{3.2}$$

can be shown by induction on the structure of ρ. We only prove the base case, where ρ is of the form $\rho = \boldsymbol{\alpha}$ for some ground action $\boldsymbol{\alpha}$. $\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \in Q_{\mathcal{P}}$ implies $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$. It holds that $\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \to_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \langle\rangle \rangle$

iff $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$ and $\mathcal{I} \succ^{\epsilon, \mathfrak{f}}_{\mathsf{poss}} \boldsymbol{\alpha}$

iff $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{I}'$ and $\mathcal{I} \succ_{\mathsf{poss}} \boldsymbol{\alpha}$ (by definition of $\mathcal{E}_{\epsilon, \mathfrak{f}}$ and $\succ^{\epsilon, \mathfrak{f}}_{\mathsf{poss}}$)

iff $\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \langle\rangle \rangle$.

The third claim

$$\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}) \text{ iff } \langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D})$$

follows from (3.1) and (3.2).

□

It is a direct consequence of this lemma that there are only the three types of paths defined in Definition 2.47.

3.2 Undecidability of DL-based ConGolog with Guarded Pick

It comes as no surprise that the verification problem is undecidable even if the base logic is a decidable DL and the underlying dynamical system is represented with a local effect DL-action theory. We show undecidability for an instance of the verification problem with an input that consists of:

- a local effect $\mathcal{ALC}$-action theory $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$,
- a closed program expression δ composed of actions from Act and with tests formulated as ABox assertions and
- an $\mathcal{ALC}$-CTL* state formula Φ.

The problem is to check whether Φ is valid in $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$. We interpret the programs under the SNA with $\mathsf{N_O}$ as the set of standard names (see Definition 2.4). It turns out that one of the main sources of undecidability are the guarded pick expressions. We show undecidability by a reduction of the halting problem of two-counter machines [Min67].

Definition 3.3. A *two-counter machine* M manipulates the non-negative integer values of two counters, denoted by c_0 and c_1 in the following. A machine M is given by a finite sequence of instructions of the form

$$\mathsf{M} = \mathsf{J}_0; \cdots ; \mathsf{J}_m.$$

Let $i, j \in \{0, \ldots, m\}$ and $\ell \in \{0, 1\}$. Each instruction in the sequence $\mathsf{J}_0, \ldots, \mathsf{J}_m$ has one of the following forms:

- $\mathsf{Inc}(\ell, i)$: Increment c_ℓ by one and jump to instruction J_i.
- $\mathsf{Dec}(\ell, i, j)$: If $c_\ell = 0$ jump to J_i, else if $c_\ell > 0$ decrement c_ℓ by one and jump to J_j.
- Halt: The machine stops.

A *configuration* of M is of the form (i, n_0, n_1) where $i \in \{0, \ldots, m\}$ is the index of the instruction to be executed next and $v_0, v_1 \in \mathbb{N}$ are the values of the two counters. M induces a transition relation on configurations, denoted by $\vdash_{\mathsf{M}}$, that is defined as explained above. ▲

We assume that both counters are initialized with zero and that the execution of the machine M starts with instruction J_0. We say that M *halts* iff there exists a computation such that

$$(0, 0, 0) \vdash_{\mathsf{M}}{}^{*} (j, n_0, n_1)$$

for some $n_0, n_1 \in \mathbb{N}$ and $\mathsf{J}_j = \mathsf{Halt}$. This means that a configuration with the halting instruction is reachable from the initial configuration $(0, 0, 0)$. The problem of deciding whether a given two-counter machine halts is undecidable [Min67].

Now we describe how a machine $\mathsf{M} = \mathsf{J}_0; \cdots ; \mathsf{J}_m$ over the two counters c_0 and c_1 is simulated in a Golog program. The following concept names, object names and action terms are used:

- There are concept names $J_0, \ldots, J_m$ and *Halt* and an object name s such that the literal $J_i(s)$ is true for some $i = 0, \ldots, m$ iff the i-th instruction is the next one to be executed. $\mathit{Halt}(s)$ indicates that the machine is in its halting state.
- We use two role names P_0, P_1 and two objects a_0 and a_1 to represent the counter values. The current value of counter c_0 corresponds to the number of objects related to the object a_0 via the role P_0 and analogously the value of counter c_1 is the number of P_1-successors of a_1.
- For each counter c_ℓ, $\ell \in \{0, 1\}$ there is an increment and a decrement action, denoted by $\mathtt{inc}_\ell(a_\ell, x)$ and $\mathtt{dec}_\ell(a_\ell, x)$, respectively. Jumping to the i-th instruction with $i \in \{0, \ldots, m\}$ is done using the action $\mathtt{jump}_i(s)$. And there is an action for entering the halting state $\mathtt{halt}(s)$.

The initial situation is described by the ABox

$$\begin{aligned} \mathcal{A}_{\mathsf{M}} = \{ & s \in J_0 \sqcap \neg J_1 \sqcap \neg J_2 \sqcap \cdots \sqcap \neg J_m \sqcap \neg \mathit{Halt}, \\ & a_0 \in (\forall P_0.\bot), a_1 \in (\forall P_1.\bot)\}. \end{aligned} \tag{3.3}$$

The machine starts with the instruction J_0 and is not in the halting state. The last two assertions express that a_0 and a_1 are not related to any object via the role P_0 and P_1, respectively. In our encoding of the counter values, this corresponds to an initialization with the value zero. Next, we define the preconditions and effects of the primitive actions using the mappings pre and eff. The $\mathtt{inc}_\ell(a_\ell, x)$ action adds the argument x as a new P_ℓ-successor of a_ℓ and dually, the $\mathtt{dec}_\ell(a_\ell, x)$ action decrements the ℓ-th counter by deleting a P_ℓ-successor of a_ℓ:

$$\begin{aligned}\mathsf{pre}(\mathtt{inc}_\ell(a_\ell, x)) &= \{(a_\ell, x) \sqsubseteq \neg P_\ell\}, & \mathsf{eff}(\mathtt{inc}_\ell(a_\ell, x)) &= \left\{\langle P_\ell, \{(a_\ell, x)\}\rangle^+\right\},\\ \mathsf{pre}(\mathtt{dec}_\ell(a_\ell, x)) &= \{(a_\ell, x) \sqsubseteq P_\ell\}, & \mathsf{eff}(\mathtt{dec}_\ell(a_\ell, x)) &= \left\{\langle P_\ell, \{(a_\ell, x)\}\rangle^-\right\}\end{aligned}$$

with $\ell \in \{0,1\}$. The $\mathtt{jump}_i(s)$ action makes the label of the i-th instruction $s \sqsubseteq J_i$ true and falsifies all the other labels. $\mathtt{halt}(s)$ sets $s \sqsubseteq \mathit{Halt}$ to true. We have

$$\begin{aligned}\mathsf{pre}(\mathtt{jump}_i(s)) &= \emptyset, & \mathsf{eff}(\mathtt{jump}_i(s)) &= \left\{\langle J_i, \{s\}\rangle^+\right\} \cup \left\{\left\langle J_j, \{s\}\right\rangle^- \mid j \in \{0, \dots, m\} \text{ and } i \neq j\right\},\\ \mathsf{pre}(\mathtt{halt}(s)) &= \emptyset, & \mathsf{eff}(\mathtt{halt}(s)) &= \left\{\langle \mathit{Halt}, \{s\}\rangle^+\right\}\end{aligned}$$

with $i \in \{0, \dots, m\}$. For each instruction J_k with $k \in \{0, \dots, m\}$ a program expression δ_k is defined. In case $\mathsf{J}_k = \mathsf{Inc}(\ell, i)$ we define δ_k as follows:

$$\mathsf{pick}(x) \to ((a_\ell, x) \sqsubseteq \neg P_\ell)?; \mathtt{inc}_\ell(a_\ell, x); \mathtt{jump}_i(s).$$

An object satisfying the precondition of the increment of the ℓ-th counter is chosen, it is then executed and the label of the next instruction is made true. If $\mathsf{J}_k = \mathsf{Dec}(\ell, i, j)$, then

$$\textbf{if } a_\ell \sqsubseteq (\forall P_\ell.\bot) \textbf{ then } \mathtt{jump}_i(s) \textbf{ else } \mathsf{pick}(x) \to ((a_\ell, x) \sqsubseteq P_\ell)?; \mathtt{dec}_\ell(a_\ell, x); \mathtt{jump}_j(s) \textbf{ end}.$$

The assertion $a_\ell \sqsubseteq (\forall P_\ell.\bot)$ is tested to check whether the ℓ-th counter is zero. In the if-branch we directly jump to the i-th instruction and in the else branch the decrement is instantiated, executed and the next instruction is preset. The program for the remaining case with $\mathsf{J}_k = \mathsf{Halt}$ is just the ground action $\mathtt{halt}(s)$. The program expression δ_{M} for the machine M is assembled as follows:

$$\delta_{\mathsf{M}} := \textbf{while } \neg \mathit{Halt}(s) \textbf{ do } (s \sqsubseteq J_0?; \delta_0) \mid (s \sqsubseteq J_1?; \delta_1) \mid \cdots \mid (s \sqsubseteq J_m?; \delta_m) \textbf{ end}.$$

The overall Golog program is based on the local effect $\mathcal{ALC}$-action theory

$$\Sigma_{\mathsf{M}} = (\mathcal{K}_{\mathsf{M}}, \mathsf{Act}_{\mathsf{M}}, \mathsf{eff}, \mathsf{pre}),$$

where $\mathsf{Act}_{\mathsf{M}}$ is the set of action terms described above. The initial knowledge base $\mathcal{K}_{\mathsf{M}}$ consists only of the ABox $\mathcal{A}_{\mathsf{M}}$ (see (3.3)). The TBox is empty. It is now straightforward to prove the following lemma.

Lemma 3.4. *Let $\mathcal{P}_{\mathsf{M}} = (\mathfrak{D}(\Sigma_{\mathsf{M}}), \delta_{\mathsf{M}})$ be the Golog program simulating the two-counter machine* M *as defined above. It holds that the $\mathcal{ALC}$-CTL* state formula* $\mathsf{EF}(s \sqsubseteq \mathit{Halt})$ *is satisfiable in $\mathcal{P}_{\mathsf{M}}$ iff* M *halts.*

Recall that the variables in the tests and actions are substituted with object names from the countably infinite set $\mathsf{N_O}$ according to the semantics of guarded pick expressions. Since we have made the SNA, there is always an infinite supply of objects for the next increment of a counter. Thus, the program that implements the increment instruction never fails. This is no longer true if the SNA and UNA are both dropped. To handle also this case we only have to add the inequality assertion $a_0 \not\approx a_1$ to $\mathcal{A}_\mathsf{M}$. The preconditions and guards of the increment and decrement actions ensure that the program still counts correctly. However, there are initial states with interpretations $\mathcal{I}$ for which the set $\{o^\mathcal{I} \mid o \in \mathsf{N_O}\}$ is finite. In this case, the cardinality of $\{o^\mathcal{I} \mid o \in \mathsf{N_O}\}$ introduces an upper bound on the possible counter values. In case one of the counters tries to exceed the bound with an increment the precondition and guard ensure that the program enters a failure state. This behavior excludes false positives, i.e. initial states satisfying $\mathsf{EF}(s \sqsubseteq \mathit{Halt})$ in case M never halts. And even if the SNA and UNA are not made, a canonical interpretation that interprets all object names differently and satisfies the initial ABox $\mathcal{A}_\mathsf{M}$ obviously exists. Since Lemma 3.4 only requires satisfiability, the existence of a single initial state from which all increments go through is sufficient. Therefore, the undecidability result is not affected by the SNA or UNA.

Note that we don't use the full expressiveness of $\mathcal{ALC}$-CTL*. The property $\mathsf{EF}(s \sqsubseteq \mathit{Halt})$ is formulated within the CTL fragment of $\mathcal{ALC}$-CTL*. Furthermore, the halting problem can be also formulated as an $\mathcal{ALC}$-LTL verification problem. Obviously, the state formula $\mathsf{EF}s \sqsubseteq \mathit{Halt})$ is satisfied in some initial state iff there exists an infinite path satisfying the $\mathcal{ALC}$-LTL formula $\mathsf{F}(s \sqsubseteq \mathit{Halt})$. Undecidability also carries over to the verification of $\mathcal{ALC}$-CTL and $\mathcal{ALC}$-LTL specifications.

Theorem 3.5. *Verifying $\mathcal{ALC}$-CTL properties of Golog programs based on local effect $\mathcal{ALC}$-action theories is undecidable.*

For the reduction we have used only a small fragment of $\mathcal{ALC}$. The value restriction $\forall P_\ell.\bot$ in the initial ABox is the only complex concept occurring in the program.

It turns out that guarded pick expressions cause problems even if we further restrict the other available programming constructs and also disallow iteration. Under the SNA the pick can act as an additional existential quantifier. We can exploit this feature to reduce Boolean conjunctive query entailment w.r.t. an $\mathcal{ALC}$-TBox to the verification problem.

Definition 3.6. A *Boolean conjunctive query* (BCQ) is an FO sentence of the form $\exists\bar{x}.\psi$, where ψ is a conjunction of positive and negative literals with unary and binary predicate names. All variable names in ψ are among $\bar{x}$. Let $\mathcal{I}$ be an interpretation. Satisfaction of $\exists\bar{x}.\psi$ in $\mathcal{I}$ is defined using the usual first-order logic semantics. A *BCQ $\exists\bar{x}.\psi$ is entailed by an $\mathcal{ALC}$-TBox $\mathcal{T}$* iff $\exists\bar{x}.\psi$ is satisfied in all models of $\mathcal{T}$. ▲

In this general case BCQ entailment w.r.t. an $\mathcal{ALC}$-TBox is undecidable as shown by Rosati [Ros07]. The undecidability result also holds for BCQs without object names. In this case, it is obvious that the SNA is without loss of generality. For an $\mathcal{ALC}$-TBox $\mathcal{T}$ and BCQ $\exists\bar{x}.\psi$ without object names it holds that $\exists\bar{x}.\psi$ is satisfied in all models of $\mathcal{T}$ iff $\exists\bar{x}.\psi$ is satisfied in all models of $\mathcal{T}$ that satisfy the SNA.

Let $\mathcal{T}$ be an $\mathcal{ALC}$-TBox and $\exists\bar{x}.\psi$ a BCQ without object names. ψ can be viewed as a conjunction of ABox literals. We define a local effect $\mathcal{ALC}$-action theory with a single ground action:

$$\Sigma = (\mathcal{K} = (\mathcal{T}, \mathcal{A} = \{a \sqsubseteq \neg A\}), \mathsf{Act} = \{\alpha(a)\}, \mathsf{pre}, \mathsf{eff}),$$

where $\mathsf{pre}(\alpha(a)) := \emptyset$ and $\mathsf{eff}(\alpha(a)) := \left\{\langle A, \{a\}\rangle^{+}\right\}$ and $a \in \mathsf{N_O}$ and A is a concept name not mentioned in $\mathcal{T}$. A guarded pick expression that checks $\exists\bar{x}.\psi$ is given as follows:

$$\delta = \mathsf{pick}(\bar{x}) \rightarrow \psi?; \alpha(a).$$

Once the guard is passed the literal $a \sqsubseteq A$ is set to true by $\alpha(a)$. The $\mathcal{ALC}$-CTL state formula $\mathsf{EX}(a \sqsubseteq A)$ is satisfied in an initial state $\langle \mathcal{I}, \langle\rangle, \delta\rangle$ of the program $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$ iff an instantiation of the variables $\bar{x}$ can be chosen such that the instantiated ψ is satisfied in the model $\mathcal{I}$ of $\mathcal{T}$. It is easy to see that $\mathsf{EX}(a \sqsubseteq A)$ is valid in $\mathcal{P}$ iff $\exists\bar{x}.\psi$ is entailed by $\mathcal{T}$.

Theorem 3.7. *Verifying $\mathcal{ALC}$-CTL properties of Golog programs based on local effect $\mathcal{ALC}$-action theories under the SNA is undecidable even if iteration is disallowed and only a single ground action is used.*

3.3 Reachable Subprogram Expressions

In this section ConGolog programs $\mathcal{P} = (\mathfrak{D}, \delta)$ are considered, where δ is pick-free. In this cases the program expression δ only mentions a finite set of ground actions. For this restricted programs we introduce a finite representation of the set of all reachable subprograms. The results in this section are independent of a particular representation of $\mathfrak{D}$.

We start with recapitulating the possible *sources of infiniteness* in the transition system of a program. States are triples consisting of a first-order interpretation, a ground action sequence and a closed program expression. The initial KB of the underlying dynamical system provides only incomplete information and has infinitely many models. This leads to infinitely many initial states in the transition system of the program. In the third component of each state, we keep track of the program expression representing the part of the program that remains to be executed. Fortunately, this state component is *not* an additional source of infiniteness if only ground actions are used and we show in the following that there are only finitely many reachable subprograms. This is an important auxiliary result we later use to show decidability of the verification problem in fragments of ConGolog.

To define the set of all reachable subprograms, we consider the “symbolic execution” of a program. For this purpose, a program expression is split up into its atomic pieces that are then executed one by one. Such an atomic piece is called *guarded action*. It is a ground action preceded by a (possibly empty) sequence of tests.

Definition 3.8. Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS. A program expression over $\mathfrak{D}$ is called *guarded action* if it is of the form

$$\psi_1?; (\psi_2?; (\ldots; (\psi_n?; \boldsymbol{\alpha}))),$$

where $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$, $n \geq 0$ and each $\psi_i?$ for $i = 1, \cdots, n$ is a test. We will often use the symbol $\mathfrak{a}$ to denote a guarded action. If $n = 0$, then the guarded action is actually an ordinary

$$
\begin{aligned}
&\mathsf{head}(\langle\rangle) := \{\epsilon\};\\
&\mathsf{head}(\boldsymbol{\alpha}) := \{\boldsymbol{\alpha}\} \text{ for all } \boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act});\\
&\mathsf{head}(\psi?) := \{\psi?;\epsilon\};\\
&\mathsf{head}(\delta^*) := \{\epsilon\} \cup \mathsf{head}(\delta);\\
&\mathsf{head}(\delta_1;\delta_2) := \{\mathfrak{a} \mid \mathfrak{a} = \psi_1?;...;\psi_n?;\boldsymbol{\alpha} \in \mathsf{head}(\delta_1) \wedge \boldsymbol{\alpha} \neq \epsilon\} \cup\\
&\qquad\qquad \{\psi_1?;\ldots;\psi_n?;\mathfrak{a} \mid \psi_1?;...;\psi_n?;\epsilon \in \mathsf{head}(\delta_1) \ \wedge \mathfrak{a} \in \mathsf{head}(\delta_2)\};\\
&\mathsf{head}(\delta_1|\delta_2) := \mathsf{head}(\delta_1) \cup \mathsf{head}(\delta_2);\\
&\mathsf{head}(\delta_1\|\delta_2) := \{\mathfrak{a} \mid \mathfrak{a} = \psi_1?;\ldots;\psi_n?;\boldsymbol{\alpha} \in \mathsf{head}(\delta_i) \wedge i \in \{1,2\} \wedge \alpha \neq \epsilon\} \cup\\
&\qquad\qquad \{\psi_1?;\ldots;\psi_n?;\mathfrak{a} \mid \psi_1?;\ldots;\psi_n?;\epsilon \in \mathsf{head}(\delta_i) \wedge\\
&\qquad\qquad\qquad \mathfrak{a} \in \mathsf{head}(\delta_j) \ \wedge i,j \in \{1,2\}, i \neq j\};
\end{aligned}
$$

Figure 3.1: Head of a program expression

ground action, and thus $\mathfrak{a}$ may also denote a ground action. The preceding sequence of tests is called *guard*. When writing a guarded action we will often omit the parentheses.

Let $\mathcal{I}$ be an interpretation. The guarded action $\psi_1?;\cdots;\psi_n?;\boldsymbol{\alpha}$ is *executable in* $\mathcal{I}$ iff

$$\mathcal{I} \models \psi_i \text{ for all } i = 1,\ldots,n \text{ and } \mathcal{I} \succ_{\mathsf{poss}} \boldsymbol{\alpha}.$$

▲

Next, we introduce two functions $\mathsf{head}(\cdot)$ and $\mathsf{tail}(\cdot,\cdot)$. Intuitively, $\mathsf{head}(\delta)$ contains those guarded actions that can be executed first when executing the program expression δ. For $\mathfrak{a} \in \mathsf{head}(\delta)$, $\mathsf{tail}(\mathfrak{a},\delta)$ yields the remainder of the program, i.e., the part that still needs to be executed after $\mathfrak{a}$ has been executed. Due to the non-deterministic nature of ConGolog programs, $\mathsf{tail}(\mathfrak{a},\delta)$ is also a set of program expressions rather than a single one.

In the remainder of this section we always assume that program expressions are pick-free.

The function $\mathsf{head}(\cdot)$ is formally defined as follows.

Definition 3.9. The function $\mathsf{head}(\cdot)$ maps a program expression over some first-oder dynamical system $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ to a set of guarded actions over $\mathfrak{D} \uplus \{\epsilon, \mathfrak{f}\}$. It is defined by induction on the structure of program expressions as given in Figure 3.1. ▲

The empty program represents the final state which means that ϵ is executed next. Since tests do not cause a separate execution step, the head of a test is given by the termination action ϵ preceded by the test itself as a guard. Executing δ^* means executing δ zero ore more times. Hence, the head of δ^* consists of the termination action ϵ and the heads of δ. Consider the definition of $\mathsf{head}(\delta_1;\delta_2)$. In this case, we first have to execute the program δ_1. Therefore, the first guarded action to be executed for the sequence is one of the heads of δ_1. However, if $\psi_1?;\ldots;\psi_n?;\epsilon$ is contained in the head of δ_1, then δ_1 can terminate successfully if the tests are satisfied. But in this case the subsequent program δ_2 still needs to be executed. Therefore, we must continue with a head of δ_2. This is achieved by replacing ϵ

in $\psi_1?;\cdots;\psi_n?;\epsilon$ with a head of δ_2. Our definition of $\mathsf{head}(\delta_1\|\delta_2)$ can be explained in a similar way. To do $\delta_1|\delta_2$ a head of δ_1 or one of δ_2 has to be done in the next step.

Next, we need to define the program(s) that remain to be executed once a guarded action from the head has been executed.

Definition 3.10. The function $\mathsf{tail}(\cdot,\cdot)$ maps a guarded action over $\mathfrak{D}\uplus\{\epsilon,\mathfrak{f}\}$ and a program expression over $\mathfrak{D}$ to a set of program expressions over $\mathfrak{D}$.

- If $\mathfrak{a}\notin\mathsf{head}(\delta)$, then $\mathsf{tail}(\mathfrak{a},\delta)=\emptyset$.
- If $\mathfrak{a}\in\mathsf{head}(\delta)$ and $\mathfrak{a}=\psi_1?;...;\psi_n?;\epsilon$ for some $n\geq 0$, then $\mathsf{tail}(\mathfrak{a},\delta)=\{\langle\rangle\}$.
- If $\mathfrak{a}\in\mathsf{head}(\delta)$ and $\mathfrak{a}=\psi_1?;...;\psi_n?;\boldsymbol{\alpha}$ with $\boldsymbol{\alpha}\neq\epsilon$ for some $n\geq 0$, then $\mathsf{tail}(\mathfrak{a},\delta)$ is defined by induction on size of δ as given in Figure 3.2.

▲

$$
\begin{aligned}
&\mathsf{tail}(\mathfrak{a},\boldsymbol{\alpha}) := \{\langle\rangle\}\\
&\mathsf{tail}(\mathfrak{a},\psi?) := \{\langle\rangle\}\\
&\mathsf{tail}(\mathfrak{a},\delta^*) := \{\delta';(\delta)^* \mid \delta'\in\mathsf{tail}(\mathfrak{a},\delta)\};\\
&\mathsf{tail}(\mathfrak{a},\delta_1;\delta_2) := \{\delta';\delta_2 \mid \delta'\in\mathsf{tail}(\mathfrak{a},\delta_1)\}\cup\\
&\qquad\{\delta'' \mid \exists j, 0\leq j\leq n \text{ such that } \psi_1?;...;\psi_j?;\epsilon\in\mathsf{head}(\delta_1)\wedge\\
&\qquad\quad \psi_{j+1}?;...;\psi_n?;\boldsymbol{\alpha}\in\mathsf{head}(\delta_2)\wedge\\
&\qquad\quad \delta''\in\mathsf{tail}(\psi_{j+1}?;...;\psi_n?;\boldsymbol{\alpha},\delta_2)\};\\
&\mathsf{tail}(\mathfrak{a},\delta_1|\delta_2) := \mathsf{tail}(\mathfrak{a},\delta_1)\cup\mathsf{tail}(\mathfrak{a},\delta_2);\\
&\mathsf{tail}(\mathfrak{a},\delta_1\|\delta_2) := \{\delta'\|\delta_2 \mid \delta'\in\mathsf{tail}(\mathfrak{a},\delta_1)\}\cup\{\delta_1\|\delta' \mid \delta'\in\mathsf{tail}(\mathfrak{a},\delta_2)\}\cup\\
&\qquad\{\delta'' \mid \exists j, 0\leq j\leq n \text{ such that } \psi_1?;...;\psi_j?;\epsilon\in\mathsf{head}(\delta_i)\wedge\\
&\qquad\quad \psi_{j+1}?;...;\psi_n?;\boldsymbol{\alpha}\in\mathsf{head}(\delta_{i'})\wedge\\
&\qquad\quad \delta''\in\mathsf{tail}(\psi_{j+1}?;...;\psi_n?;\boldsymbol{\alpha},\delta_{i'})\wedge i,i'\in\{1,2\}, i\neq i'\};
\end{aligned}
$$

In the definitions of $\mathsf{tail}(\mathfrak{a},\delta^*)$, $\mathsf{tail}(\mathfrak{a},\delta_1;\delta_2)$, and $\mathsf{tail}(\mathfrak{a},\delta_1\|\delta_2)$, we omit δ' if $\delta'=\langle\rangle$.

Figure 3.2: Tail of a program expression and one of its heads

Intuitively, executing a program "symbolically" means first executing a guarded action of its head, then a guarded action of the head of its tail, etc. We call a program expression that can be reached by a sequence of such head and tail applications a reachable subprogram.

Definition 3.11. Let δ be a program expression over some FO-DS $\mathfrak{D}$. The program expression ρ over $\mathfrak{D}$ is a *reachable subprogram* of δ if there is an $n\geq 0$ and program expressions over $\mathfrak{D}$ $\delta_0,\delta_1,\ldots,\delta_n$ such that $\delta_0=\delta$, $\delta_n=\rho$, and for all $i=0,\cdots,n-1$ there exists $\mathfrak{a}_i\in\mathsf{head}(\delta_i)$ such that $\delta_{i+1}\in\mathsf{tail}(\mathfrak{a}_i,\delta_i)$. We denote the *set of all reachable subprograms of* δ by $\mathsf{sub}(\delta)$. We say that the program expression ρ is reachable in $n \geq 0$ steps from δ if a sequence of program expressions of length n satisfying the conditions above exists. The set $\mathsf{sub}^n(\delta) \subseteq \mathsf{sub}(\delta)$ denotes the set of all subprograms reachable in n steps from δ. ▲

One can also obtain a graph representation of $\mathsf{sub}(\delta)$ and define

$$\delta \xrightarrow{\mathfrak{a}} \delta' \text{ iff } \mathfrak{a} \in \mathsf{head}(\delta) \wedge \delta' \in \mathsf{tail}(\mathfrak{a}, \delta).$$

The resulting graph corresponds to the *characteristic graphs* of ConGolog programs used in [CL08]. We show that the definitions of head and tail are correct by establishing the correspondence with the transition semantics given in Definition 2.43.

Lemma 3.12. *Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS and $\langle \mathcal{I}, \sigma, \delta \rangle \in \mathsf{States}(\mathfrak{D})$ a program state over $\mathfrak{D}$.*

1. *$\langle \mathcal{I}, \sigma, \delta \rangle \in \mathsf{Final}(\mathfrak{D})$ iff there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\epsilon} \in \mathsf{head}(\delta)$ for some $n \geq 0$ such that $\mathfrak{a}$ is executable in $\mathcal{I}$.*
2. *$\langle \mathcal{I}, \sigma, \delta \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$ iff there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\delta)$ for some $n \geq 0$ and $\boldsymbol{\alpha} \neq \boldsymbol{\epsilon}$ such that $\delta' \in \mathsf{tail}(\mathfrak{a}, \delta)$ and $\mathfrak{a}$ is executable in $\mathcal{I}$.*

Proof. We show the claims by induction on the structure of δ.

$\delta = \langle\rangle$:

1. By definition of the set of final states we have $\langle \mathcal{I}, \sigma, \langle\rangle \rangle \in \mathsf{Final}(\mathfrak{D})$ for all interpretations and action sequences. The definition of head yields $\boldsymbol{\epsilon} \in \mathsf{head}(\langle\rangle)$ and $\boldsymbol{\epsilon}$ is executable in all interpretations.
2. According to the transition rules states of the form $\langle \mathcal{I}, \sigma, \langle\rangle \rangle$ do not have any outgoing transition w.r.t. "$\to_{\mathfrak{D}}$". And it holds that $\mathsf{head}(\langle\rangle) = \{\boldsymbol{\epsilon}\}$. This implies the claim.

$\delta = \boldsymbol{\alpha}$:

1. By definition of the set of final states we have $\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \notin \mathsf{Final}(\mathfrak{D})$ for all interpretations and action sequences. Since program expressions do not contain the action $\boldsymbol{\epsilon}$ we have $\boldsymbol{\alpha} \neq \boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon} \notin \mathsf{head}(\boldsymbol{\alpha}) = \{\boldsymbol{\alpha}\}$.
2. For all states of the form $\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \in \mathsf{States}(\mathfrak{D})$ we have $\boldsymbol{\alpha} \notin \{\boldsymbol{\epsilon}, \mathfrak{f}\}$. It holds that

$$\langle \mathcal{I}, \sigma, \boldsymbol{\alpha} \rangle \to_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$$

   iff $\delta' = \langle\rangle$ and $\mathcal{I} \succ_{\mathsf{poss}} \boldsymbol{\alpha}$ (by definition of "$\to_{\mathfrak{D}}$")

   iff $\mathsf{head}(\boldsymbol{\alpha}) = \{\boldsymbol{\alpha}\}$, $\mathsf{tail}(\boldsymbol{\alpha}, \boldsymbol{\alpha}) = \{\langle\rangle\}$ and $\boldsymbol{\alpha}$ is executable in $\mathcal{I}$ (by definition of head and tail).

$\delta = \psi?$:

1. Obviously, it holds that $\mathsf{head}(\psi?) = \{\psi?; \boldsymbol{\epsilon}\}$. We have $\langle \mathcal{I}, \sigma, \psi? \rangle \in \mathsf{Final}(\mathfrak{D})$

iff $\mathcal{I} \models \psi$

iff the guarded action $\psi?;\boldsymbol{\epsilon}$ is executable in $\mathcal{I}$, because $\boldsymbol{\epsilon}$ is possible in all interpretations.

2. States of the form $\langle \mathcal{I}, \sigma, \psi? \rangle$ do not have any outgoing transition w.r.t. "$\rightarrow_{\mathfrak{D}}$" and $\mathsf{head}(\psi?) = \{\psi?;\boldsymbol{\epsilon}\}$ which implies the claim.

$\delta = \delta_1;\delta_2$:

1. It holds that $\langle \mathcal{I}, \sigma, \delta_1;\delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$

   iff $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ and $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ (by definition of the set of final states)

   iff there are guarded actions with

$$\psi_1?;\cdots;\psi_n?;\boldsymbol{\epsilon} \in \mathsf{head}(\delta_1) \text{ and } \psi'_1?;\cdots;\psi'_m?;\boldsymbol{\epsilon} \in \mathsf{head}(\delta_2)$$

   for some $n, m \geq 0$ such that both are executable in $\mathcal{I}$ (by using the induction hypothesis)

   iff there exists a guarded action $\psi_1?;\cdots;\psi_n?;\psi'_1?;\cdots;\psi'_m?;\boldsymbol{\epsilon} \in \mathsf{head}(\delta_1;\delta_2)$ for some $n, m \geq 0$ that is executable in $\mathcal{I}$ (by using the definition of the head function and the definition of executability of guarded actions).

2. It holds that $\langle \mathcal{I}, \sigma, \delta_1;\delta_2 \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$

   iff one of the following is true

   - $\delta' = \delta'_1;\delta_2$ such that $\langle \mathcal{I}, \sigma, \delta_1 \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta'_1 \rangle$ or
   - $\delta' = \delta'_2$ such that $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ and $\langle \mathcal{I}, \sigma, \delta_2 \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta'_2 \rangle$

   (by definition of "$\rightarrow_{\mathfrak{D}}$")

   iff one of the following is true

   - $\delta' = \delta'_1;\delta_2$ and there exists a guarded action $\mathfrak{a} = \psi_1?;\cdots;\psi_n?;\boldsymbol{\alpha} \in \mathsf{head}(\delta_1)$ for some $n \geq 0$ such that $\boldsymbol{\alpha} \notin \{\boldsymbol{\epsilon}, \mathfrak{f}\}$ and $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\delta'_1 \in \mathsf{tail}(\mathfrak{a}, \delta_1)$
   - $\delta' = \delta'_2$ and there exists a guarded action $\psi'_1?;\cdots;\psi'_m?;\boldsymbol{\epsilon} \in \mathsf{head}(\delta_1)$ for some $m \geq 0$ such that $\psi'_1?;\cdots;\psi'_m?;\boldsymbol{\epsilon}$ is executable in $\mathcal{I}$ and there exists a guarded action $\mathfrak{a}' = \widehat{\psi}_1?;\cdots;\widehat{\psi}_k?;\boldsymbol{\alpha} \in \mathsf{head}(\delta_2)$ for some $k \geq 0$ such that $\boldsymbol{\alpha} \notin \{\boldsymbol{\epsilon}, \mathfrak{f}\}$ and $\mathfrak{a}'$ is executable in $\mathcal{I}$ and $\delta'_2 \in \mathsf{tail}(\mathfrak{a}', \delta_2)$

   (by induction and claim 1.).

   iff one of the following is true

   - $\delta' = \delta'_1;\delta_2$ and there exists a guarded action $\mathfrak{a} = \psi_1?;\cdots;\psi_n?;\boldsymbol{\alpha} \in \mathsf{head}(\delta_1;\delta_2)$ for some $n \geq 0$ such that $\boldsymbol{\alpha} \notin \{\boldsymbol{\epsilon}, \mathfrak{f}\}$ and $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\delta' \in \mathsf{tail}(\mathfrak{a}, \delta_1;\delta_2)$
   - $\delta' = \delta'_2$ and there exists a guarded action

$$\widehat{\mathfrak{a}} = \psi'_1?;\cdots;\psi'_m?;\widehat{\psi}_1?;\cdots;\widehat{\psi}_k?;\boldsymbol{\alpha} \in \mathsf{head}(\delta_1;\delta_2)$$

for some $m, k \geq 0$ such that $\boldsymbol{\alpha} \notin \{\epsilon, \mathfrak{f}\}$ and $\widehat{\mathfrak{a}}$ is executable in $\mathcal{I}$ and $\delta' \in \mathsf{tail}(\widehat{\mathfrak{a}}, \delta_1; \delta_2)$

(by definition of head and tail). This finishes the proof of the claim.

$\delta = \zeta^*$:

1. By definition of the set of final states we have $\langle \mathcal{I}, \sigma, \zeta^* \rangle \in \mathsf{Final}(\mathfrak{D})$ for all interpretations and action sequences and by definition of the head function we have $\epsilon \in \mathsf{head}(\zeta^*)$ and ϵ is possible in any interpretation.
2. It holds that $\langle \mathcal{I}, \sigma, \zeta^* \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$

   iff $\delta' = \zeta'; (\zeta)^*$ and $\langle \mathcal{I}, \sigma, \zeta \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \zeta' \rangle$ (by definition of "$\rightarrow_{\mathfrak{D}}$")

   iff $\delta' = \zeta'; (\zeta)^*$ and there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\zeta)$ for some $n \geq 0$ such that $\boldsymbol{\alpha} \notin \{\epsilon, \mathfrak{f}\}$, $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\zeta' \in \mathsf{tail}(\mathfrak{a}, \zeta)$ (by induction)

   iff $\delta' = \zeta'; (\zeta)^*$ and there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\zeta^*)$ for some $n \geq 0$ such that $\boldsymbol{\alpha} \notin \{\epsilon, \mathfrak{f}\}$, $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\delta' \in \mathsf{tail}(\mathfrak{a}, \zeta^*)$ (by definition of head and tail).

$\delta = \delta_1 | \delta_2$:

1. It holds that $\langle \mathcal{I}, \sigma, \delta_1 | \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$

   iff $\langle \mathcal{I}, \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D})$ or $\langle \mathcal{I}, \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D})$ (by definition of the set of final state)

   iff there exists a guarded action $\psi_1?; \cdots; \psi_n?; \epsilon \in \mathsf{head}(\delta_1)$ for some $n \geq 0$ such that $\psi_1?; \cdots; \psi_n?; \epsilon$ is executable in $\mathcal{I}$ or there exists a guarded action $\psi'_1?; \cdots; \psi'_m?; \epsilon \in \mathsf{head}(\delta_2)$ for some $m \geq 0$ and $\psi'_1?; \cdots; \psi'_m?; \epsilon$ is executable in $\mathcal{I}$ (by using the induction hypothesis)

   iff there exists a guarded action $\widehat{\psi_1}?; \cdots; \widehat{\psi_k}?; \epsilon \in \mathsf{head}(\delta_1 | \delta_2)$ for some $k \geq 0$ such that $\widehat{\psi_1}?; \cdots; \widehat{\psi_k}?; \epsilon$ is executable in $\mathcal{I}$ (by definition of the head function).
2. It holds that $\langle \mathcal{I}, \sigma, \delta_1 | \delta_2 \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$

   iff one of the following is true

   - $\langle \mathcal{I}, \sigma, \delta_1 \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$ or
   - $\langle \mathcal{I}, \sigma, \delta_2 \rangle \rightarrow_{\mathfrak{D}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \delta' \rangle$

   (by definition of "$\rightarrow_{\mathfrak{D}}$")

   iff one of the following is true

   - there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\delta_1)$ for some $n \geq 0$ such that $\boldsymbol{\alpha} \notin \{\epsilon, \mathfrak{f}\}$, $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\delta' \in \mathsf{tail}(\mathfrak{a}, \delta_1)$ or
   - there exists a guarded action $\mathfrak{a}' = \psi'_1?; \cdots; \psi'_m?; \boldsymbol{\alpha} \in \mathsf{head}(\delta_2)$ for some $m \geq 0$ such that $\boldsymbol{\alpha} \notin \{\epsilon, \mathfrak{f}\}$, $\mathfrak{a}'$ is executable in $\mathcal{I}$ and $\delta' \in \mathsf{tail}(\mathfrak{a}', \delta_2)$

(by induction)

iff there exists a guarded action $\widehat{\mathfrak{a}} = \widehat{\psi_1}?; \cdots ; \widehat{\psi_k}?; \boldsymbol{\alpha} \in \mathsf{head}(\delta_1|\delta_2)$ for some $k \geq 0$ such that $\widehat{\mathfrak{a}}$ is executable in $\mathcal{I}$ and $\delta' \in \mathsf{tail}(\widehat{\mathfrak{a}}, \delta_1|\delta_2)$ (by definition of the head and tail function).

$\delta = \delta_1 \| \delta_2$: We omit the proof. It is similar to the case with $\delta = \delta_1; \delta_2$.

□

The program expressions occurring in the reachable part of the transition system of a ConGolog program $\mathcal{P} = (\mathfrak{D}, \delta)$, where δ is pick-free, are contained in $\mathsf{sub}(\delta)$. It also follows that in a reachable failure state $\langle \mathcal{I}, \sigma, \zeta \rangle \in \mathsf{Fail}(\mathfrak{D})$ of $\mathcal{P}$ no guarded action in the head of ζ is executable in $\mathcal{I}$.

The next step is to show that the cardinality of $\mathsf{sub}(\delta)$ is finite and bounded by the length of δ. It might be the case that a subprogram expression $\rho \in \mathsf{sub}(\delta)$ of δ is longer than δ itself in presence of the iteration constructor. First, we define the *length* and the *star height* of a program expression.

Definition 3.13. Let δ be a program expression over some FO-DS $\mathfrak{D}$. The *length of* δ, denoted by $|\delta|$, is defined as follows:

$$\begin{aligned}
&|\langle\rangle| && := 1;\\
&|\alpha(\bar{t})| && := 1 \text{ for some action term } \alpha(\bar{t});\\
&|\psi?| && := 1;\\
&|\delta^*| && := |\delta| + 1;\\
&|\delta_1 \star \delta_2| && := |\delta_1| + |\delta_2| + 1 \text{ with } \star \in \{;, |, \|\}.
\end{aligned}$$

The *star height of* δ, denoted by $\mathsf{h}(\delta)$, is the maximal nesting depth of the iteration construct $\cdot^*$ and is defined as follows:

$$\begin{aligned}
&\mathsf{h}(\langle\rangle) && := 0;\\
&\mathsf{h}(\alpha(\bar{t})) && := 0 \text{ for some action term } \alpha(\bar{t});\\
&\mathsf{h}(\psi?) && := 0;\\
&\mathsf{h}(\delta^*) && := \mathsf{h}(\delta) + 1;\\
&\mathsf{h}(\delta_1 \star \delta_2) && := \max(\mathsf{h}(\delta_1), \mathsf{h}(\delta_2)) \text{ with } \star \in \{;, |, \|\}.
\end{aligned}$$

▲

Before we establish a bound on the cardinality of $\mathsf{sub}(\delta)$ and on the length of subprogram expressions, another auxiliary lemma about the structure of subprogram expressions is needed.

Lemma 3.14. *Let δ, δ_1 and δ_2 be program expressions over some FO-DS $\mathfrak{D}$.*

1. *For $\rho \in \mathsf{sub}(\delta^*)$ it holds that ρ is of the form $\langle\rangle$, $(\delta)^*$ or $\delta'; (\delta)^*$ with $\delta' \in \mathsf{sub}(\delta)$.*

2. *For $\rho \in \mathsf{sub}(\delta_1; \delta_2)$ it holds that ρ is of the form (i) $\delta_1'; \delta_2$ with $\delta_1' \in \mathsf{sub}(\delta_1)$ or of the form (ii) δ_2' with $\delta_2' \in \mathsf{sub}(\delta_2)$.*

3. *For $\rho \in \mathsf{sub}(\delta_1|\delta_2)$ it holds that $\rho = \delta_1|\delta_2$ or $\rho \in \mathsf{sub}(\delta_1)$ or $\rho \in \mathsf{sub}(\delta_2)$.*
4. *For $\rho \in \mathsf{sub}(\delta_1\|\delta_2)$ it holds that either $\rho = \delta_1'\|\delta_2'$ with $\delta_1' \in \mathsf{sub}(\delta_1)$ and $\delta_2' \in \mathsf{sub}(\delta_2)$ or $\rho \in \mathsf{sub}(\delta_i)$ with $i \in \{1,2\}$.*

Proof. For any program expression δ over some FO-DS $\mathfrak{D}$ we have

$$\mathsf{sub}(\delta) = \bigcup_{n=0}^{\infty} \mathsf{sub}^n(\delta),$$

where $\mathsf{sub}^n(\delta)$ is the set of all subprograms that are reachable from δ in n steps. The proof is by induction on n.

1. We show that if $\rho \in \mathsf{sub}^n(\delta^*)$, then ρ is of the form $\langle\rangle$, $(\delta)^*$ or $\delta';(\delta)^*$ with $\delta' \in \mathsf{sub}(\delta)$ for all $n \in \mathbb{N}$.

   $n = 0:$ It holds that $\mathsf{sub}^0((\delta)^*) = \{(\delta)^*\}$.

   $n = 1:$ Let $\rho \in \mathsf{sub}^1((\delta)^*)$. Since $\mathsf{head}((\delta)^*) = \{\epsilon\} \cup \mathsf{head}(\delta)$ by definition of the head function, it either holds that $\rho \in \mathsf{tail}(\epsilon,(\delta)^*)$ or $\rho \in \mathsf{tail}(\mathfrak{a},(\delta)^*)$ with $\mathfrak{a} \in \mathsf{head}(\delta)$. In the first case we have $\rho = \langle\rangle$ and in the second case we have $\rho = \delta';(\delta)^*$ by definition of $\mathsf{tail}(\cdot,\cdot)$.

   $n \to n+1:$ Let $\rho' \in \mathsf{sub}^{n+1}((\delta)^*)$. By definition of the reachable subprograms there exists a program expression $\rho \in \mathsf{sub}^n((\delta)^*)$ such that $\rho' \in \mathsf{sub}^1(\rho)$. The induction hypothesis implies that ρ has one of the following forms: $\langle\rangle$, $(\delta)^*$ or $\delta';(\delta)^*$ with $\delta' \in \mathsf{sub}(\delta)$. If $\rho = \langle\rangle$, then also $\rho' = \langle\rangle$. If $\rho = (\delta)^*$, then also ρ' has the desired form as shown in the base case with $n = 1$.

   Now, assume $\rho = \delta';(\delta)^*$ for some $\delta' \in \mathsf{sub}(\delta)$ and $\rho' \in \mathsf{sub}^1(\delta';(\delta)^*)$. There exists $\mathfrak{a} \in \mathsf{head}(\delta';(\delta)^*)$ such that $\rho' \in \mathsf{tail}(\mathfrak{a},\delta';(\delta)^*)$. The definition of the tail function for sequences of program expression yields the following:

   $$\rho' \in \{\delta'';(\delta)^* \mid \delta'' \in \mathsf{tail}(\mathfrak{a},\delta')\} \cup \{\delta'' \mid \delta'' \in \mathsf{tail}(\mathfrak{a}',(\delta)^*)\}$$

   for some guarded action $\mathfrak{a}'$. Now, assume $\rho' \in \{\delta'';(\delta)^* \mid \delta'' \in \mathsf{tail}(\mathfrak{a},\delta')\}$. By assumption on $\rho = \delta';(\delta)^*$ we know that $\delta' \in \mathsf{sub}(\delta)$. With

   $$\delta'' \in \mathsf{sub}(\delta') \subseteq \mathsf{sub}(\delta)$$

   it follows that ρ' has the form as required in the claim. Next, assume $\rho' \in \{\delta'' \mid \delta'' \in \mathsf{tail}(\mathfrak{a}',(\delta)^*)\}$. Consequently, $\rho' \in \mathsf{sub}^1((\delta)^*)$ and using the proof of the base case with $n = 1$ the claim follows. This finishes the induction step.

2. If $\rho \in \mathsf{sub}^n(\delta_1;\delta_2)$, then ρ is of the form (i) $\delta_1';\delta_2$ with $\delta_1' \in \mathsf{sub}(\delta_1)$ or of the form (ii) δ_2' with $\delta_2' \in \mathsf{sub}(\delta_2)$ for all $n \in \mathbb{N}$.

   $n = 0:$ It is implied by the definition of the subprograms that $\mathsf{sub}^0(\delta_1;\delta_2) = \{\delta_1;\delta_2\}$. $\delta_1;\delta_2$ is of the form (i).

   $n = 1:$ Let $\rho \in \mathsf{sub}^1(\delta_1;\delta_2)$. There exists a guarded action $\mathfrak{a} \in \mathsf{head}(\delta_1;\delta_2)$ such that $\rho \in \mathsf{tail}(\mathfrak{a},\delta_1;\delta_2)$. By definition of $\mathsf{tail}(\mathfrak{a},\delta_1;\delta_2)$, ρ is of the form (i) or (ii).

$n \to n+1:$ Let $\rho \in \mathsf{sub}^{n+1}(\delta_1;\delta_2)$. There exists $\theta \in \mathsf{sub}^n(\delta_1;\delta_2)$ such that $\rho \in \mathsf{sub}^1(\theta)$. By induction, θ is of the form (i) or (ii). Assume $\theta = \delta_1';\delta_2$ for some $\delta_1' \in \mathsf{sub}(\delta_1)$. Since $\rho \in \mathsf{tail}(\mathfrak{a},\delta_1';\delta_2)$ for a guarded action $\mathfrak{a} \in \mathsf{head}(\delta_1';\delta_2)$, it is implied that either $\rho = \delta_1'';\delta_2$ with $\delta_1'' \in \mathsf{tail}(\mathfrak{a},\delta_1')$ or there exists $\mathfrak{a}'$ such that $\rho = \delta_2'$ with $\delta_2' \in \mathsf{tail}(\mathfrak{a}',\delta_2)$. In the fist case ρ has the form (i), because it follows that $\delta_1'' \in \mathsf{sub}(\delta_1)$. In the latter case ρ has the form (ii). Now assume $\theta = \delta_2'$ with $\delta_2' \in \mathsf{sub}(\delta_2)$. In this case ρ has the form (ii).

3. If $\rho \in \mathsf{sub}^n(\delta_1|\delta_2)$, then $\rho = \delta_1|\delta_2$ or $\rho \in \mathsf{sub}(\delta_1)$ or $\rho \in \mathsf{sub}(\delta_2)$ for all $n \in \mathbb{N}$.

   $n = 0:$ We have $\mathsf{sub}^0(\delta_1|\delta_2) = \{\delta_1|\delta_2\}$.

   $n = 1:$ Let $\rho \in \mathsf{sub}^1(\delta_1|\delta_2)$. There exists a guarded action $\mathfrak{a} \in \mathsf{head}(\delta_1|\delta_2) = \mathsf{head}(\delta_1) \cup \mathsf{head}(\delta_2)$ such that $\rho \in \mathsf{tail}(\mathfrak{a},\delta_1|\delta_2) = \mathsf{tail}(\mathfrak{a},\delta_1) \cup \mathsf{tail}(\mathfrak{a},\delta_2)$. Thus, $\rho \in \mathsf{sub}(\delta_1) \cup \mathsf{sub}(\delta_2)$.

   $n \to n+1:$ Let $\rho \in \mathsf{sub}^{n+1}(\delta_1|\delta_2)$ with $n \geq 1$. There exists $\theta \in \mathsf{sub}^n(\delta_1|\delta_2)$ and $\mathfrak{a} \in \mathsf{head}(\theta)$ with $\rho \in \mathsf{tail}(\mathfrak{a},\theta)$. Using the induction hypothesis we obtain $\rho \in \mathsf{sub}^1(\theta) \subseteq \mathsf{sub}(\delta_i)$ with $i \in \{1,2\}$.

4. If $\rho \in \mathsf{sub}^n(\delta_1\|\delta_2)$, then either $\rho = \delta_1'\|\delta_2'$ with $\delta_1' \in \mathsf{sub}(\delta_1)$ and $\delta_2' \in \mathsf{sub}(\delta_2)$ or $\rho \in \mathsf{sub}(\delta_i)$ with $i \in \{1,2\}$ for all $n \in \mathbb{N}$.

   $n = 1:$ Let $\rho \in \mathsf{sub}^1(\delta_1\|\delta_2)$. There exists a guarded action $\mathfrak{a} \in \mathsf{head}(\delta_1\|\delta_2)$ such that $\rho \in \mathsf{tail}(\mathfrak{a},\delta_1\|\delta_2)$. By definition of $\mathsf{tail}(\mathfrak{a},\delta_1\|\delta_2)$ we have that $\rho = \delta_1'\|\delta_2$ with $\delta_1' \in \mathsf{sub}(\delta_1)$ or $\rho = \delta_1\|\delta_2'$ with $\delta_2' \in \mathsf{sub}(\delta_2)$ or $\rho = \delta_i'$ with $\delta_i' \in \mathsf{sub}(\delta_i)$. In the definition of the tails, we have omitted the empty program. Therefore in the latter case the expression δ_i' can be written as $\delta_i'\|\langle\rangle$ or as $\langle\rangle\|\delta_i'$.

   $n \to n+1:$ Let $\rho \in \mathsf{sub}^{n+1}(\delta_1\|\delta_2)$. There exists $\theta \in \mathsf{sub}^n(\delta_1\|\delta_2)$ such that $\rho \in \mathsf{sub}^1(\theta)$. By induction, we have that $\theta = \delta_1'\|\delta_2'$ with $\delta_1' \in \mathsf{sub}(\delta_1)$ and $\delta_2' \in \mathsf{sub}(\delta_2)$ or $\theta \in \mathsf{sub}(\delta_i)$ with $i = 1,2$. Since $\rho \in \mathsf{sub}^1(\theta)$, we have that $\rho = \delta_1''\|\delta_2'$ with $\delta_1'' \in \mathsf{sub}(\delta_1')$ or $\rho = \delta_1'\|\delta_2''$ with $\delta_2'' \in \mathsf{sub}(\delta_2')$ or $\rho = \delta_i''$ with $\delta_i'' \in \mathsf{sub}(\delta_i')$. Obviously, $\delta_i'' \in \mathsf{sub}(\delta_i)$ for $i = 1,2$.

□

Now, we are ready to prove some upper bounds on the number and the length of subprograms.

Theorem 3.15. *Let δ be a program expression over some FO-DS $\mathfrak{D}$.*

1. *It holds that* $|\mathsf{sub}(\delta)| \leq 2^{|\delta|}$.
2. *If the interleaving construct does not occur in δ, then* $|\mathsf{sub}(\delta)| \leq |\delta| + 1$.
3. *For a subprogram expression $\rho \in \mathsf{sub}(\delta)$ it holds that* $|\rho| \leq (\mathsf{h}(\delta)+1) \cdot |\delta|$.

Proof. The following equalities and inequalities are a consequence of Lemma 3.14.

a) $|\mathsf{sub}(\boldsymbol{\alpha})| = 2$, $|\mathsf{sub}(\psi?)| = 2$, $|\mathsf{sub}(\langle\rangle)| = 1$ for some action term $\boldsymbol{\alpha}$ and test $\psi?$;

b) $|\mathsf{sub}((\delta)^*)| \leq |\mathsf{sub}(\delta)| + 1$;

c) $|\mathsf{sub}(\delta_1;\delta_2)| \leq |\mathsf{sub}(\delta_1)| + |\mathsf{sub}(\delta_2)|$;

d) $|\mathsf{sub}(\delta_1|\delta_2)| \leq |\mathsf{sub}(\delta_1)| + |\mathsf{sub}(\delta_2)|$;

e) $|\mathsf{sub}(\delta_1\|\delta_2)| \leq |\mathsf{sub}(\delta_1)| \cdot |\mathsf{sub}(\delta_2)|$.

Note that for instance the program expression ρ_1 for some $\rho_1 \in \mathsf{sub}(\delta_1)$ and $\rho_1\|\langle\rangle$ denote the same reachable subprogram of $\delta_1\|\delta_2$. Therefore we obtain the inequality e).

1. We prove
$$|\mathsf{sub}(\delta)| \leq 2^{|\delta|}$$
by induction on the structure of δ using a)-e). The inequality is satisfied if δ is an action term, a test or the empty program. Assume δ is of the form ρ^* and $|\mathsf{sub}(\rho)| \leq 2^{|\rho|}$. With b) and the induction hypothesis it follows that
$$|\mathsf{sub}(\rho^*)| \leq |\mathsf{sub}(\rho)| + 1 \leq 2^{|\rho|} + 1 < 2^{|\rho|+1} = 2^{|\rho^*|}.$$
Assume δ is of the form $\rho_1 \star \rho_2$ with $\star \in \{;,|\}$ and $|\mathsf{sub}(\rho_i)| \leq 2^{|\rho_i|}$, $i = 1,2$. With c) and d) and the induction hypothesis we obtain
$$|\mathsf{sub}(\rho_1 \star \rho_2)| \leq |\mathsf{sub}(\rho_1)| + |\mathsf{sub}(\rho_2)| \leq 2^{|\rho_1|} + 2^{|\rho_2|} < 2^{|\rho_1|+|\rho_2|+1} = 2^{|(\rho_1 \star \rho_2)|}.$$
Assume δ is of the form $\rho_1 \parallel \rho_2$ using d) it follows that
$$|\mathsf{sub}(\rho_1 \parallel \rho_2)| \leq 2^{|\rho_1|} \cdot 2^{|\rho_2|} = 2^{|\rho_1|+|\rho_2|} < 2^{|(\rho_1 \| \rho_2)|}.$$

2. Assume δ does not contain the interleaving constructor. We prove
$$|\mathsf{sub}(\delta)| \leq |\delta| + 1.$$
by induction on the structure of δ using a)-d). It is easy to see that the claim is true in case δ is an action term, a test or the empty program. Let δ be of the form ρ^* and assume $|\mathsf{sub}(\rho)| \leq |\rho| + 1$. With b) and the induction hypothesis it follows that
$$|\mathsf{sub}(\rho^*)| \leq |\mathsf{sub}(\rho)| + 1 \leq |\rho| + 1 + 1 = |\rho^*| + 1.$$
Let δ be of the form $\rho_1 \star \rho_2$ with $\star \in \{;,|\}$ and $|\mathsf{sub}(\rho_i)| \leq |\rho_i| + 1$, $i = 1,2$. With c) and d) and the induction hypothesis we obtain
$$|\mathsf{sub}(\rho_1 \star \rho_2)| \leq |\mathsf{sub}(\rho_1)| + |\mathsf{sub}(\rho_2)| \leq |\rho_1| + 1 + |\rho_2| + 1 = |\rho_1 \star \rho_2| + 1.$$

3. Let $\rho \in \mathsf{sub}(\delta)$. By induction on the structure of δ we prove that
$$|\rho| \leq (\mathsf{h}(\delta) + 1) \cdot |\delta|.$$
The claim is trivial in case δ is an action term, a test or the empty program.

$\delta = \xi^*$: Let δ be of the form ξ^* and $\rho \in \mathsf{sub}(\xi^*)$. According to Lemma 3.14 we have that ρ is of the form $\langle\rangle$ or ξ^* or $\xi';(\xi)^*$ for some $\xi' \in \mathsf{sub}(\xi)$. The claim is obviously true for $\langle\rangle$ and ξ^*. Assume ρ is of the form $\xi'; (\xi)^*$ for some $\xi' \in \mathsf{sub}(\xi)$. By induction it holds that $|\xi'| \le (\mathsf{h}(\xi)+1)\cdot|\xi|$. It holds that

$$\begin{aligned}
& |\xi'; (\xi)^*| \\
= \;& |\xi'| + |\xi^*| + 1 \\
\le \;& (\mathsf{h}(\xi)+1)\cdot|\xi| + |\xi^*| + 1 && \text{(by induction)} \\
= \;& \mathsf{h}(\xi^*)\cdot(|\xi^*|-1) + |\xi^*| + 1 && \text{(with } \mathsf{h}(\xi) = \mathsf{h}(\xi^*)-1 \text{ and } |\xi| = |\xi^*|-1) \\
= \;& (\mathsf{h}(\xi^*)+1)\cdot|\xi^*| - \mathsf{h}(\xi^*) + 1 \\
= \;& (\mathsf{h}(\xi^*)+1)\cdot|\xi^*| - \mathsf{h}(\xi) && \text{(with } \mathsf{h}(\xi^*) = \mathsf{h}(\xi)+1) \\
< \;& (\mathsf{h}(\xi^*)+1)\cdot|\xi^*|.
\end{aligned}$$

$\delta = \xi_1; \xi_2$: Let δ be of the form $\xi_1; \xi_2$ and $\rho \in \mathsf{sub}(\xi_1; \xi_2)$. According to Lemma 3.14 either $\rho = \xi_1'; \xi_2$ for some $\xi_1' \in \mathsf{sub}(\xi_1)$ or $\rho = \xi_2'$ for some $\xi_2' \in \mathsf{sub}(\xi_2)$. In the latter case the claim follows directly from the induction hypothesis. Assume $\rho = \xi_1'; \xi_2$ for some $\xi_1' \in \mathsf{sub}(\xi_1)$. It holds that

$$\begin{aligned}
& |\xi_1'; \xi_2| = |\xi_1'| + |\xi_2| + 1 \\
\le \;& (\mathsf{h}(\xi_1)+1)\cdot|\xi_1| + |\xi_2| + 1 && \text{(by induction)} \\
= \;& \mathsf{h}(\xi_1)\cdot|\xi_1| + |\xi_1; \xi_2| \\
< \;& (\mathsf{h}(\xi_1; \xi_2)+1)\cdot|\xi_1; \xi_2|.
\end{aligned}$$

$\delta = \xi_1 \| \xi_2$: Let δ be of the form $\xi_1 \| \xi_2$ and $\rho \in \mathsf{sub}(\xi_1 \| \xi_2)$. According to Lemma 3.14 either $\rho = \xi_1' \| \xi_2'$ with $\xi_1' \in \mathsf{sub}(\xi_1)$ and $\xi_2' \in \mathsf{sub}(\xi_2)$ or $\rho \in \mathsf{sub}(\xi_i)$ for some $i \in \{1, 2\}$. In the latter case the claim directly follows from the induction hypothesis. Assume $\rho = \xi_1' \| \xi_2'$ with $\xi_1' \in \mathsf{sub}(\xi_1)$ and $\xi_2' \in \mathsf{sub}(\xi_2)$. It holds that

$$\begin{aligned}
& |(\xi_1' \| \xi_2')| = |\xi_1'| + |\xi_2'| + 1 \\
\le \;& (\mathsf{h}(\xi_1)+1)\cdot|\xi_1| + (\mathsf{h}(\xi_2)+1)\cdot|\xi_2| + 1 && \text{(by induction)} \\
= \;& \mathsf{h}(\xi_1)\cdot|\xi_1| + \mathsf{h}(\xi_2)\cdot|\xi_2| + |(\xi_1 \| \xi_2)| \\
\le \;& \mathsf{max}(\mathsf{h}(\xi_1), \mathsf{h}(\xi_2))\cdot(|\xi_1| + |\xi_2|) + |(\xi_1 \| \xi_2)| \\
< \;& (\mathsf{h}(\xi_1 \| \xi_2)+1)\cdot|(\xi_1 \| \xi_2)|.
\end{aligned}$$

$\delta = \xi_1 | \xi_2$: Let δ be of the form $\xi_1 | \xi_2$ and $\rho \in \mathsf{sub}(\xi_1 | \xi_2)$. According to Lemma 3.14 either $\rho \in \mathsf{sub}(\xi_1)$ or $\rho \in \mathsf{sub}(\xi_2)$. The claim follows directly from the induction hypothesis.

□

Note that, in the presence of the interleaving operator, the exponential bound is actually reached. Consider the program expression

$$\delta = \alpha_1 \,\|\, (\alpha_2 \,\|\, \cdots (\alpha_{n-1} \,\|\, \alpha_n) \cdots).$$

We claim that $\mathsf{sub}(\delta)$ contains at least 2^n many reachable subprograms. In fact, it is easy to see that for every subset $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$ with $i_1 \le \cdots \le i_k$ the expression

$$\alpha_{i_1} \,\|\, (\alpha_{i_2} \,\|\, \cdots (\alpha_{i_{k-1}} \,\|\, \alpha_{i_k}) \cdots)$$

is a reachable subprogram of δ.

3.4 Abstract Transition Systems and Bisimulations

In this section, the notion of a propositional *bisimilar abstraction of a pick-free program over ground actions* is defined. The transition system of a ConGolog program is an infinite first-order transition system. Ideally, we would like to obtain a finite propositional abstraction that preserves all temporal properties of the original infinite transition system and can be used as an input for a model checking procedure in order to decide the verification problem. For this purpose, the abstraction has to exactly mimic the behavior of the concrete system. To capture this formally we adapt the standard notions of *simulation* and *bisimulation* relations [Mil71] to our setting. Intuitively, bisimulation relations are used to characterize behavioral equivalence of transition systems.

First, the class of ConGolog programs over a set of $\mathcal{L}$-definable ground actions, where $\mathcal{L}$ is some fragment of FO, is defined.

Definition 3.16. Let $\mathcal{L}$ be some fragment of FO and let $\mathcal{P} = (\mathfrak{D}, \delta)$ be ConGolog program over an FO-DS $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$, where δ is a pick-free program expression over ground actions. $\mathcal{P}$ is called a pick-free *$\mathcal{L}$-ConGolog program* iff the following conditions are satisfied

- the set of all ground actions occurring in δ is $\mathcal{L}$-definable in $\mathfrak{D}$, and
- the tests occurring in δ are formulated in $\mathcal{L}$.

Let $\mathcal{C}$ be an $\mathcal{L}$-context over $\mathcal{F}$. We say that *$\mathcal{C}$ is a proper context for $\mathcal{P}$* iff the following conditions are satisfied

- the set of all ground actions in δ are $\mathcal{L}$-definable in $\mathfrak{D}$ w.r.t. $\mathcal{C}$ and
- for all test ψ? occurring in δ we have $\psi \in \mathcal{C}$.

The subclass of programs, where the base logic $\mathcal{L}$ is restricted to a DL, is called *DL-ConGolog*. ▲

In the remainder of this section $\mathcal{L}$ denotes an arbitrary fragment of FO.

For an $\mathcal{L}$-ConGolog program a proper context always exists. A context can always be extended with additional sentences. The following lemma directly follows from the definitions.

Lemma 3.17. *Let $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ be an FO-DS, $\boldsymbol{A} \subseteq \mathsf{ground}(\mathsf{Act})$ a finite set of ground actions, and $\mathcal{C}$ and $\mathcal{C}'$ $\mathcal{L}$-contexts over $\mathcal{F}$ with $\mathcal{C} \subseteq \mathcal{C}'$.*

It holds that if $\boldsymbol{A}$ is $\mathcal{L}$-definable in $\mathfrak{D}$ w.r.t. $\mathcal{C}$, then $\boldsymbol{A}$ is also $\mathcal{L}$-definable in $\mathfrak{D}$ w.r.t. $\mathcal{C}'$.

An $\mathcal{L}$-context that is part of an $\mathcal{L}$-admissible representation can always be extended with the $\mathcal{L}$-tests occurring in a program expression or the $\mathcal{L}$-axioms in a temporal specification.

Since a proper context for an $\mathcal{L}$-ConGolog program contains all the tests in the program, executability of a guarded action is fully determined by the static type of the interpretation that represents the current state.

Lemma 3.18. *Let $\mathcal{P} = (\mathfrak{D}, \delta)$ be a pick-free $\mathcal{L}$-ConGolog program with the FO-DS $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ and let $\mathcal{C}$ be a proper $\mathcal{L}$-context for $\mathcal{P}$. For two interpretations $\mathcal{I}, \mathcal{J} \in \mathbb{I}$ with $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J})$ and a reachable subprogram $\rho \in \mathsf{sub}(\delta)$ it holds that $\mathfrak{a}$ is executable in $\mathcal{I}$ iff $\mathfrak{a}$ is executable in $\mathcal{J}$ for any $\mathfrak{a} \in \mathsf{head}(\rho)$.*

Proof. Let $\rho \in \mathsf{sub}(\delta)$ and $\mathfrak{a} = \psi_1?; \cdots ; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$. The definition of the head function ensures that the tests $\psi_1?, \ldots, \psi_n?$ occur in δ and $\psi_1, \ldots, \psi_n$ are therefore contained in $\mathcal{C}$. The assumption $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J})$ for two interpretations $\mathcal{I}, \mathcal{J} \in \mathbb{I}$ implies that $\mathcal{I} \models \psi_i$ iff $\mathcal{J} \models \psi_i$ for all $i = 1, \ldots, n$. Let $\boldsymbol{A}$ be the set of all ground actions occurring in δ. Since $\mathcal{P}$ is a pick-free $\mathcal{L}$-ConGolog program and $\mathcal{C}$ a proper context for $\mathcal{P}$, there exists an $\mathcal{L}$-admissible representation $\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ of $\boldsymbol{A}$ with $\mathfrak{D}|_{\boldsymbol{A}} = \mathfrak{D}(\Sigma_{\boldsymbol{A}})$. Let $\mathsf{Pre}_{\boldsymbol{A}}$ be the precondition relation of $\mathfrak{D}(\Sigma_{\boldsymbol{A}})$. It coincides with the relation in $\mathfrak{D}$ for the actions in $\boldsymbol{A}$. By definition we have $(\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{Y}), \boldsymbol{\beta}) \in \mathsf{Pre}_{\mathcal{C}}$ iff $(\mathcal{Y}, \boldsymbol{\beta}) \in \mathsf{Pre}_{\boldsymbol{A}}$ for all $\mathcal{Y} \in \mathbb{I}$ and all $\boldsymbol{\beta} \in \boldsymbol{A}$.

Therefore, $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J})$ implies $(\mathcal{I}, \boldsymbol{\alpha}) \in \mathsf{Pre}$ iff $(\mathcal{J}, \boldsymbol{\alpha}) \in \mathsf{Pre}$. Consequently, $\mathfrak{a}$ is executable in $\mathcal{I}$ iff it is executable in $\mathcal{J}$. □

Executability of guarded actions can be defined on the level of static types. Let $\mathcal{C}$ be an $\mathcal{L}$-context, $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ a static type and ψ a Boolean combination of formulas from $\mathcal{C}$. $\mathfrak{s}$ is viewed as a Boolean valuation of the axioms in $\mathcal{C}$. *Satisfaction of ψ in $\mathfrak{s}$*, denoted by $\mathfrak{s} \models_{\mathcal{C}} \psi$, is defined in the obvious way.

Definition 3.19. Let $\mathcal{P} = (\mathfrak{D}(\Sigma_{\boldsymbol{A}}), \delta)$ be a pick-free $\mathcal{L}$-ConGolog program, where $\boldsymbol{A}$ is the set of all ground actions occurring in δ, $\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ an $\mathcal{L}$-admissible representation of $\boldsymbol{A}$ and $\mathcal{C}$ a proper $\mathcal{L}$-context for $\mathcal{P}$. Furthermore, let $\rho \in \mathsf{sub}(\delta)$ a reachable subprogram of δ, $\mathfrak{a} = \psi_1?; \cdots ; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ a guarded action and $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ a static type.

We say that $\mathfrak{a}$ *is executable in* $\mathfrak{s}$ iff $\mathfrak{s} \models_{\mathcal{C}} \psi_1 \wedge \cdots \wedge \psi_n$ and $(\mathfrak{s}, \boldsymbol{\alpha}) \in \mathsf{Pre}_{\mathcal{C}}$. ▲

The next lemma is a direct consequence of Lemma 3.18 and the definition above.

Lemma 3.20. *Let $\mathcal{P} = (\mathfrak{D}(\Sigma_{\boldsymbol{A}}), \delta)$ and $\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ be as in Definition 3.19, $\rho \in \mathsf{sub}(\delta)$ a reachable subprogram of δ, $\mathfrak{a} \in \mathsf{head}(\rho)$ a guarded action and $\mathcal{I}$ an interpretation from the state space of $\mathfrak{D}(\Sigma_{\boldsymbol{A}})$. It holds that $\mathfrak{a}$ is executable in $\mathcal{I}$ iff it is executable in* $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I})$.

We define the notion of a *bisimulation relation* between states of a first-order transition system and of a propositional one based on a context and its static types. A bisimulation requires that transitions preserve static types.

Definition 3.21. Let $\mathcal{C}$ be an $\mathcal{L}$-context, AP a finite set of atomic propositions such that a bijection $\iota_{\mathcal{C}} : \mathcal{C} \rightarrow \mathsf{AP}$ between $\mathcal{C}$ and AP exists, and let $\mathfrak{I} = (Q_{\mathfrak{I}}, I_{\mathfrak{I}}, \hookrightarrow_{\mathfrak{I}}, \lambda_{\mathfrak{I}})$ be a first-order transition system and $\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$ a propositional transition system over AP. A binary relation $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ is called $\mathcal{C}$*-bisimulation* iff the following conditions are satisfied:

- $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ implies $\lambda_{\mathfrak{T}}(q_{\mathfrak{T}}) = \{\iota_{\mathcal{C}}(\psi) \mid \psi \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I})\}$, where $\lambda_{\mathfrak{I}}(q_{\mathfrak{I}}) = \mathcal{I}$.
- If $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and there is a transition $q_{\mathfrak{I}} \hookrightarrow_{\mathfrak{I}} q'_{\mathfrak{I}}$, then there exists a transition $q_{\mathfrak{T}} \hookrightarrow_{\mathfrak{T}} q'_{\mathfrak{T}}$ such that $q'_{\mathfrak{I}} \simeq_{\mathcal{C}} q'_{\mathfrak{T}}$.
- If $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and there is a transition $q_{\mathfrak{T}} \hookrightarrow_{\mathfrak{T}} q'_{\mathfrak{T}}$, then there exists a transition $q_{\mathfrak{I}} \hookrightarrow_{\mathfrak{I}} q'_{\mathfrak{I}}$ such that $q'_{\mathfrak{I}} \simeq_{\mathcal{C}} q'_{\mathfrak{T}}$.

The relation $\simeq_{\mathcal{C}}$ is extended to paths as follows. Let π be a path in $\mathfrak{I}$ and $\mathfrak{p}$ a path in $\mathfrak{T}$. We write $\pi \simeq_{\mathcal{C}} \mathfrak{p}$ iff $\pi[i] \simeq_{\mathcal{C}} \mathfrak{p}[i]$ for all $i \geq 0$.

We say that $\mathfrak{I}$ and $\mathfrak{T}$ are *$\mathcal{C}$-bisimilar* iff there exists a $\mathcal{C}$-bisimulation $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ such that

- for all $q_{\mathfrak{I}} \in I_{\mathfrak{I}}$ there exists $q_{\mathfrak{T}} \in I_{\mathfrak{T}}$ such that $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and
- for all $q_{\mathfrak{T}} \in I_{\mathfrak{T}}$ there exists $q_{\mathfrak{I}} \in I_{\mathfrak{I}}$ such that $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$.

▲

The definition of a $\mathcal{C}$-bisimulation leads to the notion of a *bisimilar propositional abstraction of a program*.

Definition 3.22. Let $\mathcal{P} = (\mathfrak{D}, \delta)$ be an $\mathcal{L}$-ConGolog program, $\mathcal{C}$ a proper $\mathcal{L}$-context for $\mathcal{P}$, AP a finite set of atomic propositions and $\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$ a propositional transition system over AP.

We say that $\mathfrak{T}$ is a *bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. $\mathcal{C}$* iff the following conditions are satisfied

- There exists a bijection $\iota_{\mathcal{C}} : \mathcal{C} \to \mathsf{AP}$ between $\mathcal{C}$ and AP.
- The transition system $\mathfrak{I}_{\mathcal{P}}$ induced by $\mathcal{P}$ and the propositional transition system $\mathfrak{T}$ are $\mathcal{C}$-bisimilar.

▲

$\mathcal{C}$-bisimilarity ensures that satisfaction of temporal properties over axioms in $\mathcal{C}$ is preserved in the abstract transition system. We first introduce some additional notions needed for the proof. Let $\mathcal{C}$ be an $\mathcal{L}$-context, AP a finite set of atomic proposition, $\iota_{\mathcal{C}}$ a bijection between $\mathcal{C}$ and AP and Ξ an $\mathcal{L}$-CTL* state or path formula such that all axioms in Ξ are contained in $\mathcal{C}$. With $\iota_{\mathcal{C}}(\Xi)$ we denote the propositional CTL* formula over AP that is obtained from Ξ be replacing all occurrences of axioms in Ξ by their image in $\iota_{\mathcal{C}}$.

The following auxiliary lemma states a property of paths starting in two $\mathcal{C}$-bisimilar states.

Lemma 3.23. *Let $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ be a $\mathcal{C}$-bisimulation between states of a first-order transition system $\mathfrak{I}$ and states in a propositional transition system $\mathfrak{T}$ as defined in Definition 3.21.*

For states $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ it holds that for every path $\pi_{\mathfrak{I}} \in \mathsf{paths}(\mathfrak{I}, q_{\mathfrak{I}})$ there exists a path $\pi_{\mathfrak{T}} \in \mathsf{paths}(\mathfrak{T}, q_{\mathfrak{T}})$ such that $\pi_{\mathfrak{I}} \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}$, and vice versa: for every path $\pi_{\mathfrak{T}} \in \mathsf{paths}(\mathfrak{T}, q_{\mathfrak{T}})$ there exists a path $\pi_{\mathfrak{I}} \in \mathsf{paths}(\mathfrak{I}, q_{\mathfrak{I}})$ such that $\pi_{\mathfrak{I}} \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}$.

Now we are ready to prove the lemma about preserving temporal properties in a $\mathcal{C}$-bisimilar propositional abstraction.

Lemma 3.24. *Let $\mathcal{C}$ be an $\mathcal{L}$-context, AP a finite set of atomic propositions such that a bijection $\iota_{\mathcal{C}} : \mathcal{C} \to \mathsf{AP}$ between $\mathcal{C}$ and AP exists, and let $\mathfrak{I} = (Q_{\mathfrak{I}}, I_{\mathfrak{I}}, \hookrightarrow_{\mathfrak{I}}, \lambda_{\mathfrak{I}})$ be a first-order transition system, $\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$ a propositional transition system over AP and $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ a $\mathcal{C}$-bisimulation.*

1. *For an $\mathcal{L}$-CTL* state formula Φ that mentions only axioms from $\mathcal{C}$ it holds that if $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ for two states, then $\mathfrak{I}, q_{\mathfrak{I}} \models \Phi$ iff $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Phi)$.*

2. *For an $\mathcal{L}$-CTL* path formula Ψ that mentions only axioms from $\mathcal{C}$ it holds that if $\pi_{\mathfrak{I}} \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}$ for two paths, then $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi$ iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Psi)$.*

Proof. Let $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ be a $\mathcal{C}$-bisimulation as stated in the claim, and $\iota_{\mathcal{C}}$ the bijection between $\mathcal{C}$ and the set of atomic propositions AP.

For the base case we assume that $\Phi = \varrho$ for some $\mathcal{L}$-axiom ϱ. By assumption on Φ we have that $\varrho \in \mathcal{C}$. For two states $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ it holds that $\mathfrak{I}, q_{\mathfrak{I}} \models \varrho$

iff $\mathcal{I}_{q_{\mathfrak{I}}} \models \varrho$ with $\lambda_{\mathfrak{I}}(q_{\mathfrak{I}}) = \mathcal{I}_{q_{\mathfrak{I}}}$

iff $\varrho \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}_{q_{\mathfrak{I}}})$ since $\varrho \in \mathcal{C}$

iff $\iota_{\mathcal{C}}(\varrho) \in \lambda_{\mathfrak{T}}(q_{\mathfrak{T}})$ since $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ implies $\lambda_{\mathfrak{T}}(q_{\mathfrak{T}}) = \{\iota_{\mathcal{C}}(\varrho') \mid \varrho' \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}_{q_{\mathfrak{I}}})\}$

iff $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\varrho)$.

For the induction step assume that Claim 1 holds for Φ_1, Φ_2 and Claim 2 for Ψ.

$\Phi = \neg\Phi_1$: It holds that $\mathfrak{I}, q_{\mathfrak{I}} \models \neg\Phi_1$

iff $\mathfrak{I}, q_{\mathfrak{I}} \not\models \Phi_1$

iff $\mathfrak{T}, q_{\mathfrak{T}} \not\models \iota_{\mathcal{C}}(\Phi_1)$ (using the induction hypothesis)

iff $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\neg\Phi_1)$.

$\Phi = \Phi_1 \wedge \Phi_2$: It holds that $\mathfrak{I}, q_{\mathfrak{I}} \models \Phi_1 \wedge \Phi_2$

iff $\mathfrak{I}, q_{\mathfrak{I}} \models \Phi_1$ and $\mathfrak{I}, q_{\mathfrak{I}} \models \Phi_2$

iff $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Phi_1)$ and $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Phi_2)$ (using the induction hypothesis)

iff $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Phi_1 \wedge \Phi_2)$.

$\Phi = \mathsf{E}\Psi$: It holds that $\mathfrak{I}, q_{\mathfrak{I}} \models \mathsf{E}\Psi$

iff there exists a path $\pi_{\mathfrak{I}} \in \mathsf{paths}(\mathfrak{I}, q_{\mathfrak{I}})$ such that $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi$

iff there exists a path $\pi_{\mathfrak{I}} \in \mathsf{paths}(\mathfrak{I}, q_{\mathfrak{I}})$ such that $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi$ and there exists a path $\pi_{\mathfrak{T}} \in \mathsf{paths}(\mathfrak{T}, q_{\mathfrak{T}})$ such that $\pi_{\mathfrak{I}} \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}$ by assumption $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and Lemma 3.23

iff there exists $\pi_{\mathfrak{T}} \in \mathsf{paths}(\mathfrak{T}, q_{\mathfrak{T}})$ such that $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Psi)$ (using the induction hypothesis)

iff $\mathfrak{T}, q_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\mathsf{E}\Psi)$.

Assume Claim 1 holds for Φ and Claim 2 for Ψ_1 and Ψ_2. Let $\pi_{\mathfrak{I}} \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}$.

$\Psi = \Phi$: We have $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Phi$

iff $\mathfrak{I}, \pi_{\mathfrak{I}}[0] \models \Phi$

iff $\mathfrak{T}, \pi_{\mathfrak{T}}[0] \models \iota_{\mathcal{C}}(\Phi)$ (using $\pi_{\mathfrak{I}} \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}$ implies $\pi_{\mathfrak{I}}[0] \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}[0]$ and the induction hypothesis)

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Phi)$.

$\Psi = \neg\Psi_1$: It holds that $\mathfrak{I}, \pi_{\mathfrak{I}} \models \neg\Psi_1$

iff $\mathfrak{I}, \pi_{\mathfrak{I}} \not\models \Psi_1$

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \not\models \iota_{\mathcal{C}}(\Psi_1)$ (using the induction hypothesis)

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\neg\Psi_1)$.

$\Phi = \Phi_1 \wedge \Phi_2$: It holds that $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi_1 \wedge \Psi_2$

iff $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi_1$ and $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi_2$

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Psi_1)$ and $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Psi_2)$ (using the induction hypothesis)

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Psi_1 \wedge \Psi_2)$.

$\Psi = \mathsf{X}\Psi_1$: It holds that $\mathfrak{I}, \pi_{\mathfrak{I}} \models \mathsf{X}\Psi_1$

iff $\mathfrak{I}, \pi_{\mathfrak{I}}[1..] \models \Psi_1$

iff $\mathfrak{T}, \pi_{\mathfrak{T}}[1..] \models \iota_{\mathcal{C}}(\Psi_1)$ (with $\pi_{\mathfrak{I}}[1..] \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}[1..]$ and the induction hypothesis)

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\mathsf{X}\Psi_1)$.

$\Psi = \Psi_1 \,\mathsf{U}\, \Psi_2$: It holds that $\mathfrak{I}, \pi_{\mathfrak{I}} \models \Psi_1 \,\mathsf{U}\, \Psi_2$

iff there exists a k such that $\mathfrak{I}, \pi_{\mathfrak{I}}[k..] \models \Psi_2$ and $\mathfrak{I}, \pi_{\mathfrak{I}}[j..] \models \Psi_1$ holds for all $j = 0, \ldots, k-1$

iff there exists a k such that $\mathfrak{T}, \pi_{\mathfrak{T}}[k..] \models \iota_{\mathcal{C}}(\Psi_2)$ and $\mathfrak{T}, \pi_{\mathfrak{T}}[j..] \models \iota_{\mathcal{C}}(\Psi_1)$ for all $j = 0, \ldots, k-1$ (with $\pi_{\mathfrak{I}}[i..] \simeq_{\mathcal{C}} \pi_{\mathfrak{T}}[i..]$ for all $i = 0, 1, 2, \ldots$ and the induction hypothesis)

iff $\mathfrak{T}, \pi_{\mathfrak{T}} \models \iota_{\mathcal{C}}(\Psi_1 \,\mathsf{U}\, \Psi_2)$.

□

It follows from this lemma that *if* we can effectively compute a *finite* bisimilar propositional abstraction of a pick-free $\mathcal{L}$-ConGolog program w.r.t. a proper context, *then* the verification problem boils down to a decidable propositional model checking problem.

Chapter 4

Verifying Pick-Free Programs over Local-Effect Actions

In this chapter, the class of pick-free DL-ConGolog programs is considered. We investigate the computational properties of the verification problem for the restricted case in which all actions only have *local effects*. An action has only local effects if only the named objects mentioned in the respective action term are affected by its execution. In the literature on reasoning about actions the restriction to only local-effect actions is quite common. For instance, in [VLL08] it has be shown that progression in a Situation Calculus action theory is first-order definable for local-effect actions. Note that also in the DL-based action formalism introduced in [Baa+05a] only local-effect actions can be defined.

We first choose this simple class of programs to introduce our main abstraction technique. Our general approach to prove decidability is based on a construction of a finite bisimilar propositional abstraction that allows a reduction to propositional model checking. Furthermore, we make use of previous work on reasoning about local-effect actions in a DL-based setting and reuse some techniques that were first developed in [Baa+05a] for reducing the projection problem to a standard DL reasoning problem. We also investigate the complexity of the verification problem for pick-free programs based on local effect DL-action theories (see Definition 2.29).

4.1 Local-Effect Actions

Local-effect actions are defined as a syntactical restriction.

Definition 4.1. Let $\mathcal{L}$ be a syntactical fragment of FO, $\boldsymbol{A}$ a finite set of ground action terms, and $\Sigma_{\boldsymbol{A}} = (KB, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ an $\mathcal{L}$-admissible representation of $\boldsymbol{A}$. We say that $\Sigma_{\boldsymbol{A}}$ is an $\mathcal{L}$-admissible *local-effect* representation of $\boldsymbol{A}$ iff for all

$$(\mathfrak{s}, \boldsymbol{\alpha}, F) \in (\mathfrak{S}_{\mathcal{C}} \times \boldsymbol{A} \times \mathcal{F}) \text{ with } \mathsf{ar}(F) = n \text{ for some } n > 0$$

it holds that the formulas $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$ and $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$ are disjunctions of formulas of the form

$$x_1 \approx o_1 \wedge \cdots \wedge x_n \approx o_n \tag{4.1}$$

where the object names $o_1, \ldots, o_n$ are mentioned as arguments of the action term $\boldsymbol{\alpha}$. Thus, each formula $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$ and $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$ enumerates a finite set of object tuples.

In the following, we use $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ to denote a $\mathcal{L}$-admissible local-effect representation of $\boldsymbol{A}$. For a tuple of object names $\bar{o}$ we write

$$\bar{o} \in \mathsf{L}^{+}[\mathfrak{s}, \boldsymbol{\alpha}, F] \text{ or } \bar{o} \in \mathsf{L}^{-}[\mathfrak{s}, \boldsymbol{\alpha}, F]$$

to denote that $\bar{x} \approx \bar{o}$ (abbreviation of 4.1) is a disjunct in $\mathsf{L}^{+}[\mathfrak{s}, \boldsymbol{\alpha}, F]$ or $\mathsf{L}^{-}[\mathfrak{s}, \boldsymbol{\alpha}, F]$, respectively. ▲

If $\mathcal{L}$ is a DL, then the formula (4.1) can be written as a nominal $\{o\}$ in case F is a concept name or as a nominal role $\{(o, o')\}$ if F is a role name.

Notice that a local-effect $\mathcal{L}$-action theory of the form $\Sigma = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff})$ (Definition 2.29), where $\boldsymbol{A}$ is a finite set of ground action terms, can be viewed as an $\mathcal{L}$-admissible local-effect representation of $\boldsymbol{A}$ for some DL $\mathcal{L}$.

In this chapter, we consider *pick-free DL-ConGolog programs over local-effect actions* and the following instance of the verification problem.

Definition 4.2. Let $\mathcal{L} \subseteq \mathcal{DL}$ be a DL. A *pick-free $\mathcal{L}$-ConGolog program over local effect actions* consists of the following components:

- a finite set of relevant ground actions terms $\boldsymbol{A}$,
- an $\mathcal{L}$-admissible local effect representation $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^{+}, \mathsf{L}^{-}, \mathsf{Pre}_{\mathcal{C}})$ of $\boldsymbol{A}$, and
- a pick-free program expression δ, where all action terms mentioned in δ are contained in $\boldsymbol{A}$ and for all tests $\psi?$ occurring in δ the formula ψ is a Boolean $\mathcal{L}$-KB contained in the relevant context $\mathcal{C}$.

Let $\boldsymbol{A}$, $\mathsf{Loc}_{\boldsymbol{A}}$ and δ be as described above and Φ an $\mathcal{L}$-CTL* state formula over axioms contained in $\mathcal{C}$. The *verification problem* asks whether Φ is valid in $\mathcal{P} = (\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}), \delta)$. ▲

In the remainder of this chapter $\mathcal{L}$ denotes an arbitrary DL that is a sublogic of $\mathcal{DL}$. In this chapter we do *not make the UNA* but we assume that different object names mentioned in the input are also interpreted differently.

Note that the program given in Example 2.48 is almost a pick-free $\mathcal{ALCIO}$-ConGolog program over local-effect actions. The only action with non-local effects is $\mathtt{discharged}(\mathsf{bat})$. There can be an unbounded number of unknown devices connected to bat that are affected in case bat gets fully discharged. A local-effect version of the action can be obtained by adding the following ABox assertion to the description of the initial situation:

$$\mathsf{bat} \sqsubseteq \forall \mathsf{inv}(\mathit{ConTo}).\{\mathsf{dev}\}.$$

It expresses that only the object dev is connected to bat. Fully discharging bat now only affects the two named objects bat and dev. The set of all effects of $\mathtt{discharged}(\mathsf{bat}, \mathsf{dev})$ can be redefined as follows:

$$\begin{aligned}\mathsf{eff}(\mathtt{discharged}(\mathsf{bat}, \mathsf{dev})) := \{\, &\langle \mathit{PowerS}, \{\mathsf{bat}\}\rangle^{-}, \\ &(\mathsf{dev} \sqsubseteq \forall \mathit{ConTo}.\{\mathsf{bat}\}) \rhd \langle \mathit{On}, \{\mathsf{dev}\}\rangle^{-}\}.\end{aligned} \tag{4.2}$$

4.2 Dynamic Types and Local Effects

In this subsection, we introduce our abstraction technique for pick-free DL-ConGolog programs over local-effect actions.

Even in case of only local-effects the induced transition system is infinite. There are infinitely many initial states since the initial knowledge base has infinitely many models. To deal with this problems we are looking for an equivalence relation on the relevant state space which allows us to construct a bisimilar abstraction.

Let $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ be an $\mathcal{L}$-admissible local-effect representation of $\boldsymbol{A}$ and

$$\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre})$$

the induced FO-DS. A proper equivalence relation on the state space

$$\sim \subseteq \mathbb{I} \times \mathbb{I}$$

should have the following properties:

(D1) If $\mathcal{I} \sim \mathcal{J}$, then $\text{s-type}_{\mathcal{C}}(\mathcal{I}) = \text{s-type}_{\mathcal{C}}(\mathcal{J})$.

(D2) If $\mathcal{I} \sim \mathcal{J}$, then for all $\boldsymbol{\alpha} \in \boldsymbol{A}$ and all $\mathcal{I}'$ and $\mathcal{J}'$ with $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})} \mathcal{I}'$ and $\mathcal{J} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})} \mathcal{J}'$ it also holds that $\mathcal{I}' \sim \mathcal{J}'$.

These two conditions can be viewed as a reformulation of the definition of $\mathcal{C}$-bisimulations in Definition 3.21 that guarantee behavioral equivalence. First, the goal is to obtain such a relation as an equivalence relation on the state space with finitely many equivalence classes. And second, appropriate finite representations of these equivalence classes should be effectively computable.

A first idea to construct an equivalence relation is to choose a "large enough" context $\mathcal{C}$ such that the relation that is defined by

$$\text{s-type}_{\mathcal{C}}(\mathcal{I}) = \text{s-type}_{\mathcal{C}}(\mathcal{J})$$

also guarantees property (D2). As the next example shows it is at least not obvious how to define such a context.

Example 4.3. As a running example in this chapter we define a program using the concept names *PowerS* and *Dev*, the role name *ConTo*, the object names dev and bat and the ground action term toggle(dev, bat). The initial situation is described by the ABox

$$\mathcal{A} := \{\mathsf{dev} \sqsubseteq \mathit{Dev}, \quad \mathsf{bat} \sqsubseteq \mathit{PowerS}, \quad (\mathsf{dev}, \mathsf{bat}) \sqsubseteq \mathit{ConTo}\}.$$

The action toggle(dev, bat) toggles the connection between dev and bat and has the following preconditions and effects

$$\begin{aligned}
&\mathsf{pre}(\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})) := \emptyset,\\
&\mathsf{eff}(\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})) := \{\neg((\mathsf{dev}, \mathsf{bat}) \sqsubseteq \mathit{ConTo}) \triangleright \langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^+,\\
&\qquad\qquad ((\mathsf{dev}, \mathsf{bat}) \sqsubseteq \mathit{ConTo}) \triangleright \langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^-\}.
\end{aligned}$$

The local-effect action theory is given by

$$\Sigma := (\mathcal{K} := (\mathcal{T} := \emptyset, \mathcal{A}), \boldsymbol{A} := \{\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})\}, \mathsf{pre}, \mathsf{eff}).$$

A program expression and a temporal property are given by

$$\delta := \big(\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})\big)^* \text{ and } \Phi := \mathsf{EG}\,(\mathsf{dev} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}).$$

Thus, the program executes $\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})$ zero or more times and the property describes a state with a path where dev is always connected to some power supply. One can identify the following two relevant axioms in this domain:

$$\varphi_{\mathsf{con}} := ((\mathsf{dev}, \mathsf{bat}) \sqsubseteq \mathit{ConTo}) \text{ and } \varphi_{\exists} := (\mathsf{dev} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}).$$

As a proper context $\mathcal{C}$ for the program we choose

$$\mathcal{C} := \{\varphi_{\mathsf{con}}, \varphi_{\exists}, \neg\varphi_{\mathsf{con}}, \neg\varphi_{\exists}\}.$$

In this context, we simply ignore the initial facts $(\mathsf{dev} \sqsubseteq \mathit{Dev})$ and $(\mathsf{bat} \sqsubseteq \mathit{PowerS})$ because the concept names *Dev* and *PowerS* are not affected by actions in this example. Two interpretation $\mathcal{I}_1, \mathcal{I}_2 \in \mathbb{I}_{\mathfrak{D}(\Sigma)}$ are defined over the set of standard names

$$\mathsf{N_O} := \{\mathsf{dev}, \mathsf{dev}_1, \mathsf{dev}_2, \ldots\} \cup \{\mathsf{bat}, \mathsf{bat}_1, \mathsf{bat}_2, \ldots\}$$

such that for all $i \in \{1, 2\}$ we have

$$\begin{aligned} \mathit{PowerS}^{\mathcal{I}_i} &:= \{\mathsf{bat}, \mathsf{bat}_1, \mathsf{bat}_2, \ldots\}; \\ \mathit{Dev}^{\mathcal{I}_i} &:= \{\mathsf{dev}, \mathsf{dev}_1, \mathsf{dev}_2, \ldots\}; \\ \mathit{ConTo}^{\mathcal{I}_1} &:= \{(\mathsf{dev}, \mathsf{bat})\} \text{ and} \\ \mathit{ConTo}^{\mathcal{I}_2} &:= \{(\mathsf{dev}, \mathsf{bat}), (\mathsf{dev}, \mathsf{bat}_1)\}. \end{aligned}$$

$\mathcal{I}_1$ and $\mathcal{I}_2$ are identical except for the interpretation of the role name *ConTo*. In $\mathcal{I}_2$ the object dev has bat_1 as an additional connected power supply. Let $\mathcal{J}_1$ and $\mathcal{J}_2$ be the interpretations with $\mathcal{I}_1 \Rightarrow^{\alpha}_{\mathfrak{D}(\Sigma)} \mathcal{J}_1$ and $\mathcal{I}_2 \Rightarrow^{\alpha}_{\mathfrak{D}(\Sigma)} \mathcal{J}_2$ and $\alpha = \mathtt{toggle}(\mathsf{dev}, \mathsf{bat})$. We have

$$\text{s-type}_{\mathcal{C}}(\mathcal{I}_1) = \text{s-type}_{\mathcal{C}}(\mathcal{I}_2) \text{ and } \text{s-type}_{\mathcal{C}}(\mathcal{J}_1) \neq \text{s-type}_{\mathcal{C}}(\mathcal{J}_2)$$

because $\neg\varphi_{\exists} \in \text{s-type}_{\mathcal{C}}(\mathcal{J}_1)$ but $\varphi_{\exists} \in \text{s-type}_{\mathcal{C}}(\mathcal{J}_2)$. Thus, equal static types do not necessarily guarantee that the observable behavior is also equal. With the chosen context $\mathcal{C}$, $\mathcal{I}_1$ and $\mathcal{I}_2$ cannot be distinguished. ▲

Intuitively, the static type does not necessarily take into account how the interpretation evolves. The context may contain complex concept with quantification over the whole domain. It becomes apparent that also domain elements of interpretations, that are *not* referred to by any of the objects mentioned in the actions, are relevant for the outcome of an action. To deal with this problem we are going to introduce *dynamic types* as an appropriate extension of static types. To distinguish different interpretations by means of dynamic types we directly take all the effects of ground action sequences into account.

Some auxiliary notions are defined first.

Definition 4.4. Let $\boldsymbol{A}$ be a finite set of ground actions and $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ an $\mathcal{L}$-admissible local effect representation of $\boldsymbol{A}$. For a static type $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ and action $\boldsymbol{\alpha} \in \boldsymbol{A}$ we define a *set of unconditional local effect descriptions* (see Def. 2.26), denoted by $\mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$, as follows

$$\begin{aligned}\mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha}) := {} & \{\langle A, \{o\}\rangle^+ \mid A \in \mathcal{F} \cap \mathsf{N_C}, o \in \mathsf{L}^+[\mathfrak{s}, \boldsymbol{\alpha}, A]\} \cup \\ & \{\langle A, \{o\}\rangle^- \mid A \in \mathcal{F} \cap \mathsf{N_C}, o \in \mathsf{L}^-[\mathfrak{s}, \boldsymbol{\alpha}, A]\} \cup \\ & \{\left\langle P, \{(o, o')\}\right\rangle^+ \mid P \in \mathcal{F} \cap \mathsf{N_R}, (o, o') \in \mathsf{L}^+[\mathfrak{s}, \boldsymbol{\alpha}, P]\} \cup \\ & \{\left\langle P, \{(o, o')\}\right\rangle^- \mid P \in \mathcal{F} \cap \mathsf{N_R}, (o, o') \in \mathsf{L}^-[\mathfrak{s}, \boldsymbol{\alpha}, P]\}.\end{aligned}$$

The *set of all relevant local effects*, denoted by $\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$, is defined by

$$\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}}) := \bigcup_{\substack{\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}, \\ \boldsymbol{\alpha} \in \boldsymbol{A}}} \mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha}).$$

▲

The functions $\mathsf{L}^+[\cdot]$ and $\mathsf{L}^-[\cdot]$ are computable. Thus, also the sets $\mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$ are computable. The following lemma is a direct consequence of the definitions above.

Lemma 4.5. *Let* $\boldsymbol{A}$ *and* $\mathsf{Loc}_{\boldsymbol{A}}$ *be as in Def. 4.4 and* $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})$ *the induced FO-DS. For each* $\boldsymbol{\alpha} \in \boldsymbol{A}$ *and state* $\mathcal{I}$ *in* $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})$ *it holds that* $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})} \mathcal{I}^{\mathsf{L}}$ *with* $\mathsf{L} = \mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$.

Proof. The claim follows directly from Definition 4.4, the definition of interpretation updates (Definition 2.28) and Lemma 2.31. □

Also the effects of a sequence of actions can be described as a single set of local effects as we show in the following.

For a set of unconditional local effects $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ the *set of complementary effects*, denoted by $\neg\mathsf{L}$, is given by

$$\neg\mathsf{L} := \{\langle F, X\rangle^+ \mid \langle F, X\rangle^- \in \mathsf{L}\} \cup \{\langle F, X\rangle^- \mid \langle F, X\rangle^+ \in \mathsf{L}\}.$$

We can reduce iterated update operations with local effects to an update with a single set of local effects:

Lemma 4.6. *Let* $\boldsymbol{A}$*,* $\mathsf{Loc}_{\boldsymbol{A}}$ *and* $\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ *be as in Def. 4.4,* $\mathsf{L}, \mathsf{L}' \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ *sets of local effects and* $\mathcal{I}$ *an interpretation. It holds that*

$$(\mathcal{I}^{\mathsf{L}})^{\mathsf{L}'} = \mathcal{I}^{((\mathsf{L} \setminus \neg\mathsf{L}') \cup \mathsf{L}')}.$$

Proof. By definition of the update operation we have

$$\Delta_{(\mathcal{I}^{\mathsf{L}})^{\mathsf{L}'}} = \Delta_{\mathcal{I}^{((\mathsf{L} \setminus \neg\mathsf{L}') \cup \mathsf{L}')}} = \Delta_{\mathcal{I}}$$

and

$$o^{(\mathcal{I}^{\mathsf{L}})^{\mathsf{L}'}} = o^{\mathcal{I}^{((\mathsf{L} \setminus \neg\mathsf{L}') \cup \mathsf{L}')}} = o^{\mathcal{I}} \text{ for all } o \in \mathsf{N_O}.$$

Next, we show

$$A^{(\mathcal{I}^{\mathsf{L}})^{\mathsf{L}'}} = A^{\mathcal{I}^{((\mathsf{L}\setminus\neg\mathsf{L}')\cup\mathsf{L}')}}$$

for all $A \in \mathsf{N_C}$. Let $A \in \mathsf{N_C}$. With the definition of updates and basic set theory it follows that

$$\begin{aligned}
& A^{(\mathcal{I}^{\mathsf{L}})^{\mathsf{L}'}} \\
= \; & A^{\mathcal{I}^{\mathsf{L}}} \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\} \cup \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}'\} \\
= \; & \Big(\big((A^{\mathcal{I}} \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}\}) \cup \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}\}\big) \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\}\Big) \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}'\} \\
= \; & \big(A^{\mathcal{I}} \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}\}\big) \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\} \cup \\
& \big(\{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}\} \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\}\big) \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}'\}. \\
= \; & \Big(A^{\mathcal{I}} \setminus \big(\{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}\} \cup \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\}\big)\Big) \cup \\
& \big(\{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}\} \setminus \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\}\big) \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}'\}. \\
= \; & \Big(A^{\mathcal{I}} \setminus \big(\{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}, \langle A,\{b\}\rangle^+ \notin \mathsf{L}'\} \cup \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\}\big)\Big) \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}, \langle A,\{b\}\rangle^- \notin \mathsf{L}'\} \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}'\}. \\
= \; & \Big(A^{\mathcal{I}} \setminus \big(\{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in (\mathsf{L} \setminus \neg\mathsf{L}')\} \cup \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^- \in \mathsf{L}'\}\big)\Big) \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in (\mathsf{L} \setminus \neg\mathsf{L}')\} \cup \\
& \{b^{\mathcal{I}} \mid \langle A,\{b\}\rangle^+ \in \mathsf{L}'\}. \\
= \; & A^{\mathcal{I}^{((\mathsf{L}\setminus\neg\mathsf{L}')\cup\mathsf{L}')}}.
\end{aligned}$$

We omit the proof for role names. It is analogous to the one for concept names. □

Thus, the resulting interpretation after executing a sequence of local-effect actions from $\boldsymbol{A}$ can be expressed as an update of the initial interpretation w.r.t. a subset of $\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$. As a consequence of this lemma it follows that there are only finitely many reachable interpretations from a fixed initial state in the transition system induced by a program over local effect actions, because there are only finitely many possible updates of an initial interpretation.

Based on these observations we can now define an appropriate representation of dynamic types in presence of local-effect actions.

Definition 4.7. Let $\boldsymbol{A}$ be a finite set of ground actions, $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ an $\mathcal{L}$-admissible local effect representation of $\boldsymbol{A}$ and $\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ the set of all relevant local effects as defined above.

A *dynamic type w.r.t.* $\mathsf{Lit}(\mathsf{Loc}_A)$ *and* $\mathcal{C}$ is a set

$$\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_A)}$$

satisfying the following two conditions

- $\mathfrak{t}$ is *complete*: for all $\psi \in \mathcal{C}$ and all $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_A)$ it holds that $(\psi, \mathsf{L}) \in \mathfrak{t}$ or $(\neg\psi, \mathsf{L}) \in \mathfrak{t}$ (modulo elimination of double negation).
- $\mathfrak{t}$ is *realizable*: there exists an interpretation $\mathcal{I}$ in the state space $\mathbb{I}$ of $\mathfrak{D}(\mathsf{Loc}_A)$ such that for all $(\psi, \mathsf{L}) \in \mathfrak{t}$ it holds that $\mathcal{I}^{\mathsf{L}} \models \psi$.

The *set of all dynamic types w.r.t.* $\mathsf{Lit}(\mathsf{Loc}_A)$ *and* $\mathcal{C}$ is denoted by $\mathsf{D\text{-}Types}(\mathsf{Loc}_A)$.

Let $\mathcal{I}$ be an element of the state space of $\mathfrak{D}(\mathsf{Loc}_A)$. The *dynamic type of* $\mathcal{I}$ *in* $\mathfrak{D}(\mathsf{Loc}_A)$, denoted by $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$, is defined by

$$\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) := \{(\psi, \mathsf{L}) \mid \psi \in \mathcal{C}, \mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_A), \mathcal{I}^{\mathsf{L}} \models \psi\}.$$

▲

A dynamic type is a set of pairs. A pair of the form (ψ, L), where ψ is a context axiom and L a set of unconditional local effects, belongs to a dynamic type of an interpretation $\mathcal{I}$ if ψ is true *after* an update of $\mathcal{I}$ with L.

Example 4.8 (Example 4.3 continued). The context is $\mathcal{C} := \{\varphi_{\mathsf{con}}, \varphi_{\exists}, \neg\varphi_{\mathsf{con}}, \neg\varphi_{\exists}\}$ with

$$\varphi_{\mathsf{con}} := ((\mathsf{dev}, \mathsf{bat}) \sqsubseteq \mathit{ConTo}) \text{ and } \varphi_{\exists} := (\mathsf{dev} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}).$$

There is only the single action $\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})$ in the action theory Σ. Thus, the set of all relevant effects is given by

$$\mathsf{Lit}(\Sigma) := \big\{\langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^+, \langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^-\big\}.$$

For the construction of the dynamic types the following subsets of $\mathsf{Lit}(\Sigma)$ are relevant:

$$\mathsf{L}_0 := \emptyset; \quad \mathsf{L}_1 := \big\{\langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^+\big\}; \quad \mathsf{L}_2 := \big\{\langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^-\big\}.$$

Note that L_0 describes the effects of the empty action sequence. Since φ_{con} is true initially, L_1 describes the changes after an *even number* of $\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})$ executions and L_2 is the set of all effects of action sequences that consist of an *odd number* of $\mathtt{toggle}(\mathsf{dev}, \mathsf{bat})$ actions. Thus, we capture all relevant changes of the program. We can ignore the set $\big\{\langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^+, \langle \mathit{ConTo}, \{(\mathsf{dev}, \mathsf{bat})\}\rangle^-\big\}$ because it leads to the same update as L_1.

We consider the dynamic types of $\mathcal{I}_1$ and $\mathcal{I}_2$ (see Example 4.3). Recall that in $\mathcal{I}_1$ the power supply bat is the only one connected to dev whereas in $\mathcal{I}_2$ there are two power supplies for dev. Our notion of dynamic types captures this difference. The dynamic types of $\mathcal{I}_1$ and $\mathcal{I}_2$ w.r.t. $\mathsf{Lit}(\Sigma)$ and $\mathcal{C}$ are

$$\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_1) = \{(\varphi_{\mathsf{con}}, \mathsf{L}_0), (\varphi_{\exists}, \mathsf{L}_0), (\varphi_{\mathsf{con}}, \mathsf{L}_1), (\varphi_{\exists}, \mathsf{L}_1), (\neg\varphi_{\mathsf{con}}, \mathsf{L}_2), \boldsymbol{(\neg\varphi_{\exists}, \mathsf{L}_2)}\};$$
$$\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_2) = \{(\varphi_{\mathsf{con}}, \mathsf{L}_0), (\varphi_{\exists}, \mathsf{L}_0), (\varphi_{\mathsf{con}}, \mathsf{L}_1), (\varphi_{\exists}, \mathsf{L}_1), (\neg\varphi_{\mathsf{con}}, \mathsf{L}_2), \boldsymbol{(\varphi_{\exists}, \mathsf{L}_2)}\}.$$

We have that $\varphi_\exists$ is false in the update of $\mathcal{I}_1$ with L_2 but is still true in the corresponding update of $\mathcal{I}_2$. Note that the two types described above are the only relevant realizable dynamic types in this domain. φ_{con} and $\varphi_\exists$ are true initially according in to the initial KB. Therefore, the pairs $(\varphi_{\mathsf{con}}, \mathsf{L}_0)$ and $(\varphi_\exists, \mathsf{L}_0)$ are fixed. Connecting dev and bat (as it is done with L_1) obviously causes φ_{con} and $\varphi_\exists$ to be true. Consequently, the pairs $(\varphi_{\mathsf{con}}, \mathsf{L}_1)$ and $(\varphi_\exists, \mathsf{L}_1)$ are part of every realizable dynamic type. The same is true for the pair $(\neg\varphi_{\mathsf{con}}, \mathsf{L}_2)$. Hence, there are only two realizable dynamic types satisfying the initial KB. ▲

Next, we prove some basic properties of dynamic types.

Lemma 4.9. *Let $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ be the FO-DS induced by $\mathsf{Loc}_{\boldsymbol{A}}$.*

1. $\mathsf{D\text{-}Types}(\mathsf{Loc}_{\boldsymbol{A}}) = \{\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) \mid \mathcal{I} \in \mathbb{I}\}$.
2. *For all $\mathcal{I}, \mathcal{J} \in \mathbb{I}$ it holds that*
$$\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J}) \textit{ implies } \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J}).$$
3. *For all $\mathcal{I}, \mathcal{J} \in \mathbb{I}$ and all $\boldsymbol{\alpha} \in \boldsymbol{A}$ it holds that*
$$\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J}) \textit{ implies } \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}') = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J}'), \textit{ where}$$
$\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})} \mathcal{I}'$ *and* $\mathcal{J} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})} \mathcal{J}'$.

Proof. Let $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ be the FO-DS induced by $\mathsf{Loc}_{\boldsymbol{A}}$.

1. Let $\mathfrak{t} \in \mathsf{D\text{-}Types}(\mathsf{Loc}_{\boldsymbol{A}})$ and $\mathcal{I} \in \mathbb{I}$ such that $\mathcal{I}^{\mathsf{L}} \models \psi$ for all $(\psi, \mathsf{L}) \in \mathfrak{t}$. Such an interpretation exists because $\mathfrak{t}$ is realizable. We show that $\mathfrak{t} = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$. By assumption, we have $\mathfrak{t} \subseteq \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$. We show $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) \subseteq \mathfrak{t}$. Let $(\psi', \mathsf{L}') \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$ and assume to the contrary that $(\psi', \mathsf{L}') \notin \mathfrak{t}$. Since $\mathfrak{t}$ is complete it follows that $(\neg\psi', \mathsf{L}') \in \mathfrak{t}$, which implies $\mathcal{I}^{\mathsf{L}'} \models \neg\psi'$. This is a contradiction to $(\psi', \mathsf{L}') \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$. It is implied that $\mathsf{D\text{-}Types}(\mathsf{Loc}_{\boldsymbol{A}}) \subseteq \{\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) \mid \mathcal{I} \in \mathbb{I}\}$. It is easy to see that the set $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$ for some $\mathcal{J} \in \mathbb{I}$ is complete and realizable. Therefore,
$$\{\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) \mid \mathcal{I} \in \mathbb{I}\} \subseteq \mathsf{D\text{-}Types}(\mathsf{Loc}_{\boldsymbol{A}}).$$
2. The claim follows from
$$\psi \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}) \text{ iff } \mathcal{I} \models \psi \text{ iff } \mathcal{I}^{\emptyset} \models \psi \text{ iff } (\psi, \emptyset) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$$
for any $\psi \in \mathcal{C}$ and $\mathcal{I} \in \mathbb{I}$.
3. Let $\mathcal{I} \in \mathbb{I}$, $\boldsymbol{\alpha} \in \boldsymbol{A}$ and $\mathcal{I}'$ with $\mathcal{I} \Rightarrow^{\alpha}_{\mathfrak{D}} \mathcal{I}'$. Lemma 4.5 implies that
$$\mathcal{I}' = \mathcal{I}^{\mathsf{L}_{\boldsymbol{\alpha}}} \text{ with } \mathsf{L}_{\boldsymbol{\alpha}} = \mathsf{Loc}_{\boldsymbol{A}}(\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}), \boldsymbol{\alpha})$$
We first prove that for any element $(\psi, \mathsf{L}) \in \mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$ it holds that
$$(\psi, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}') \text{ iff } (\psi, (\mathsf{L}_{\boldsymbol{\alpha}} \setminus \neg\mathsf{L}) \cup \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}). \tag{4.3}$$
We have $(\psi, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}')$

iff $\mathcal{I}'^{\mathsf{L}} \models \psi$

iff $(\mathcal{I}^{\mathsf{L}_{\boldsymbol{\alpha}}})^{\mathsf{L}} \models \psi$ (with $\mathcal{I}' = \mathcal{I}^{\mathsf{L}_{\boldsymbol{\alpha}}}$)

iff $\mathcal{I}^{((\mathsf{L}_{\boldsymbol{\alpha}} \setminus \neg \mathsf{L}) \cup \mathsf{L})} \models \psi$ (Lemma 4.6)

iff $(\psi, (\mathsf{L}_{\boldsymbol{\alpha}} \setminus \neg \mathsf{L}) \cup \mathsf{L}) \in \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$.

Next, we consider another interpretation $\mathcal{J} \in \mathbb{I}$ and the corresponding resulting interpretation $\mathcal{J}'$ with $\mathcal{J} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}})} \mathcal{J}'$. Assume $\text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$ and let $(\psi, \mathsf{L}) \in \mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$. We have $(\psi, \mathsf{L}) \in \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}')$

iff $(\psi, (\mathsf{L}_{\boldsymbol{\alpha}} \setminus \neg \mathsf{L}) \cup \mathsf{L}) \in \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I})$ (by (4.3))

iff $(\psi, (\mathsf{L}_{\boldsymbol{\alpha}} \setminus \neg \mathsf{L}) \cup \mathsf{L}) \in \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$ (by assumption $\text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$)

iff $(\psi, \mathsf{L}) \in \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J}')$.

□

Intuitively, the dynamic type of a particular interpretation not only provides the set of relevant axioms that are true now (static type) but also those axioms that are true in all possible future evolutions of it.

We obtain that the binary relation on $\mathbb{I}$ defined by

$$\text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$$

satisfies the properties D1 (Lemma 4.9.2) and D2 (Lemma 4.9.3). Furthermore, there are only finitely many dynamic types.

Checking Realizability in $\mathcal{ALCQIO}$

To use dynamic types for constructing an abstract transition system of a program we have to show that the set of all dynamic types is effectively computable. We prove this for the standard DL $\mathcal{ALCQIO}$ as a base logic. Given an $\mathcal{ALCQIO}$-context $\mathcal{C}$ and the set of relevant local effects $\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ all finitely many complete subsets of $\mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$ can be enumerated. It remains to be shown that checking realizability of a complete subset of $\mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$ is decidable.

For a complete subset $\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$ based on an $\mathcal{ALCQIO}$-context $\mathcal{C}$ we construct a reduction $\mathcal{ALCQIO}$-KB $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ that is consistent iff $\mathfrak{t}$ is realizable.

With $\mathsf{Ind}(\boldsymbol{A})$ we denote the finite set of all object names occurring in $\boldsymbol{A}$, which corresponds to the objects mentioned in $\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$.

We can assume without loss of generality that all Boolean KBs in $\mathcal{C}$ are of the form

$$o \sqsubseteq C, \neg(o \sqsubseteq C), C \sqsubseteq D \text{ or } \neg(C \sqsubseteq D), \tag{4.4}$$

where o is an object name and C and D are $\mathcal{ALCQIO}$-concepts. Instead of a context with complex Boolean combinations of axioms it is sufficient to consider all ABox assertions and concept inclusions occurring in the context and their negation. Positive and negative role assertions are written as concept assertions using nominals according to the following rules

$$(o,o') \sqsubseteq R \rightsquigarrow o \sqsubseteq (\exists R.\{o'\}),$$
$$(o,o') \sqsubseteq \neg R \rightsquigarrow o \sqsubseteq (\forall R.\neg\{o'\}).$$

Note that in $\mathcal{ALCQIO}$ the role R is restricted to be a role name or the inverse of a role name.

The idea for the construction of the reduction knowledge base is based on the so called *reduction approach* that was first introduced in [Baa+05a] for solving the projection problem. An extension to a setting with concept inclusions is given in [BLL10]. The main idea is to encode an interpretation and all its possible updates into a single model of the reduction knowledge base by introducing new auxiliary concept names and role names. It is sufficient to consider subsets of literals $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ that are *non-contradictory*. L is non-contradictory iff there are *no* effects of the form

$$\left\{\left\langle P,\{(o,o')\}\right\rangle^{+},\left\langle P,\{(o,o')\}\right\rangle^{-}\right\} \subseteq \mathsf{L} \text{ or } \left\{\langle A,\{o\}\rangle^{+},\langle A,\{o\}\rangle^{-}\right\} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}}).$$

For the construction the notion of a sub-concept is used.

Definition 4.10. Given an $\mathcal{ALCQIO}$-concept C, the *set of all subconcepts of* C, denoted by $\mathsf{sub}(C)$, is defined inductively as follows:

$\mathsf{sub}(C) := \{C\}$ with C of the form $\top, \bot,$ or A, with $A \in \mathsf{N_C}$, or $\{o\}$ with $o \in \mathsf{N_O}$;
$\mathsf{sub}(C) := \{C\} \cup \mathsf{sub}(C') \cup \mathsf{sub}(D')$ with C of the form $C' \sqcap D'$ or $C' \sqcup D'$,
$\mathsf{sub}(C) := \{C\} \cup \mathsf{sub}(C')$ with C of the form $\neg C', \exists R.C', \forall R.C', {\geq} nR.C'$ or ${\leq} nR.C'$.

For a Boolean $\mathcal{ALCQIO}$-KB or a set of Boolean $\mathcal{ALCQIO}$-KBs X, the set of all subconcepts occurring in X, denoted by $\mathsf{sub}(X)$, is defined accordingly. ▲

Furthermore, let $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation. The domain elements in

$$\{o^{\mathcal{I}} \mid o \in \mathsf{Ind}(\boldsymbol{A})\}$$

are called *named* and all the other elements in $\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Ind}(\boldsymbol{A})\}$ *unnamed*.

For a given $\mathcal{ALCQIO}$-admissible local effect representation $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^{+}, \mathsf{L}^{-}, \mathsf{Pre}_{\mathcal{C}})$ of $\boldsymbol{A}$ the following new concept names and role names not contained in $\mathcal{F}$ are used for the reduction:

- for each non-contradictory set of effects $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ and each name $F \in \mathcal{F}$ there is a *fresh name denoted by* $F^{(\mathsf{L})}$. In case $\mathsf{L} = \emptyset$ the name $F^{(\emptyset)}$ represents the initial interpretation of F and $F^{(\mathsf{L})}$ with $\mathsf{L} \neq \emptyset$ stands for the *named* part of the extension of F after an update with L.

- for each non-contradictory set of effects $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ and each concept $C \in \mathsf{sub}(\mathcal{C})$ a *newly introduced concept name* $T_C^{(\mathsf{L})}$ encodes the extension of C after an update with the effects in L.

Furthermore, a new name N is introduced representing the named part of the domain. To define N the following concept definition is used:

$$N \equiv \bigsqcup_{o \in \mathsf{Ind}(A)} \{o\}.$$

For each newly introduced concept name of the form $T_C^{(\mathsf{L})}$ with $C \in \mathsf{sub}(\mathcal{C})$ and non-contradictory set $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_A)$ a corresponding concept definition is defined by induction on the structure of C in Figure 4.1. In case R is of the form $\mathsf{inv}(P)$ the expressions $R^{(\emptyset)}$ and $R^{(\mathsf{L})}$ stand for the roles $\mathsf{inv}(P^{(\emptyset)})$ and $\mathsf{inv}(P^{(\mathsf{L})})$, respectively. The TBox, denoted by

$$\mathcal{T}_{\mathsf{sub}(\mathcal{C})},$$

is defined as the set of all concept definitions for all the new concept names $T_C^{(\mathsf{L})}$ with $C \in \mathsf{sub}(\mathcal{C})$ and $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_A)$.

$$\begin{aligned}
T_A^{(\mathsf{L})} &\equiv (N \sqcap A^{(\mathsf{L})}) \sqcup (\neg N \sqcap A^{(\emptyset)}), \text{ with } A \in \mathcal{F} \cap \mathsf{N_C};\\
T_B^{(\mathsf{L})} &\equiv B \text{ with } B \text{ of the form } \{o\}, \top \text{ or } \bot;\\
T_{\neg C}^{(\mathsf{L})} &\equiv \neg T_C^{(\mathsf{L})};\\
T_{C \sqcap D}^{(\mathsf{L})} &\equiv T_C^{(\mathsf{L})} \sqcap T_D^{(\mathsf{L})};\\
T_{C \sqcup D}^{(\mathsf{L})} &\equiv T_C^{(\mathsf{L})} \sqcup T_D^{(\mathsf{L})};\\
T_{\exists R.C}^{(\mathsf{L})} &\equiv \Big(N \sqcap \big(\ (\exists R^{(\emptyset)}.(\neg N \sqcap T_C^{(\mathsf{L})})\ ) \sqcup (\ \exists R^{(\mathsf{L})}.(N \sqcap T_C^{(\mathsf{L})})\ )\ \big)\Big) \sqcup\\
&\quad (\neg N \sqcap \exists R^{(\emptyset)}.T_C^{(\mathsf{L})});\\
T_{\forall R.C}^{(\mathsf{L})} &\equiv \Big(N \rightarrow \big(\ (\forall R^{(\emptyset)}.(\neg N \rightarrow T_C^{(\mathsf{L})})\ ) \sqcap (\ \forall R^{(\mathsf{L})}.(N \rightarrow T_C^{(\mathsf{L})})\ )\ \big)\Big) \sqcap\\
&\quad (\neg N \rightarrow \forall R^{(\emptyset)}.T_C^{(\mathsf{L})});\\
T_{\geq mR.C}^{(\mathsf{L})} &\equiv \Big(N \sqcap \bigsqcup_{0 \leq j \leq m} \big(\ \geq j R^{(\mathsf{L})}.(N \sqcap T_C^{(\mathsf{L})}) \sqcap \geq (m-j) R^{(\emptyset)}.(\neg N \sqcap T_C^{(\mathsf{L})})\ \big)\Big) \sqcup\\
&\quad (\neg N \sqcap \geq m R^{(\emptyset)}.T_C^{(\mathsf{L})}).
\end{aligned}$$

Figure 4.1: Concept definition for $T_C^{(\mathsf{L})}$

To capture the changes on the relevant names we define an ABox $\mathcal{A}_{\mathsf{eff}}^{(\mathsf{L})}$ for each non-contradictory set of effects $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\mathcal{A}})$ as follows:

$$\begin{aligned}
\mathcal{A}_{\mathsf{eff}}^{(\mathsf{L})} := {} & \left\{o \in A^{(\mathsf{L})} \,\middle|\, \langle A, \{o\}\rangle^{+} \in \mathsf{L}\right\} \cup \left\{o \in \neg A^{(\mathsf{L})} \,\middle|\, \langle A, \{o\}\rangle^{-} \in \mathsf{L}\right\} \cup \\
& \left\{(o,o') \in P^{(\mathsf{L})} \,\middle|\, \big\langle P, \{(o,o')\}\big\rangle^{+} \in \mathsf{L}\right\} \cup \left\{(o,o') \in \neg P^{(\mathsf{L})} \,\middle|\, \big\langle P, \{(o,o')\}\big\rangle^{-} \in \mathsf{L}\right\} \cup \\
& \left\{o \in (A^{(\emptyset)} \rightarrow A^{(\mathsf{L})}) \,\middle|\, A \in \mathcal{F} \cap \mathsf{N_C}, o \in \mathsf{Ind}(\mathcal{A}), \langle A, \{o\}\rangle^{-} \notin \mathsf{L}\right\} \cup \\
& \left\{o \in (\neg A^{(\emptyset)} \rightarrow \neg A^{(\mathsf{L})}) \mid A \in \mathcal{F} \cap \mathsf{N_C}, o \in \mathsf{Ind}(\mathcal{A}), \langle A, \{o\}\rangle^{+} \notin \mathsf{L}\right\} \cup \\
& \left\{o \in (\exists P^{(\emptyset)}.\{o'\} \rightarrow \exists P^{(\mathsf{L})}.\{o'\}) \,\middle|\, P \in \mathcal{F} \cap \mathsf{N_R}, o, o' \in \mathsf{Ind}(\mathcal{A}), \big\langle P, \{(o,o')\}\big\rangle^{-} \notin \mathsf{L}\right\} \cup \\
& \left\{o \in (\forall P^{(\emptyset)}.\neg\{o'\} \rightarrow \forall P^{(\mathsf{L})}.\neg\{o'\}) \,\middle|\, P \in \mathcal{F} \cap \mathsf{N_R}, o, o' \in \mathsf{Ind}(\mathcal{A}), \big\langle P, \{(o,o')\}\big\rangle^{+} \notin \mathsf{L}\right\}.
\end{aligned}$$

In $\mathcal{A}_{\mathsf{eff}}^{(\mathsf{L})}$ we first state that all effects in L are satisfied after the update. And second, we encode the frame assumption for all the named elements and relevant names. In other words: only the changes listed in L are realized and nothing else is changing. For the dynamic type all possible updates are relevant. Therefore we consider the union for all non-contradictory sets of literals:

$$\mathcal{A}_{\mathsf{eff}} := \bigcup_{\substack{\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\mathcal{A}}), \\ \mathsf{L} \text{ is non-contradictory}}} \mathcal{A}_{\mathsf{eff}}^{(\mathsf{L})}. \tag{4.5}$$

Next, we define the part of the reduction knowledge base that refers to a specific complete subset $\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\mathcal{A}})}$. For the negated concept inclusions in $\mathcal{C}$, we introduce some additional object names that serve as witnesses for the violation of a concept inclusion. For each CI $C \sqsubseteq D \in \mathcal{C}$ and each non-contradictory set of literals $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\mathcal{A}})$ we use a fresh separate object name $o_{\neg(C \sqsubseteq D),\mathsf{L}}$ not contained in $\mathsf{Ind}(\mathcal{A})$ and not mentioned in $\mathcal{C}$. We now define the following ABox and TBox for a given $\mathfrak{t}$ using the concept names defined in $\mathcal{T}_{\mathsf{sub}(\mathcal{C})}$:

$$\begin{aligned}
\mathcal{A}_{\mathsf{red}}^{\mathfrak{t}} := {} & \{o \in T_C^{(\mathsf{L})} \mid (o \in C, \mathsf{L}) \in \mathfrak{t}\} \cup \\
& \{o \in \neg T_C^{(\mathsf{L})} \mid (\neg(o \in C), \mathsf{L}) \in \mathfrak{t}\} \cup \\
& \{o_{\neg(C \sqsubseteq D),\mathsf{L}} \in T_C^{(\mathsf{L})} \sqcap \neg T_D^{(\mathsf{L})} \mid (\neg(C \sqsubseteq D), \mathsf{L}) \in \mathfrak{t}\};
\end{aligned}$$

$$\mathcal{T}_{\mathsf{red}}^{\mathfrak{t}} := \{T_C^{(\mathsf{L})} \sqsubseteq T_D^{(\mathsf{L})} \mid (C \sqsubseteq D, \mathsf{L}) \in \mathfrak{t}\}.$$

The reduction knowledge base $\mathcal{K}_{\mathsf{red}}^{\mathfrak{t}}$ consists of the TBox

$$\mathcal{T}_{\mathsf{sub}(\mathcal{C})} \cup \mathcal{T}_{\mathsf{red}}^{\mathfrak{t}} \cup \{N \equiv \bigsqcup_{o \in \mathsf{Ind}(\mathcal{A})} \{o\}\}$$

and the ABox

$$\mathcal{A}_{\mathsf{red}}^{\mathfrak{t}} \cup \mathcal{A}_{\mathsf{eff}} \cup \{o \not\approx o' \mid o, o' \in \mathsf{Ind}(\mathcal{A})\}.$$

We require that all the named objects from $\mathsf{Ind}(\mathcal{A})$ are interpreted differently. But the newly introduced object names $o_{\neg(C \sqsubseteq D),\mathsf{L}}$ might refer to named or to unnamed elements.

The following lemma describes the main properties of $\mathcal{T}_{\mathsf{sub}(\mathcal{C})}$ and $\mathcal{A}_{\mathsf{eff}}$.

Lemma 4.11. *Let* $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ *be an* $\mathcal{ALCQIO}$*-admissible local effect representation and* $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ *the induced FO-DS.*

1. *For every interpretation* $\mathcal{I} \in \mathbb{I}$ *with* $\mathcal{I} \models \{o \not\approx o' \mid o, o' \in \mathsf{Ind}(\boldsymbol{A})\}$ *there exists an interpretation* $\mathcal{J}$ *with*

$$\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C})} \cup \left\{ N \equiv \bigsqcup_{o \in \mathsf{Ind}(\boldsymbol{A})} \{o\} \right\} \textit{ and } \mathcal{J} \models \mathcal{A}_{\mathsf{eff}} \cup \left\{ o \not\approx o' \mid o, o' \in \mathsf{Ind}(\boldsymbol{A}) \right\}$$

such that

   a) *for all non-contradictory sets of effects* $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ *it holds that*

$$\mathcal{I}^{\mathsf{L}} \models o \in A \textit{ iff } \mathcal{J} \models o \in A^{(\mathsf{L})},$$

   for all concept names $A \in \mathcal{F}$ *and* $o \in \mathsf{Ind}(\boldsymbol{A})$*, and*

$$\mathcal{I}^{\mathsf{L}} \models (o', o'') \in P \textit{ iff } \mathcal{J} \models (o', o'') \in P^{(\mathsf{L})}$$

   for all $P \in \mathcal{F}$ *and* $o', o'' \in \mathsf{Ind}(\boldsymbol{A})$*;*

   b) *for all non-contradictory sets of effects* $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ *and concepts* $C \in \mathsf{sub}(\mathcal{C})$ *it holds that* $C^{\mathcal{I}^{\mathsf{L}}} = \big(T_C^{(\mathsf{L})}\big)^{\mathcal{J}}$.

2. *For every interpretation* $\mathcal{J}$ *with*

$$\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C})} \cup \left\{ N \equiv \bigsqcup_{o \in \mathsf{Ind}(\boldsymbol{A})} \{o\} \right\} \textit{ and } \mathcal{J} \models \mathcal{A}_{\mathsf{eff}} \cup \left\{ o \not\approx o' \mid o, o' \in \mathsf{Ind}(\boldsymbol{A}) \right\}$$

there exists an interpretation $\mathcal{I} \in \mathbb{I}$ *such that*

   a) *for all non-contradictory sets of literals* $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ *it holds that*

$$\mathcal{I}^{\mathsf{L}} \models o \in A \textit{ iff } \mathcal{J} \models o \in A^{(\mathsf{L})},$$

   for all $A \in \mathcal{F}$ *and* $o \in \mathsf{Ind}(\boldsymbol{A})$*, and*

$$\mathcal{I}^{\mathsf{L}} \models (o', o'') \in P \textit{ iff } \mathcal{J} \models (o', o'') \in P^{(\mathsf{L})}$$

   for all role names $P \in \mathcal{F}$ *and* $o', o'' \in \mathsf{Ind}(\boldsymbol{A})$*;*

   b) *for all non-contradictory sets of literals* $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ *and concepts* $C \in \mathsf{sub}(\mathcal{C})$ *it holds that* $C^{\mathcal{I}^{\mathsf{L}}} = \big(T_C^{(\mathsf{L})}\big)^{\mathcal{J}}$.

Proof. For the construction of $\mathcal{T}_{\mathsf{sub}(\mathcal{C})}$ and $\mathcal{A}_{\mathsf{eff}}$ we have used the ideas from [Baa+05a] and [Lip14]. The proof works by following the lines of the proof of Lemma 15 in [Baa+05b] and the proof of Lemma 6.30 in [Lip14]. We outline the main steps in the following.

1. Let $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}}) \in \mathbb{I}$ be an interpretation with $\mathcal{I} \models \{o \not\approx o' \mid o, o' \in \mathsf{Ind}(\boldsymbol{A})\}$. We define a corresponding interpretation $\mathcal{J} = (\Delta_{\mathcal{J}}, \cdot^{\mathcal{J}})$ as follows:

- $\Delta_{\mathcal{J}} := \Delta_{\mathcal{I}}$;
- $o^{\mathcal{J}} := o^{\mathcal{I}}$ for every $o \in \mathsf{N_O}$;
- $N^{\mathcal{J}} := \left\{ o^{\mathcal{J}} \;\middle|\; o \in \mathsf{Ind}(\boldsymbol{A}) \right\}$;
- $\left(A^{(\mathsf{L})}\right)^{\mathcal{J}} := A^{\mathcal{I}^{\mathsf{L}}}$ for every $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ and $A \in \mathcal{F} \cap \mathsf{N_C}$;
- $\left(P^{(\mathsf{L})}\right)^{\mathcal{J}} := P^{\mathcal{I}^{\mathsf{L}}}$ for every $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ and $P \in \mathcal{F} \cap \mathsf{N_R}$;
- $\left(T_C^{(\mathsf{L})}\right)^{\mathcal{J}} := C^{\mathcal{I}^{\mathsf{L}}}$ for every $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ and $C \in \mathsf{sub}(\mathcal{C})$.

By definition $\mathcal{J}$ satisfies the properties 1a and 1b. It remains to be shown that

$$\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C})} \text{ and } \mathcal{J} \models \mathcal{A}_{\mathsf{eff}}.$$

First, we show $\mathcal{J} \models \mathcal{A}_{\mathsf{eff}}$. Let $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$ be non-contradictory. It has to be shown that $\mathcal{J} \models \mathcal{A}_{\mathsf{eff}}^{(\mathsf{L})}$. Let $\langle A, \{o\}\rangle^+ \in \mathsf{L}$. It holds that $\mathcal{I}^{\mathsf{L}} \models o \sqsubseteq A$. Consequently, with property 1a we have $\mathcal{J} \models o \sqsubseteq A^{(\mathsf{L})}$. The cases with effects in L of the form $\left\langle P, \{o, o'\}\right\rangle^{\pm}$, $\langle A, \{o\}\rangle^-$ are similar. Let $A \in \mathcal{F} \cap \mathsf{N_C}$ and $o \in \mathsf{Ind}(\boldsymbol{A})$. First, assume $\langle A, \{o\}\rangle^- \notin \mathsf{L}$. It follows that $\mathcal{I} \models o \sqsubseteq A$ implies $\mathcal{I}^{\mathsf{L}} \models o \sqsubseteq A$. With property 1$a$ it follows that $\mathcal{J} \models o \sqsubseteq \left(A^{(\emptyset)} \rightarrow A^{(\mathsf{L})}\right)$. The other cases are analogous.

For the proof of $\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C})}$ the following properties are needed:

$$A^{\mathcal{I}} \setminus N^{\mathcal{J}} = A^{\mathcal{I}^{\mathsf{L}}} \setminus N^{\mathcal{J}} \text{ for all } A \in \mathcal{F} \cap \mathsf{N_C}, \mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}}),$$

and similarly for all $P \in \mathcal{F} \cap \mathsf{N_R}$

$$P^{\mathcal{I}} \setminus (N^{\mathcal{J}} \times N^{\mathcal{J}}) = P^{\mathcal{I}^{\mathsf{L}}} \setminus (N^{\mathcal{J}} \times N^{\mathcal{J}}) \text{ for all } \mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}}).$$

Using these properties it can be shown by induction on the structure of concepts that $\mathcal{J}$ satisfies all definitions in $\mathcal{T}_{\mathsf{sub}(\mathcal{C})}$.

2. The proof of this direction is omitted. It is easy to see that the steps used in the proof of Lemma 6.30 in [Lip14] can be adapted to the present setting as well.

□

Finally, we are ready to prove that consistency of $\mathcal{K}_{\mathsf{red}}^{\mathfrak{t}}$ corresponds to realizability of $\mathfrak{t}$.

Lemma 4.12. *Let* $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ *be as above and* $\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$ *a complete set. It holds that* $\mathcal{K}_{\mathsf{red}}^{\mathfrak{t}}$ *is consistent (without the UNA) iff* $\mathfrak{t}$ *is realizable.*

Proof. Let $\mathfrak{t}$ be as in the claim. Let $\mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})$. A corresponding non-contradictory set denoted by $\widehat{\mathsf{L}}$ is defined as follows:

$$\widehat{\mathsf{L}} := \{\langle F, \{\bar{o}\}\rangle^- \mid \langle F, \{\bar{o}\}\rangle^- \in \mathsf{L}, \langle F, \{\bar{o}\}\rangle^+ \notin \mathsf{L}\} \cup \{\langle F, \{\bar{o}\}\rangle^+ \in \mathsf{L}\}.$$

$\widehat{\mathsf{L}}$ consists of all positive effects contained in L and of all negative ones from L that have no positive counterpart in L. We give precedence to positive effects which is compatible with the semantics of actions. Therefore, for any set of effects L and interpretation $\mathcal{I} \in \mathbb{I}$ it holds that $\mathcal{I}^{\mathsf{L}} = \mathcal{I}^{\widehat{\mathsf{L}}}$. It follows that if $\mathfrak{t}$ is realizable, then

$$(\psi, \mathsf{L}) \in \mathfrak{t} \text{ iff } (\psi, \widehat{\mathsf{L}}) \in \mathfrak{t}$$

for all elements $(\psi, \mathsf{L}) \in \mathfrak{t}$. Consequently, it is sufficient to consider only non-contradictory sets of literals.

"⇒": First, we assume that $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ is consistent and show that $\mathfrak{t}$ is realizable. There exists an interpretation $\mathcal{J}$ with $\mathcal{J} \models \mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$. From Lemma 4.11 it follows that there exists an interpretation $\mathcal{I} \in \mathbb{I}$ with $C^{\mathcal{I}^{\mathsf{L}}} = \left(T_C^{(\mathsf{L})}\right)^{\mathcal{J}}$ for all non-contradictory sets of effects L and concepts $C \in \mathsf{sub}(\mathcal{C})$. It easy to see that $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \mathfrak{t}$.

"⇐": Assume $\mathfrak{t}$ is realizable. There exists an interpretation $\mathcal{I} \in \mathbb{I}$ with $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \mathfrak{t}$. For any interpretation $\mathcal{I}'$ satisfying

$$\begin{aligned}&\Delta_{\mathcal{I}'} = \Delta_{\mathcal{I}};\\ &F^{\mathcal{I}'} = F^{\mathcal{I}} \text{ for all } F \in \mathcal{F} \text{ and}\\ &o^{\mathcal{I}'} = o^{\mathcal{I}} \text{ for all object names } o \text{ occurring in } \boldsymbol{A} \text{ or } \mathcal{C}\end{aligned}$$

it holds that $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}')$. Since the new object names of the form $o_{\neg(C \sqsubseteq D),\mathsf{L}}$ do not occur in $\boldsymbol{A}$ or $\mathcal{C}$, the dynamic type of an interpretation is independent of the interpretation of the new object names. Consequently, we can choose an interpretation $\mathcal{Y} \in \mathbb{I}$ such that $\mathcal{Y}$ has the same domain as $\mathcal{I}$, agrees with $\mathcal{I}$ on the interpretation of all relevant names in $\boldsymbol{A}$ and $\mathcal{C}$ and in addition satisfies

$$o^{\mathcal{Y}}_{\neg(C \sqsubseteq D),\mathsf{L}} \in (C \sqcap \neg D)^{\mathcal{Y}} \text{ iff } (\neg(C \sqsubseteq D), \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{Y}).$$

The chosen interpretation $\mathcal{Y}$ satisfies $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{Y}) = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}) = \mathfrak{t}$. According to Lemma 4.11 there exists an interpretation $\mathcal{J}$ such that

$$\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C})} \cup \{N \equiv \bigsqcup_{o \in \mathsf{Ind}(\boldsymbol{A})} \{o\}\} \text{ and } \mathcal{J} \models \mathcal{A}_{\mathsf{eff}} \cup \{o \not\approx o' \mid o, o' \in \mathsf{Ind}(\boldsymbol{A})\} \text{ and}$$

$C^{\mathcal{Y}^{\mathsf{L}}} = \left(T_C^{(\mathsf{L})}\right)^{\mathcal{J}}$ for all non-contradictory sets of literals L and concepts $C \in \mathsf{sub}(\mathcal{C})$. It easily follows that $\mathcal{J} \models \mathcal{A}^{\mathfrak{t}}_{\mathsf{red}}$ and $\mathcal{J} \models \mathcal{T}^{\mathfrak{t}}_{\mathsf{red}}$. Therefore, we have $\mathcal{J} \models \mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$.

□

Note that the reduction approach also works within a smaller base logic

$$\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}.$$

Inverse roles and at least restrictions and at most restrictions are not needed to formulate the reduction knowledge base.

4.3 Deciding the Verification Problem

Finally, we can assemble a finite bisimilar propositional abstraction of the transition system induced by an $\mathcal{ALCQIO}$-ConGolog program over local effect actions.

Let $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ be an $\mathcal{ALCQIO}$-admissible local effect representation of a finite set of ground actions $\boldsymbol{A}$ such that the concept names *Final* and *Fail* are not contained in $\mathcal{F}$ and the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ are not contained in $\boldsymbol{A}$. The termination action $\boldsymbol{\epsilon}$ and the failure action $\mathfrak{f}$ are definable as local effect actions.

We define the *termination and failure extension* of $\mathsf{Loc}_{\boldsymbol{A}}$. Let

$$\boldsymbol{B} := \boldsymbol{A} \cup \{\boldsymbol{\epsilon}, \mathfrak{f}\}.$$

The extension is an $\mathcal{ALCQIO}$-admissible local effect representation, denoted by $\mathsf{Loc}_{\boldsymbol{B}}$, that is obtained from $\mathsf{Loc}_{\boldsymbol{A}}$ as follows. We have

$$\begin{aligned}\mathsf{Loc}_{\boldsymbol{B}} = (\, & \mathcal{K} \cup \{\neg(\mathsf{prog} \sqsubseteq \mathit{Final}), \neg(\mathsf{prog} \sqsubseteq \mathit{Fail})\},\\ & \mathcal{F} \cup \{\mathit{Final}, \mathit{Fail}\},\\ & \mathcal{C}_{\boldsymbol{B}} := \mathcal{C} \cup \{(\mathsf{prog} \sqsubseteq \mathit{Final}), (\mathsf{prog} \sqsubseteq \mathit{Fail}), \neg(\mathsf{prog} \sqsubseteq \mathit{Final}), \neg(\mathsf{prog} \sqsubseteq \mathit{Fail})\},\\ & \mathsf{L}^+, \mathsf{L}^-,\\ & \mathsf{Pre}_{\mathcal{C}_{\boldsymbol{B}}}),\end{aligned}$$

where the functions $\mathsf{L}^+, \mathsf{L}^-$ are extended such that the following conditions are satisfied:

$$\begin{aligned}&\mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \boldsymbol{\alpha}) = \mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s} \cap \mathcal{C}, \boldsymbol{\alpha}) \text{ for all } (\mathfrak{s}, \boldsymbol{\alpha}) \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{B}}} \times \boldsymbol{A}, \text{ and}\\ &\mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \boldsymbol{\epsilon}) = \{\langle \mathit{Final}, \{\mathsf{prog}\}\rangle^+\} \text{ for all } \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{B}}}, \text{ and}\\ &\mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \mathfrak{f}) = \{\langle \mathit{Fail}, \{\mathsf{prog}\}\rangle^+\} \text{ for all } \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{B}}}.\end{aligned}$$

The extended possibility relation $\mathsf{Pre}_{\mathcal{C}_{\boldsymbol{B}}}$ is defined by

$$\begin{aligned}&(\mathfrak{s}, \boldsymbol{\alpha}) \in \mathsf{Pre}_{\mathcal{C}_{\boldsymbol{B}}} \text{ iff } ((\mathfrak{s} \cap \mathcal{C}), \boldsymbol{\alpha}) \in \mathsf{Pre}_{\mathcal{C}}, \text{ for all } (\mathfrak{s}, \boldsymbol{\alpha}) \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{B}}} \times \boldsymbol{A}, \text{ and}\\ &(\mathfrak{s}, \boldsymbol{\epsilon}) \in \mathsf{Pre}_{\mathcal{C}_{\boldsymbol{B}}} \text{ and } (\mathfrak{s}, \mathfrak{f}) \in \mathsf{Pre}_{\mathcal{C}_{\boldsymbol{B}}}, \text{ for all } \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}_{\boldsymbol{B}}}.\end{aligned}$$

To stay formally correct we view the symbols $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ as abbreviations of ground action terms with prog as the single argument. The definition of $\mathsf{Loc}_{\boldsymbol{B}}$ defines the effects and preconditions of $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ as defined in Definition 2.45. We have

$$\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}) \uplus \{\boldsymbol{\epsilon}, \mathfrak{f}\} = \mathfrak{D}(\mathsf{Loc}_{\boldsymbol{B}}). \tag{4.6}$$

We define a bisimilar propositional abstraction based on the set of all dynamic types and the characterization of the reachable subprograms $\mathsf{sub}(\delta)$ using the head and tail function.

Definition 4.13. Let $\mathcal{P} = (\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}), \delta)$ be an $\mathcal{ALCQIO}$-ConGolog program over local effect actions where $\boldsymbol{A}$, δ and $\mathsf{Loc}_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{L}^+, \mathsf{L}^-, \mathsf{Pre}_{\mathcal{C}})$ are as described in Definition 4.2. Furthermore, let $\boldsymbol{B} = \boldsymbol{A} \cup \{\boldsymbol{\epsilon}, \mathfrak{f}\}$, $\mathsf{Loc}_{\boldsymbol{B}}$ the termination and failure extension of $\mathsf{Loc}_{\boldsymbol{A}}$, $\mathcal{C}_{\boldsymbol{B}}$ the context in $\mathsf{Loc}_{\boldsymbol{B}}$ and AP a finite set of atomic propositions with a bijection $\iota : \mathcal{C}_{\boldsymbol{B}} \to \mathsf{AP}$.

The *abstraction of* $\mathfrak{I}_{\mathcal{P}}$ is a propositional transition system

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$$

over AP, where

- $Q_{\mathfrak{T}_{\mathcal{P}}} := \text{D-Types}(\mathsf{Loc}_{\boldsymbol{B}}) \times \{\mathsf{L} \mid \mathsf{L} \subseteq \mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{B}})\} \times \mathsf{sub}(\delta)$;
- $I_{\mathfrak{T}_{\mathcal{P}}} := \left\{(\mathfrak{t}, \emptyset, \delta) \in Q_{\mathfrak{T}_{\mathcal{P}}} \,\middle|\, (\varphi, \emptyset) \in \mathfrak{t} \text{ for all } \varphi \text{ occurring in the initial KB of } \mathsf{Loc}_{\boldsymbol{B}}\right\}$;
- $\hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} := \left\{\big((\mathfrak{t}, \mathsf{L}, \rho), (\mathfrak{t}', \mathsf{L}', \rho')\big) \in Q_{\mathfrak{T}_{\mathcal{P}}} \times Q_{\mathfrak{T}_{\mathcal{P}}} \,\middle|\, \mathfrak{t} = \mathfrak{t}', \text{ (i) or (ii) }\right\}$ with
  (i) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots ; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ such that $\mathfrak{a}$ is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C}_{\boldsymbol{B}} \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}$, and

  $$\mathsf{L}' = \big(\mathsf{L} \setminus \neg\mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})\big) \cup \mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$$

  and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$;

  (ii) there is *no* guarded action contained in $\mathsf{head}(\rho)$ that is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C}_{\boldsymbol{B}} \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}$ and we have

  $$\mathsf{L}' = \big(\mathsf{L} \setminus \neg\mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \mathfrak{f})\big) \cup \mathsf{Loc}_{\boldsymbol{A}}(\mathfrak{s}, \mathfrak{f})$$

  and $\rho = \rho'$;
- $\lambda_{\mathfrak{T}_{\mathcal{P}}} : (\mathfrak{t}, \mathsf{L}, \rho) \mapsto \{\iota(\psi) \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}$ for all $(\mathfrak{t}, \mathsf{L}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}}$.

▲

An abstract state in $Q_{\mathfrak{T}_{\mathcal{P}}}$ consists of three components: a dynamic type, a set of local effects representing the effects of the action sequence that has been executed so far and a reachable subprogram of δ representing the program that remains to be executed. The execution starts with a dynamic type of a model of the initial KB. Given the dynamic type of the initial model and the set of effects representing the accumulated effects of the action history, the static type $\mathfrak{s}$ of the current state is uniquely determined. Based on the static type executability of guarded actions from the head of the current program expression can be decided. Executability of a guarded action in a static type is defined as in Definition 3.19. The corresponding effects are computable in using $\mathsf{Loc}_{\boldsymbol{B}}$, and the next program expression is obtained using the tail function. Note that the abstract transition relation does *not* progress the dynamic type. In an abstract state we keep the dynamic type of the initial model and only update the set of effects and the remaining program. This is sufficient because the dynamic type of an initial model encodes the static types of all possible future evolutions of this model. The current static type itself is sufficient to determine the successor states.

Example 4.14 (Example 4.3 and 4.8 continued)**.** Recall that we consider the following program

$$\delta := \big(\texttt{toggle}(\mathsf{dev}, \mathsf{bat})\big)^*.$$

$$\begin{tikzcd}
{} \arrow[d] & \\
(\mathsf{t}_1,\mathsf{L}_0,\delta):\{\varphi_{\mathsf{con}},\varphi_{\exists},\neg\varphi_{\epsilon}\} \arrow[r] \arrow[d] & (\mathsf{t}_1,\mathsf{L}_{\epsilon},\langle\rangle):\{\varphi_{\mathsf{con}},\varphi_{\exists},\varphi_{\epsilon}\} \arrow[loop below] \\
(\mathsf{t}_1,\mathsf{L}_2,\delta):\{\neg\varphi_{\mathsf{con}},\neg\varphi_{\exists},\neg\varphi_{\epsilon}\} \arrow[r] \arrow[d, bend right] & (\mathsf{t}_1,\mathsf{L}_2\cup\mathsf{L}_{\epsilon},\langle\rangle):\{\neg\varphi_{\mathsf{con}},\neg\varphi_{\exists},\varphi_{\epsilon}\} \arrow[loop below] \\
(\mathsf{t}_1,\mathsf{L}_1,\delta):\{\varphi_{\mathsf{con}},\varphi_{\exists},\neg\varphi_{\epsilon}\} \arrow[r] \arrow[u, bend right] & (\mathsf{t}_1,\mathsf{L}_1\cup\mathsf{L}_{\epsilon},\langle\rangle):\{\varphi_{\mathsf{con}},\varphi_{\exists},\varphi_{\epsilon}\} \arrow[loop below]
\end{tikzcd}$$

Figure 4.2: Abstract transition system reachable from $(\mathsf{t}_1,\mathsf{L}_0,\delta)$

with the sets of relevant effects

$$\mathsf{L}_0 := \emptyset; \quad \mathsf{L}_1 := \big\{\langle \mathit{ConTo}, \{(\mathsf{dev},\mathsf{bat})\}\rangle^{+}\big\};$$
$$\mathsf{L}_2 := \big\{\langle \mathit{ConTo}, \{(\mathsf{dev},\mathsf{bat})\}\rangle^{-}\big\}; \quad \mathsf{L}_{\epsilon} := \big\{\langle \mathit{Final}, \{\mathsf{prog}\}\rangle^{+}\big\}$$

and the context $\mathcal{C}$ consisting of the following axioms and their corresponding negation:

$$\varphi_{\mathsf{con}} := ((\mathsf{dev},\mathsf{bat}) \sqsubseteq \mathit{ConTo}), \varphi_{\exists} := (\mathsf{dev} \sqsubseteq \exists \mathit{ConTo}.\mathit{PowerS}) \text{ and } \varphi_{\epsilon} := \mathsf{prog} \sqsubseteq \mathit{Final}$$

As we have seen in Example 4.8, there are the following two realizable dynamic types (not considering the termination action):

$$\mathsf{t}_1 := \{(\varphi_{\mathsf{con}},\mathsf{L}_0),(\varphi_{\exists},\mathsf{L}_0),(\varphi_{\mathsf{con}},\mathsf{L}_1),(\varphi_{\exists},\mathsf{L}_1),(\neg\varphi_{\mathsf{con}},\mathsf{L}_2),(\neg\varphi_{\exists},\mathsf{L}_2)\};$$
$$\mathsf{t}_2 := \{(\varphi_{\mathsf{con}},\mathsf{L}_0),(\varphi_{\exists},\mathsf{L}_0),(\varphi_{\mathsf{con}},\mathsf{L}_1),(\varphi_{\exists},\mathsf{L}_1),(\neg\varphi_{\mathsf{con}},\mathsf{L}_2),(\varphi_{\exists},\mathsf{L}_2)\}.$$

The abstract transition system has two initial states: $(\mathsf{t}_1,\mathsf{L}_0,\delta)$ and $(\mathsf{t}_2,\mathsf{L}_0,\delta)$. The reachable part from $(\mathsf{t}_1,\mathsf{L}_0,\delta)$ is depicted in Figure 4.2. The transition system starting from $(\mathsf{t}_2,\mathsf{L}_0,\delta)$ is exactly the same with the difference that the states $(\mathsf{t}_2,\mathsf{L}_2,\delta)$ and $(\mathsf{t}_2,\mathsf{L}_2\cup\mathsf{L}_{\epsilon},\langle\rangle)$ are labeled with $\varphi_{\exists}$ instead of $\neg\varphi_{\exists}$. ▲

Lemma 4.15. *$\mathfrak{T}_{\mathcal{P}}$ is effectively computable.*

Proof. We consider the transition system

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$$

as defined above. First, the set $\mathsf{D\text{-}Types}(\mathsf{Loc}_A)$ is effectively computable. All finitely many complete subsets of $\mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_A)}$ can be enumerated and checking realizability of a complete set is decidable due to Lemma 4.12. According to Theorem 3.15 $\mathsf{sub}(\delta)$ is finite. The characterization of the reachable subprograms based on the head and tail function yields that $\mathsf{sub}(\delta)$ is effectively computable. Therefore, $Q_{\mathfrak{T}_{\mathcal{P}}}$ and $I_{\mathfrak{T}_{\mathcal{P}}}$ are effectively computable. It is decidable to check whether

$$(\mathfrak{t}, \mathsf{L}, \rho) \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} (\mathfrak{t}, \mathsf{L}', \rho')$$

holds for two states in $Q_{\mathfrak{T}_{\mathcal{P}}}$: The static type $\{\psi \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}$ can be read off given $\mathfrak{t}$ and L. The guarded actions in $\mathsf{head}(\rho)$ are effectively computable as well as the program expressions in $\mathsf{tail}(\mathfrak{a}, \rho)$ for some $\mathfrak{a} \in \mathsf{head}(\rho)$. Since the static type is part of the abstract state and the precondition is provided by Σ, checking executability of a guarded action in a given state is decidable. The effects of a ground action are explicitly in Σ. Therefore, the accumulated effects of a successor state are effectively computable as well. $\square$

Let $\mathbb{I}$ be the state space of $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{B}})$ and $\mathcal{C}_{\boldsymbol{B}}$ the context in $\mathsf{Loc}_{\boldsymbol{B}}$ with $\boldsymbol{B} = \boldsymbol{A} \cup \{\epsilon, \mathfrak{f}\}$. A relation "$\simeq_{\mathcal{C}_{\boldsymbol{B}}}$" between the states of

$$\mathfrak{I}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}}) \text{ and } \mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$$

is defined as follows: we have

$$\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}_{\boldsymbol{B}}} (\mathfrak{t}, \mathsf{L}, \rho') \text{ iff the following conditions are satisfied}$$

- $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from some initial state $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \in I_{\mathcal{P}}$ such that

$$\mathfrak{t} = \text{d-type}^{\mathsf{loc}}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I}_0) \text{ and } \mathcal{I} = \mathcal{I}_0^{\mathsf{L}};$$

- $\rho = \rho'$.

Lemma 4.16. *$\simeq_{\mathcal{C}_{\boldsymbol{B}}}$ is a $\mathcal{C}_{\boldsymbol{B}}$-bisimulation.*

Proof. Let $\mathbb{I}$ be the state space of $\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{B}})$ and $\mathcal{C}_{\boldsymbol{B}}$ the context of the extended representation $\mathsf{Loc}_{\boldsymbol{B}}$ with $\boldsymbol{B} = \boldsymbol{A} \cup \{\epsilon, \mathfrak{f}\}$. Recall that $\mathfrak{T}_{\mathcal{P}}$ is a propositional transition system over AP and $\iota : \mathcal{C}_{\boldsymbol{B}} \to \mathsf{AP}$ is a bijection.

Assume $\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}_{\boldsymbol{B}}} (\mathfrak{t}, \mathsf{L}, \rho)$ for two states $\langle \mathcal{I}, \sigma, \rho \rangle \in Q_{\mathcal{P}}$ and $(\mathfrak{t}, \mathsf{L}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}}$. There exists $\mathcal{I}_0 \in \mathbb{I}$ such that $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from the initial state $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \in I_{\mathcal{P}}$ and

$$\text{d-type}^{\mathsf{loc}}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I}_0) = \mathfrak{t} \text{ and } \mathcal{I} = \mathcal{I}_0^{\mathsf{L}}.$$

First, we show that

$$\lambda_{\mathfrak{T}_{\mathcal{P}}}((\mathfrak{t}, \mathsf{L}, \rho)) = \{\iota(\psi) \mid \psi \in \text{s-type}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I})\}.$$

We have

$$\begin{aligned}
\lambda_{\mathfrak{T}_{\mathcal{P}}}((\mathfrak{t}, \mathsf{L}, \rho)) &= \{\iota(\psi) \mid (\psi, \mathsf{L}) \in \mathfrak{t}\} \\
&= \{\iota(\psi) \mid (\psi, \mathsf{L}) \in \text{d-type}^{\mathsf{loc}}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I}_0)\} \\
&= \{\iota(\psi) \mid \mathcal{I}_0^{\mathsf{L}} \models \psi\} \\
&= \{\iota(\psi) \mid \mathcal{I} \models \psi\} \\
&= \{\iota(\psi) \mid \psi \in \text{s-type}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I})\}.
\end{aligned}$$

Let

$$\mathfrak{s} = \{\psi \in \mathcal{C}_{\boldsymbol{B}} \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}.$$

$\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}_B} (\mathsf{t}, \mathsf{L}, \rho)$ implies $\mathsf{s\text{-}type}_{\mathcal{C}_B}(\mathcal{I}) = \mathfrak{s}$. We have to show that for every outgoing transition of $\langle \mathcal{I}, \sigma, \rho \rangle$ there is a matching transition from $(\mathsf{t}, \mathsf{L}, \rho)$ leading to a state that is in $\simeq_{\mathcal{C}_B}$-relation with the successor of $\langle \mathcal{I}, \sigma, \rho \rangle$ and vice versa. We distinguish four cases regarding σ.

(I) First, assume $\sigma \in \boldsymbol{A}^*$. We distinguish three types of transitions: a transition via an action from $\boldsymbol{A}$, a transition with the termination action ϵ and one with the failing action $\mathfrak{f}$.

(a) It holds that $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle$ for some $\boldsymbol{\alpha} \in \boldsymbol{A}$

iff $\langle \mathcal{I}, \sigma, \rho \rangle \rightarrow_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle$ for some $\boldsymbol{\alpha} \in \boldsymbol{A}$ (by definition of $\hookrightarrow_{\mathcal{P}}$)

iff $\langle \mathcal{I}, \sigma, \rho \rangle \rightarrow_{\mathfrak{D}(\mathsf{Loc}_A)} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle$ for some $\boldsymbol{\alpha} \in \boldsymbol{A}$ (by Lemma 3.2)

iff there exists a guarded action $\mathfrak{a} = \Lambda_1; \cdots; \Lambda_n; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$ such that $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$ and $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\mathsf{Loc}_A)} \mathcal{I}'$ (by Lemma 2.44 and Lemma 3.12)

iff there exists a guarded action $\mathfrak{a} = \Lambda_1; \cdots; \Lambda_n; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$ such that $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$ and $\mathfrak{a}$ is executable in $\mathsf{s\text{-}type}_{\mathcal{C}_B}(\mathcal{I}) = \mathfrak{s}$ and $\mathcal{I}' = \mathcal{I}^{\mathsf{Loc}_A(\mathfrak{s}, \boldsymbol{\alpha})}$ (by Lemma 3.20 and Lemma 4.5).

(b) It holds that $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \epsilon, \rho' \rangle$

iff $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\})$, $\rho' = \langle\rangle$ and $\mathcal{I} \Rightarrow^{\epsilon}_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$

iff $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Final}(\mathfrak{D}(\mathsf{Loc}_A))$, $\rho' = \langle\rangle$ and $\mathcal{I} \Rightarrow^{\epsilon}_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$ (by Lemma 3.2)

iff there exists $\mathfrak{a} = \Lambda_1; \cdots; \Lambda_n; \epsilon \in \mathsf{head}(\rho)$ for some $n \geq 0$ such that $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\mathcal{I} \Rightarrow^{\epsilon}_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$ (by Lemma 3.12)

iff there exists $\mathfrak{a} = \Lambda_1; \cdots; \Lambda_n; \epsilon \in \mathsf{head}(\rho)$ for some $n \geq 0$ such that $\mathfrak{a}$ is executable in $\mathsf{s\text{-}type}_{\mathcal{C}_B}(\mathcal{I}) = \mathfrak{s}$ and $\mathcal{I}' = \mathcal{I}^{\mathsf{Loc}_B(\mathfrak{s}, \epsilon)}$ (by Lemma 3.20 and by Lemma 4.5).

(c) It holds that $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \mathfrak{f}, \rho \rangle$

iff $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\})$ and $\mathcal{I} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$ (by definition of $\hookrightarrow_{\mathcal{P}}$)

iff $\langle \mathcal{I}, \sigma, \rho \rangle \in \mathsf{Fail}(\mathfrak{D}(\mathsf{Loc}_A))$ and $\mathcal{I} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$ (by Lemma 3.2)

iff there exists no $\mathfrak{a} \in \mathsf{head}(\rho)$ such that $\mathfrak{a}$ is executable in $\mathcal{I}$ and

$$\mathcal{I} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D}(\mathsf{Loc}_A) \uplus \{\epsilon, \mathfrak{f}\}} \mathcal{I}'$$

(by Lemma 3.12)

iff there exists no $\mathfrak{a} \in \mathsf{head}(\rho)$ such that $\mathfrak{a}$ is executable in $\mathsf{s\text{-}type}_{\mathcal{C}_B}(\mathcal{I}) = \mathfrak{s}$ and $\mathcal{I}' = \mathcal{I}^{\mathsf{Loc}_B(\mathfrak{s}, \mathfrak{f})}$ (by Lemma 3.20 and by Lemma 4.5).

Let $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \beta, \rho \rangle$ be an arbitrary outgoing transition of $\langle \mathcal{I}, \sigma, \rho \rangle$. We have $\beta \in \boldsymbol{A} \cup \{\epsilon, \mathfrak{f}\}$. Using (Ia), (Ib), (Ic) and the definition of $\hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}$ it follows that

$$(\mathsf{t}, \mathsf{L}, \rho) \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} \big(\mathsf{t}, \big(\mathsf{L} \setminus \neg\mathsf{Loc}_B(\mathfrak{s}, \beta)\big) \cup \mathsf{Loc}_B(\mathfrak{s}, \beta), \rho'\big),$$

where $\mathfrak{s} = \{\psi \in \mathcal{C}_B \mid (\psi, \mathsf{L}) \in \mathfrak{t}\} = \text{s-type}_{\mathcal{C}_B}(\mathcal{I})$. As shown above, we have

$$\mathcal{I}' = \mathcal{I}^{\mathsf{Loc}_B(\mathfrak{s},\beta)} \text{ and } \mathcal{I} = \mathcal{I}_0{}^{\mathsf{L}} \text{ with } \mathfrak{t} = \text{d-type}^{\text{loc}}_{\mathcal{C}_B}(\mathcal{I}_0).$$

Lemma 4.6 implies

$$\mathcal{I}' = \mathcal{I}_0{}^{\left(\mathsf{L}\setminus\neg\mathsf{Loc}_B(\mathfrak{s},\beta)\cup\mathsf{Loc}_B(\mathfrak{s},\beta)\right)}.$$

Consequently,

$$\langle \mathcal{I}', \sigma \cdot \beta, \rho \rangle \simeq_{\mathcal{C}_B} \big(\mathfrak{t}, \big(\mathsf{L} \setminus \neg\mathsf{Loc}_B(\mathfrak{s}, \beta)\big) \cup \mathsf{Loc}_B(\mathfrak{s}, \beta), \rho'\big).$$

Let $(\mathfrak{t}, \mathsf{L}, \rho) \hookrightarrow_{\mathfrak{T}_\mathcal{P}} (\mathfrak{t}, \mathsf{L}', \rho')$ be an outgoing transition of $(\mathfrak{t}, \mathsf{L}, \rho)$ in $\mathfrak{T}_\mathcal{P}$. It follows that

$$\mathsf{L}' = \big(\mathsf{L} \setminus \neg\mathsf{Loc}_B(\mathfrak{s}, \gamma)\big) \cup \mathsf{Loc}_B(\mathfrak{s}, \gamma)$$

for some $\gamma \in \boldsymbol{A} \cup \{\epsilon, \mathfrak{f}\}$ with $\mathfrak{s} = \{\psi \in \mathcal{C}_B \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}$. If $\gamma = \mathfrak{f}$, then there is no guarded action in $\mathsf{head}(\rho)$ that is executable in $\mathfrak{s}$. Otherwise, if $\gamma \in \boldsymbol{A} \cup \{\epsilon\}$, then there exists $\mathfrak{a} = \Lambda_1; \cdots; \Lambda_n; \epsilon \in \mathsf{head}(\rho)$ for some $n \geq 0$ such that $\mathfrak{a}$ is executable in $\mathfrak{s}$ and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$. Using (Ia), (Ib) and (Ic) it follows that

$$\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_\mathcal{P} \langle \mathcal{I}^{\mathsf{Loc}_B(\mathfrak{s},\gamma)}, \sigma \cdot \gamma, \rho' \rangle.$$

As shown above, we obtain

$$\langle \mathcal{I}^{\mathsf{Loc}_B(\mathfrak{s},\gamma)}, \sigma \cdot \gamma, \rho' \rangle \simeq_{\mathcal{C}_B} (\mathfrak{t}, \mathsf{L}', \rho').$$

(II) We assume $\sigma = \widehat{\sigma} \cdot \epsilon$ for some $\widehat{\sigma} \in (\boldsymbol{A}^* \cdot \{\epsilon\}^*)$. Since $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from an initial state, we have $\rho = \langle\rangle$.

It holds that $\langle \mathcal{I}, \sigma, \langle\rangle \rangle \hookrightarrow_\mathcal{P} \langle \mathcal{I}', \sigma', \rho' \rangle$ iff

$$\sigma' = \sigma \cdot \epsilon, \rho' = \langle\rangle \text{ and } \mathcal{I} \Rightarrow^{\epsilon}_{\mathfrak{D}(\mathsf{Loc}_A)\uplus\{\epsilon,\mathfrak{f}\}} \mathcal{I}'.$$

Since $\mathsf{head}(\langle\rangle) = \{\epsilon\}$ and ϵ is executable in all static types, we have

$$(\mathfrak{t}, \mathsf{L}, \langle\rangle) \hookrightarrow_{\mathfrak{T}_\mathcal{P}} \big(\mathfrak{t}, \big(\mathsf{L} \setminus \neg\mathsf{Loc}_B(\mathfrak{s}, \epsilon)\big) \cup \mathsf{Loc}_B(\mathfrak{s}, \epsilon), \langle\rangle\big)$$

by definition of $\hookrightarrow_{\mathfrak{T}_\mathcal{P}}$. It can be shown that $\langle \mathcal{I}, \sigma, \langle\rangle \rangle \simeq_{\mathcal{C}_B} (\mathfrak{t}, \mathsf{L}, \langle\rangle)$ implies also

$$\langle \mathcal{I}', \sigma \cdot \epsilon, \langle\rangle \rangle \simeq_{\mathcal{C}_B} \big(\mathfrak{t}, \big(\mathsf{L} \setminus \neg\mathsf{Loc}_B(\mathfrak{s}, \epsilon)\big) \cup \mathsf{Loc}_B(\mathfrak{s}, \epsilon), \langle\rangle\big).$$

The other direction is analogous, because both states above are the only successor states in case $\rho = \langle\rangle$.

(III) We assume $\sigma = \widehat{\sigma} \cdot \tilde{\sigma}$ with $\widehat{\sigma} \in \boldsymbol{A}^*$ and $\tilde{\sigma} \in \{\mathfrak{f}\}^*$ with $|\tilde{\sigma}| \geq 1$. As a consequence of Lemma 3.2 we have that only the failure action leads to a successor state of $\langle \mathcal{I}, \sigma, \rho \rangle$. Since $\mathfrak{f}$ was already executed before, the literal $\mathsf{prog} \sqsubseteq \mathit{Fail}$ is already true in $\mathcal{I}$ and executing $\mathfrak{f}$ does not change anything. Thus, we have

$$\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle \text{ implies } \mathcal{I}' = \mathcal{I}, \boldsymbol{\alpha} = \mathfrak{f} \text{ and } \rho' = \rho.$$

Consequently, $\langle \mathcal{I}, \sigma \cdot \mathfrak{f}, \rho \rangle$ is the only successor state of $\langle \mathcal{I}, \sigma, \rho \rangle$.

Since $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from the initial state $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle$, there must states in $\mathfrak{I}_{\mathcal{P}}$ of the form $\langle \mathcal{J}, \widehat{\sigma}, \rho \rangle$ and $\langle \mathcal{J}', \widehat{\sigma} \cdot \mathfrak{f}, \rho \rangle$ such that

$$\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \hookrightarrow_{\mathcal{P}}^{*} \langle \mathcal{J}, \widehat{\sigma}, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{J}', \widehat{\sigma} \cdot \mathfrak{f}, \rho \rangle \hookrightarrow_{\mathcal{P}}^{*} \langle \mathcal{I}, \sigma, \rho \rangle$$

We have $\langle \mathcal{J}, \widehat{\sigma}, \rho \rangle \in \mathsf{Fail}(\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}))$ and $\mathcal{J}' = \mathcal{J}^{\{\langle \mathit{Fail}, \{\mathsf{prog}\}\rangle^+\}}$ and $\mathcal{I} = \mathcal{J}'$. It follows that

$$\text{s-type}_{\mathcal{C}}(\mathcal{J}) = \text{s-type}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I}) \cap \mathcal{C} = \mathfrak{s} \cap \mathcal{C}$$

with $\mathfrak{s} = \{\psi \in \mathcal{C}_{\boldsymbol{B}} \mid (\psi, \mathsf{L}) \in \mathfrak{t}\}$. Let $\mathfrak{a} \in \mathsf{head}(\rho)$ be an arbitrary guarded action. With Lemma 3.12 $\langle \mathcal{J}, \widehat{\sigma}, \rho \rangle \in \mathsf{Fail}(\mathfrak{D}(\mathsf{Loc}_{\boldsymbol{A}}))$ implies that $\mathfrak{a}$ is not executable in $\mathcal{J}$. Therefore, $\mathfrak{a}$ is also not executable in $\mathfrak{s}$ due to Lemma 3.20. It follows that

$$\big(\mathfrak{t}, \big(\mathsf{L} \setminus \neg\mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \mathfrak{f})\big) \cup \mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \mathfrak{f}), \rho\big)$$

is the only successor state of $(\mathfrak{t}, \mathsf{L}, \rho)$. $\mathcal{I} \models (\mathsf{prog} \sqsubseteq \mathit{Fail})$, $\mathcal{I}_0 \models (\mathsf{prog} \sqsubseteq \neg\mathit{Fail})$ and $\mathcal{I} = \mathcal{I}_0{}^{\mathsf{L}}$ imply $\langle \mathit{Fail}, \{\mathsf{prog}\}\rangle^+ \in \mathsf{L}$. It follows that

$$\mathsf{L} = \big(\mathsf{L} \setminus \neg\mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \mathfrak{f})\big) \cup \mathsf{Loc}_{\boldsymbol{B}}(\mathfrak{s}, \mathfrak{f}).$$

The transition $(\mathfrak{t}, \mathsf{L}, \rho) \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} (\mathfrak{t}, \mathsf{L}, \rho)$ is the only outgoing transition of $(\mathfrak{t}, \mathsf{L}, \rho)$. It is easy to see that

$$\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}_{\boldsymbol{B}}} (\mathfrak{t}, \mathsf{L}, \rho) \text{ implies } \langle \mathcal{I}, \sigma \cdot \mathfrak{f}, \rho \rangle \simeq_{\mathcal{C}_{\boldsymbol{B}}} (\mathfrak{t}, \mathsf{L}, \rho).$$

□

It remains to be shown that $\mathfrak{I}_{\mathcal{P}}$ and $\mathfrak{T}_{\mathcal{P}}$ are actually $\mathcal{C}_{\boldsymbol{B}}$-bisimilar. For this we have to prove that the respective initial states are related via $\simeq_{\mathcal{C}_{\boldsymbol{B}}}$.

Lemma 4.17. *$\mathfrak{T}_{\mathcal{P}}$ is a bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. $\mathcal{C}_{\boldsymbol{B}}$.*

Proof. We have shown that $\simeq_{\mathcal{C}_{\boldsymbol{B}}}$ is a $\mathcal{C}_{\boldsymbol{B}}$-bisimulation. For every initial state $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \in I_{\mathcal{P}}$ it holds that $(\text{d-type}^{\mathsf{loc}}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I}_0), \emptyset, \delta) \in I_{\mathfrak{T}_{\mathcal{P}}}$ and

$$\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \simeq_{\mathcal{C}_{\boldsymbol{B}}} (\text{d-type}^{\mathsf{loc}}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{I}_0), \emptyset, \delta).$$

And, for every initial state $(\mathfrak{t}, \emptyset, \delta) \in I_{\mathfrak{T}_{\mathcal{P}}}$ there is a model $\mathcal{J}_0$ of $\mathcal{K}$ with

$$\mathcal{J}_0 \models (\mathsf{prog} \sqsubseteq \neg\mathit{Final}) \wedge (\mathsf{prog} \sqsubseteq \neg\mathit{Fail})$$

such that $\text{d-type}^{\mathsf{loc}}_{\mathcal{C}_{\boldsymbol{B}}}(\mathcal{J}_0) = \mathfrak{t}$. It follows that $\langle \mathcal{J}_0, \langle\rangle, \delta \rangle \in I_{\mathcal{P}}$ and $\langle \mathcal{J}_0, \langle\rangle, \delta \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \emptyset, \delta)$. □

To decide whether an $\mathcal{ALCQIO}$-CTL* state formula Φ over axioms in $\mathcal{C}$ is valid in $\mathcal{P}$ it suffices to check whether $\mathfrak{T}_{\mathcal{P}}$ models $\iota(\Phi)$ which is a decidable model checking problem.

Theorem 4.18. *Verifying $\mathcal{ALCQIO}$-CTL* properties of $\mathcal{ALCQIO}$-ConGolog programs over local effect actions is decidable.*

Proof. The input program consists of a finite set of ground action terms $\boldsymbol{A}$, an $\mathcal{ALCQIO}$-admissible local effect representation $\mathsf{Loc}_{\boldsymbol{A}}$ and a program expression δ satisfying the restriction formulated in Definition 4.2. According to Lemma 4.15 and 4.17 a finite bisimilar propositional abstraction of the program is computable. Thus, the verification problem boils down to a decidable propositional CTL* model checking problem. □

The reduction to a propositional model checking also yields some complexity bounds. An upper bound on the size of $\mathfrak{T}_{\mathcal{P}}$ can be obtained. Let

$$n := |\mathcal{C}| + |\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})|.$$

The dynamic types are subsets of $\mathcal{C} \times 2^{\mathsf{Lit}(\mathsf{Loc}_{\boldsymbol{A}})}$ and it holds that

$$|\mathsf{D\text{-}Types}(\mathsf{Loc}_{\boldsymbol{A}})| \leq 2^{2^n}.$$

Consequently,

$$2^{2^n} \cdot 2^n \cdot 2^{|\delta|}$$

is an upper bound on the number of states in the abstract transition system.

The notion of an $\mathcal{ALCQIO}$-admissible local effect representation only requires that the effects of an action are effectively computable. To obtain complexity bounds for the verification problem we consider the case where action effects and preconditions are specified in a local effect $\mathcal{ALCQIO}$-action theory $\Sigma = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff})$ where $\boldsymbol{A}$ is the finite set of all relevant ground actions. The effect descriptions and preconditions of an action $\boldsymbol{\alpha} \in \boldsymbol{A}$ are directly given by $\mathsf{eff}(\boldsymbol{\alpha})$ and $\mathsf{pre}(\boldsymbol{\alpha})$, respectively. The *size of the input of the verification problem* is the sum of:

- the number of symbols needed to write down the initial KB $\mathcal{K}$,
- the number of symbols needed to write down the effect descriptions in $\bigcup_{\boldsymbol{\alpha}\in\boldsymbol{A}} \mathsf{eff}(\boldsymbol{\alpha})$ and the Boolean KBs in $\bigcup_{\boldsymbol{\alpha}\in\boldsymbol{A}} \mathsf{pre}(\boldsymbol{\alpha})$,
- the length of the program expression δ and the number symbols needed to write down the temporal property.

The natural numbers in at most and at least restriction are assumed to be encoded as binary strings.

Lemma 4.19. *Verifying $\mathcal{L}$-CTL* properties of an $\mathcal{L}$-ConGolog program over local effect actions that are specified in a local effect $\mathcal{L}$-action theory is decidable*

1. *in* 2ExpTime *if $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$, and*
2. *in* co-N2ExpTime *if $\mathcal{L} = \mathcal{ALCQIO}$.*

Proof. Let $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}, \mathcal{ALCQIO}\}$. The input for the verification problem consists of:

- a local effect $\mathcal{L}$-action theory $\Sigma = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff})$, where $\boldsymbol{A}$ is a finite set of ground actions
- a pick-free program expression over actions from $\boldsymbol{A}$, and with tests formulated as Boolean $\mathcal{L}$-KBs
- an $\mathcal{L}$-CTL* state formula Φ.

We assume that $\{\epsilon, \mathfrak{f}\} \subseteq \boldsymbol{A}$ and the preconditions and effects of the termination and failure action are defined as described above. And we assume that the literals $\mathsf{prog} \sqsubseteq \neg \mathit{Final}$ and $\mathsf{prog} \sqsubseteq \neg \mathit{Fail}$ are contained in $\mathcal{K}$. The relevant $\mathcal{L}$-context $\mathcal{C}$ consists of all axioms contained in $\mathcal{K}$, all effect conditions occurring in $\bigcup_{\boldsymbol{\alpha} \in \boldsymbol{A}} \mathsf{eff}(\boldsymbol{\alpha})$, all Boolean KBs in $\bigcup_{\boldsymbol{\alpha} \in \boldsymbol{A}} \mathsf{pre}(\boldsymbol{\alpha})$, all Boolean KBs occurring in the tests in δ and all axioms mentioned in Φ. The set of all relevant local effects $\mathsf{Lit}(\Sigma)$ is given by

$$\mathsf{Lit}(\Sigma) := \{\langle F, X\rangle^{\pm} \mid \boldsymbol{\alpha} \in \boldsymbol{A}, \psi \rhd \langle F, X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}) \text{ for some } \psi\}.$$

1. Assume $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$. We describe a decision procedure for deciding whether Φ is valid in $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$. First, the set $\mathsf{D\text{-}Types}(\Sigma, \mathcal{C})$ that consists of all complete and realizable subsets of $\mathcal{C} \times 2^{\mathsf{Lit}(\Sigma)}$ is computed. To do this we construct for each complete subset $\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathsf{Lit}(\Sigma)}$ that satisfies

   - $(\mathsf{prog} \sqsubseteq \neg \mathit{Final}, \emptyset) \in \mathfrak{t}$, $(\mathsf{prog} \sqsubseteq \neg \mathit{Fail}, \emptyset) \in \mathfrak{t}$, and
   - $(\varphi, \emptyset) \in \mathfrak{t}$ for all axioms φ in $\mathcal{K}$

   the reduction $\mathcal{L}$-KB $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ and check whether it is consistent. If it is consistent, then $\mathfrak{t}$ is included in $\mathsf{D\text{-}Types}(\Sigma, \mathcal{C})$. Each set $\mathfrak{t}$ is exponentially large in the size of the input. The corresponding KB $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ is of exponential size as well and and can be computed in exponential time. $\mathcal{L}$-KB consistency checking is in ExpTime measured w.r.t. the size of the KB. Thus, checking consistency of $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ is in 2ExpTime w.r.t. the size of the input. Since there are double exponentially many subsets of $\mathcal{C} \times 2^{\mathsf{Lit}(\Sigma)}$, the computation of $\mathsf{D\text{-}Types}(\Sigma, \mathcal{C})$ can be done with a double exponential time upper bound measured w.r.t. the size of the input. Next, we have to check whether

   $$\mathfrak{T}_{\mathcal{P}}, (\mathfrak{t}, \emptyset, \delta) \models \iota_{\mathcal{C}}(\Phi)$$

   holds for all $\mathfrak{t} \in \mathsf{D\text{-}Types}(\Sigma, \mathcal{C})$. It is sufficient to consider the fragment of $\mathfrak{T}_{\mathcal{P}}$ that is reachable from a given initial state $(\mathfrak{t}, \emptyset, \delta)$. This reachable fragment is denoted by $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}$ for a given $\mathfrak{t} \in \mathsf{D\text{-}Types}(\Sigma, \mathcal{C})$. To compute $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}$ it is sufficient to consider the states in

   $$\{\mathfrak{t}\} \times 2^{\mathsf{Lit}(\Sigma)} \times \mathsf{sub}(\delta).$$

   According to Theorem 3.15 the set $\mathsf{sub}(\delta)$ is at most exponentially large in $|\delta|$ and the size of the subprogram expressions is polynomial in the size of δ. Thus, the set $\{\mathfrak{t}\} \times 2^{\mathsf{Lit}(\Sigma)} \times \mathsf{sub}(\delta)$ contains at most exponentially many elements in the size of the input. The computation of the transition relation $\hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}$ on $\{\mathfrak{t}\} \times 2^{\mathsf{Lit}(\Sigma)} \times \mathsf{sub}(\delta)$ can be done in polynomial time in the size of $\{\mathfrak{t}\} \times 2^{\mathsf{Lit}(\Sigma)} \times \mathsf{sub}(\delta)$. Consequently, the computation of $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}$ requires exponential time in the size of the input and $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}$ is exponentially large. Furthermore, we have

$$\mathfrak{T}_{\mathcal{P}}, (\mathfrak{t}, \emptyset, \delta) \models \iota_{\mathcal{C}}(\Phi) \text{ iff } \mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}, (\mathfrak{t}, \emptyset, \delta) \models \iota_{\mathcal{C}}(\Phi).$$

According to Theorem 2.40 the check whether $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}, (\mathfrak{t}, \emptyset, \delta) \models \iota_{\mathcal{C}}(\Phi)$ holds can be done in exponential space and in double exponential time. Since $\mathsf{D\text{-}Types}(\Sigma, \mathcal{C})$ contains at most double exponentially many elements in the size of the input, one has to perform at most double exponentially many calls to a propositional CTL* model checker and each of these calls requires at most double exponential time. Consequently, the verification problem is decidable in 2EXPTIME.

2. It holds that Φ is valid in $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$ iff $\neg\Phi$ is *not* satisfiable in $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$. It follows that the complement problem of the satisfiability problem has the same complexity as the validity problem. We show that if $\mathcal{L} = \mathcal{ALCQIO}$, then the satisfiability problem is decidable in N2EXPTIME.

   A composition of three Turing machines T_1, T_2 and T_3 is constructed. T_1 is deterministic and computes a set of $\mathcal{ALCQIO}$-KBs and accepts. T_2 takes as input a set of $\mathcal{ALCQIO}$-KBs and guesses one of them and accepts. T_3 takes as input a single $\mathcal{ALCQIO}$-KB and checks its consistency. T_1 executes the following steps:

   - all subsets $\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathsf{Lit}(\Sigma)}$ with $(\mathsf{prog} \sqsubseteq \neg \mathit{Final}, \emptyset) \in \mathfrak{t}$, $(\mathsf{prog} \sqsubseteq \neg \mathit{Fail}, \emptyset) \in \mathfrak{t}$, and $(\varphi, \emptyset) \in \mathfrak{t}$ for all axioms φ in $\mathcal{K}$ are enumerated;
   - for all those sets $\mathfrak{t}$ the transition system $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}$ and the KB $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ is constructed;
   - $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}, (\mathfrak{t}, \emptyset, \delta) \models \iota_{\mathcal{C}}(\Phi)$ is checked for all $\mathfrak{t}$;
   - the output of T_1 is the set of all KBs $\mathcal{K}^{\mathfrak{t}}_{\mathsf{red}}$ where $\mathfrak{T}^{\mathfrak{t}}_{\mathcal{P}}, (\mathfrak{t}, \emptyset, \delta) \models \iota_{\mathcal{C}}(\Phi)$ is true.

   As argued above the steps can be done in 2EXPTIME. The nondeterministic machine T_2 that guesses one of the KBs of the output of T_1 gets an input of double exponential size and runs with a double exponential time upper bound. The nondeterministic machine T_3 that checks consistency of an $\mathcal{ALCQIO}$-KB runs in time exponential in the size of the KB. The input for T_3 is the output of T_2. A run of the composite machine consisting of T_1, T_2 and T_3 is accepting iff T_3 reaches an accepting state. It is a nondeterministic machine with a double exponential time upper bound.

□

4.4 Hardness of the Verification Problem

The question is whether the decision procedure described in the previous section is actually worst case optimal. To show hardness of the verification problem we reduce the consistency problem of *DL-KBs with nominal schemas* [Krö+11] to the verification problem for programs over local effect actions. A *nominal schema* is a non-ground nominal concept of the form $\{x\}$, where $x \in \mathsf{N_V}$ is a variable name. The variable names range over a given finite set of object names. A KB with nominal schemas is consistent iff the (exponentially large) set of all possible groundings is consistent.

Definition 4.20. Let $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}, \mathcal{ALCQIO}\}$ be a DL. An *$\mathcal{L}$-CI with nominal schemas* is an $\mathcal{L}$-CI of the form $C \sqsubseteq D$, where C and D are *non-ground* $\mathcal{L}$-concepts. An *$\mathcal{L}$-KB with nominal schemas* is a finite set of $\mathcal{L}$-CIs with nominal schemas.

Let $\mathcal{S}$ be an $\mathcal{L}$-KB with nominal schemas and Var the finite set of all variable names occurring in $\mathcal{S}$, and let Obj be a finite set of object names. We define the *set of all possible groundings of $\mathcal{S}$ w.r.t.* Obj as follows:

$$\mathsf{ground}(\mathcal{S}, \mathsf{Obj}) := \bigcup_{C \sqsubseteq D \in \mathcal{S}} \{C^\nu \sqsubseteq D^\nu \mid \nu \text{ with } \nu(x) \in \mathsf{Obj} \text{ for all } x \in \mathsf{Var}\}.$$

We say that $\mathcal{S}$ *is consistent w.r.t.* Obj iff the TBox $\mathsf{ground}(\mathcal{S}, \mathsf{Obj})$ is consistent. ▲

Example 4.21. Let x, y and z be variable names, Obj a finite set of object names and *ConTo* a role name. The CI with nominal schemas

$$\{x\} \sqcap \exists ConTo.(\{y\} \sqcap \exists ConTo.\{z\}) \sqsubseteq \{x\} \sqcap \exists ConTo.\{z\}$$

is consistent iff the restriction of *ConTo* to objects from Obj is transitive. Another example describes persons whose parents are married:

$$\exists HasFather.\{y\} \sqcap \exists HasMother.(\{z\} \sqcap \exists Married.\{y\}) \sqsubseteq PersonWithMarriedParents.$$

According to the semantics the variable names in nominal schemas are implicitly universally quantified over a given finite domain of object names. ▲

If the given set of object names has cardinality m, then a CI with n different variable names has m^n possible groundings. The set of all possible groundings of a KB with nominal schemas w.r.t. a finite set of object names can be exponentially large. Actually in [KR14] it is shown that nominal schemas cause an increase of the complexity of KB consistency by one exponential.

Theorem 4.22 ([KR14])**.** *Deciding consistency of an $\mathcal{L}$-KB with nominal schemas w.r.t. a finite set of object names is* 2ExpTime*-complete if $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$ and* N2ExpTime*-complete in case $\mathcal{L} = \mathcal{ALCQIO}$.*

To show hardness of the verification problem we reduce consistency of a KB with nominal schemas to the verification problem: for a given $\mathcal{L}$-KB $\mathcal{S}$ with nominal schemas and a finite set of object names Obj we construct a local effect $\mathcal{L}$-action theory

$$\Sigma_{\mathcal{S},\mathsf{Obj}} = (\mathcal{K}_{\mathcal{S},\mathsf{Obj}}, \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}, \mathsf{eff}, \mathsf{pre}),$$

a program expression $\delta_{\mathcal{S},\mathsf{Obj}}$ and a temporal property $\Phi_{\mathcal{S},\mathsf{Obj}}$ of polynomial size such that $\mathcal{S}$ is consistent w.r.t. Obj iff $\Phi_{\mathcal{S},\mathsf{Obj}}$ is satisfiable in $\mathcal{P} = (\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}), \delta_{\mathcal{S},\mathsf{Obj}})$. Let

$$\mathsf{Var} = \{x_1, \ldots, x_n\}$$

be the set of all variable names occurring in $\mathcal{S}$ and let

$$\mathsf{Obj} = \{o_1, \ldots, o_m\} \text{ for some } m \geq 1.$$

The idea is to construct a non-deterministic program that goes through all possible groundings of the variable names.

For the reduction we use the following additional fresh names not mentioned in $\mathcal{S}$:

- for each variable name x occurring in $\mathcal{S}$ we choose a fresh concept name $A[x]$, and
- a literal $s \mathrel{\varepsilon} \mathit{AllGrounded}$ to indicate that all variables are instantiated.

The extension of $A[x]$ in a particular state is supposed to contain exactly the object that instantiates the variable x. Next, we define $\Sigma_{\mathcal{S},\mathsf{Obj}} = (\mathcal{K}_{\mathcal{S},\mathsf{Obj}}, \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}, \mathsf{eff}, \mathsf{pre})$. The initial KB is given by

$$\begin{aligned}\mathcal{K}_{\mathcal{S},\mathsf{Obj}} = (\,&\mathcal{T} = \{A[x] \sqsubseteq \{o_1\} \sqcup \cdots \sqcup \{o_m\} \mid x \in \mathsf{Var}\},\\ &\mathcal{A} = \{s \mathrel{\varepsilon} \neg \mathit{AllGrounded}\}).\end{aligned}$$

We use the following set of ground action terms:

$$\boldsymbol{A}_{\mathcal{S},\mathsf{Obj}} = \{\mathtt{ground}_x(o) \mid x \in \mathsf{Var}, o \in \mathsf{Obj}\} \cup \{\mathtt{clear}(o_1,\dots,o_m)\} \cup \{\mathtt{finished}(s)\}.$$

The action $\mathtt{ground}_x(o)$ is used to instantiate the variable x with object o. For each pair $(x,o) \in \mathsf{Var} \times \mathsf{Obj}$ we define preconditions and effects as follows:

$$\mathsf{pre}(\mathtt{ground}_x(o)) = \emptyset, \qquad \mathsf{eff}(\mathtt{ground}_x(o)) = \big\{\langle A[x], \{o\}\rangle^+\big\}.$$

To instantiate the variable x with object o we add o to the extension of $A[x]$. The action $\mathtt{finished}(s)$ is used to indicate that all variables are instantiated:

$$\mathsf{pre}(\mathtt{finished}(s)) = \emptyset, \qquad \mathsf{eff}(\mathtt{finished}(s)) = \big\{\langle \mathit{AllGrounded}, \{s\}\rangle^+\big\}. \tag{4.7}$$

The action $\mathtt{clear}(o_1,\dots,o_m)$ resets the instantiation of the variables:

$$\begin{aligned}&\mathsf{pre}(\mathtt{clear}(o_1,\dots,o_m)) = \emptyset,\\ &\mathsf{eff}(\mathtt{clear}(o_1,\dots,o_m)) = \big\{\langle A[x], \{o\}\rangle^- \,\big|\, x \in \mathsf{Var}, o \in \mathsf{Obj}\big\} \cup \big\{\langle \mathit{AllGrounded}, \{s\}\rangle^-\big\}.\end{aligned}$$

For each variable name $x_i \in \mathsf{Var}$ the program expression δ^{x_i} nondeterministically chooses an instantiation:

$$\delta^{x_i} := (\mathtt{ground}_{x_i}(o_1) \mid \mathtt{ground}_{x_i}(o_2) \mid \cdots \mid \mathtt{ground}_{x_i}(o_m)).$$

The overall program expression that generates all possible groundings is given as follows:

$$\delta_{\mathcal{S},\mathsf{Obj}} := (\mathtt{clear}(o_1,\dots,o_m); \delta^{x_1}; \delta^{x_2}; \cdots; \delta^{x_n}; \mathtt{finished}(s))^*; \textsc{False}?.$$

For a CI $C \sqsubseteq D \in \mathcal{S}$ the CI $\widehat{C} \sqsubseteq \widehat{D}$ is obtained from $C \sqsubseteq D$ by simultaneously replacing all occurrences of the concepts $\{x\}$ with $x \in \mathsf{Var}$ in C and D by the corresponding concept name $A[x]$. The temporal property is defined as follows:

$$\Phi_{\mathcal{S},\mathsf{Obj}} := \mathsf{AG}\Bigg((s \mathrel{\varepsilon} \mathit{AllGrounded}) \rightarrow \Bigg(\bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D}\Bigg)\Bigg).$$

The size of $\Sigma_{\mathcal{S},\mathsf{Obj}}$, $\delta_{\mathcal{S},\mathsf{Obj}}$ and $\Phi_{\mathcal{S},\mathsf{Obj}}$ is quadratic in the size of $\mathcal{S}$ and Obj. The claim we prove in the following is that $\mathcal{S}$ is consistent w.r.t. Obj iff $\Phi_{\mathcal{S},\mathsf{Obj}}$ is satisfiable in the program $\mathcal{P} = (\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}), \delta_{\mathcal{S},\mathsf{Obj}})$. Obviously, if $\mathcal{P}$ is executed in models, where the concept names $A[x]$ with $x \in \mathsf{Var}$ are initially interpreted as empty sets, then whenever $s \sqsubseteq \mathit{AllGrounded}$ is true the concept names $A[x]$ are interpreted as singleton sets that contain objects from Obj.

Lemma 4.23. *Let $\mathcal{P} = (\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}), \delta_{\mathcal{S},\mathsf{Obj}})$ be as defined above and $\langle \mathcal{I}, \sigma, \rho \rangle$ a state in the transition system $\mathfrak{T}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ of $\mathcal{P}$ that satisfies the following conditions*

- *there exists an initial state $\langle \mathcal{I}_0, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}} \rangle \in I_{\mathcal{P}}$ with $\langle \mathcal{I}_0, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}} \rangle \hookrightarrow_{\mathcal{P}}^{*} \langle \mathcal{I}, \sigma, \rho \rangle$;*
- *$\mathcal{I} \models s \sqsubseteq \mathit{AllGrounded}$.*

It holds that for all $x \in \mathsf{Var}$ there exists an object name $o \in \mathsf{Obj}$ such that $(A[x])^{\mathcal{I}} = \{o^{\mathcal{I}}\}$.

Proof. Let $\langle \mathcal{I}, \sigma, \rho \rangle$ be a state in the transition system $\mathfrak{T}_{\mathcal{P}}$ satisfying the conditions described above. The definition of $\delta_{\mathcal{S},\mathsf{Obj}}$ ensures that there are no reachable final states. There are also no failure states because all actions in $\boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}$ are always possible. Let

$$\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}) = (\mathcal{F}, \mathcal{K}_{\mathcal{S},\mathsf{Obj}}, \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}, \mathcal{E}, \succ_{\mathsf{poss}})$$

be the FO-DS induced by $\Sigma_{\mathcal{S},\mathsf{Obj}}$. Since $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from an initial state there is a model $\mathcal{I}_0 \models \mathcal{K}_{\mathcal{S},\mathsf{Obj}}$ such that

$$\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\sigma} \mathcal{I}.$$

Since the ABox assertion $s \sqsubseteq \mathit{AllGrounded}$ is true in $\mathcal{I}$, the state $\langle \mathcal{I}, \sigma, \rho \rangle$ was reached by executing $\mathtt{finished}(s)$. It follows that σ is of the form

$$\sigma = \sigma' \; \mathtt{clear}(o_1, \ldots, o_m) \; \mathtt{ground}_{x_1}(o_{j_1}) \cdots \mathtt{ground}_{x_n}(o_{j_n}) \; \mathtt{finished}(s)$$

for some action sequence $\sigma' \in \boldsymbol{A}^{*}_{\mathcal{S},\mathsf{Obj}}$ and objects $o_{j_1}, \ldots, o_{j_n} \in \mathsf{Obj}$. Now, $\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\sigma} \mathcal{I}$ implies that there exists an interpretation $\mathcal{I}'$ satisfying the following conditions:

- $(A[x])^{\mathcal{I}'} = \emptyset$ for all $x \in \mathsf{Var}$;
- $\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\sigma' \mathtt{clear}(o_1, \ldots, o_m)} \mathcal{I}'$ and
- $\mathcal{I}' \Rightarrow_{\mathfrak{D}}^{\mathtt{ground}_{x_1}(o_{j_1})} \mathcal{I}'_1 \Rightarrow_{\mathfrak{D}}^{\mathtt{ground}_{x_2}(o_{j_2})} \cdots \Rightarrow_{\mathfrak{D}}^{\mathtt{ground}_{x_n}(o_{j_n})} \mathcal{I}'_n \Rightarrow_{\mathfrak{D}}^{\mathtt{finished}(s)} \mathcal{I}.$

It follows that

$$(A[x_i])^{\mathcal{I}} = \{o_{j_i}^{\mathcal{I}}\} \text{ for all } i = 1, \ldots, n.$$

□

We are now ready to prove that consistency of $\mathcal{S}$ w.r.t. Obj implies satisfiability of $\Phi_{\mathcal{S},\mathsf{Obj}}$ in $\mathcal{P}$.

Lemma 4.24. *If $\mathcal{S}$ is consistent w.r.t.* Obj*, then $\Phi_{\mathcal{S},\mathsf{Obj}}$ is satisfiable in $\mathcal{P} = (\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}), \delta_{\mathcal{S},\mathsf{Obj}})$.*

Proof. Assume that $\mathcal{S}$ is consistent w.r.t. Obj. Let

$$\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}) = (\mathcal{F}, \mathcal{K}_{\mathcal{S},\mathsf{Obj}}, \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}, \mathcal{E}, \succ_{\mathsf{poss}})$$

be the FO-DS induced by $\Sigma_{\mathcal{S},\mathsf{Obj}}$. The newly introduced concept names $A[x_1], \dots, A[x_n]$ and *AllGrounded* do not occur in $\mathcal{S}$. Therefore, there exists an interpretation $\mathcal{I}$ with

$$\mathcal{I} \models \mathsf{ground}(\mathcal{S}, \mathsf{Obj}) \text{ and } \mathcal{I} \models \mathcal{K}_{\mathcal{S},\mathsf{Obj}}.$$

Consequently, $\langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}} \rangle$ is an initial state in $\mathfrak{I}_{\mathcal{P}}$. We have to show that

$$\mathfrak{I}_{\mathcal{P}}, \langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}} \rangle \models \mathsf{AG}\big(s \sqsubseteq \mathit{AllGrounded} \rightarrow \big(\bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D} \big)\big).$$

Let $\pi \in \mathsf{paths}(\mathfrak{I}_{\mathcal{P}}, \langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}} \rangle)$ and $j \geq 0$ a natural number such that

$$\mathfrak{I}_{\mathcal{P}}, \pi[j] \models s \sqsubseteq \mathit{AllGrounded}.$$

Let $\pi[j] = \langle \mathcal{J}, \sigma, \rho \rangle$. It follows that $\mathcal{I} \Rightarrow_{\mathfrak{D}}^{\sigma} \mathcal{J}$. Since all the concept and role names occurring in $\mathcal{S}$ are not affected by the actions in $\boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}$, we have that $\mathcal{I} \models \mathsf{ground}(\mathcal{S}, \mathsf{Obj})$ implies also $\mathcal{J} \models \mathsf{ground}(\mathcal{S}, \mathsf{Obj})$. Furthermore, due to Lemma 4.23 it follows that there are object names $o_{j_1}, \dots, o_{j_n} \in \mathsf{Obj}$ such that

$$(A[x_i])^{\mathcal{J}} = \{o_{j_i}^{\mathcal{J}}\} \text{ for all } i = 1, \dots, n.$$

There exists a variable mapping ν_σ such that $\nu_\sigma(x_i) = o_{j_i}$ for all $i = 1, \dots, n$. With $\mathcal{J} \models \mathsf{ground}(\mathcal{S}, \mathsf{Obj})$ it follows that

$$\mathcal{J} \models \bigwedge_{C \sqsubseteq D \in \mathcal{S}} C^{\nu_\sigma} \sqsubseteq D^{\nu_\sigma}.$$

Due to $(A[x_i])^{\mathcal{J}} = \{\nu_\sigma(x_i)\}^{\mathcal{J}}$ for all $i = 1, \dots, n$ it follows that

$$\mathcal{J} \models \bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D}.$$

Consequently, we have

$$\mathfrak{I}_{\mathcal{P}}, \pi[j] \models s \sqsubseteq \mathit{AllGrounded} \rightarrow \big(\bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D} \big).$$

Note that $\pi \in \mathsf{paths}(\mathfrak{I}_{\mathcal{P}}, \langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}} \rangle)$ and $j \geq 0$ are arbitrarily chosen. The claim follows directly. □

The proof of the other direction is also straightforward.

Lemma 4.25. *If $\Phi_{\mathcal{S},\mathsf{Obj}}$ is satisfiable in $\mathcal{P} = (\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}), \delta_{\mathcal{S},\mathsf{Obj}})$, then $\mathcal{S}$ is consistent w.r.t.* Obj.

Proof. Let

$$\mathfrak{D}(\Sigma_{\mathcal{S},\mathsf{Obj}}) = (\mathcal{F}, \mathcal{K}_{\mathcal{S},\mathsf{Obj}}, \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}, \mathcal{E}, \succ_{\mathsf{poss}})$$

be the FO-DS induced by $\Sigma_{\mathcal{S},\mathsf{Obj}}$. Assume there is an initial state $\langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}}\rangle$ in $\mathfrak{T}_{\mathcal{P}}$ such that $\mathfrak{T}_{\mathcal{P}}, \langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}}\rangle \models \Phi_{\mathcal{S},\mathsf{Obj}}$. We show that for an arbitrary variable mapping ν with $\nu(x_i) \in \mathsf{Obj}$ for all $i = 1, \dots, n$ it holds that

$$\mathcal{I} \models \bigwedge_{C \sqsubseteq D \in \mathcal{S}} C^{\nu} \sqsubseteq D^{\nu}.$$

This implies $\mathcal{I} \models \mathsf{ground}(\mathcal{S}, \mathsf{Obj})$. For an arbitrary but fixed ν with $\nu(x_i) \in \mathsf{Obj}$ for all $i = 1, \dots, n$ we consider the following action sequence

$$\sigma^{\nu} = \mathtt{clear}(o_1, \dots, o_m); \mathtt{ground}_{x_1}(\nu(x_1)); \cdots ; \mathtt{ground}_{x_n}(\nu(x_n)); \mathtt{finished}(s)$$

Obviously, σ^{ν} is executable in $\mathcal{I}$ and is admitted in $\delta_{\mathcal{S},\mathsf{Obj}}$. There exists an interpretation $\mathcal{J}^{\nu}$ with $\mathcal{I} \Rightarrow_{\mathfrak{D}}^{\sigma^{\nu}} \mathcal{J}^{\nu}$ and subprogram $\rho \in \mathsf{sub}(\delta_{\mathcal{S},\mathsf{Obj}})$ such that $\langle \mathcal{J}^{\nu}, \sigma^{\nu}, \rho\rangle$ is a state in $\mathfrak{T}_{\mathcal{P}}$ and is reachable from $\langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}}\rangle$. Initially, we have

$$\mathcal{I} \models \{A[x] \sqsubseteq \{o_1\} \sqcup \cdots \sqcup \{o_m\} \mid x \in \mathsf{Var}\}.$$

It follows that after executing σ in $\mathcal{I}$ the resulting interpretation $\mathcal{J}^{\nu}$ satisfies:

$$(A[x_i])^{\mathcal{J}^{\nu}} = \{\nu(x_i)^{\mathcal{J}^{\nu}}\} \text{ for all } i = 1, \dots, n \text{ and } \mathcal{J}^{\nu} \models s \in \mathit{AllGrounded}.$$

The assumption $\mathfrak{T}_{\mathcal{P}}, \langle \mathcal{I}, \langle\rangle, \delta_{\mathcal{S},\mathsf{Obj}}\rangle \models \Phi_{\mathcal{S},\mathsf{Obj}}$ implies that

$$\mathcal{J}^{\nu} \models \bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D}.$$

With $(A[x_i])^{\mathcal{J}^{\nu}} = \{\nu(x_i)^{\mathcal{J}^{\nu}}\}$ for all $i = 1, \dots, n$ it follows that

$$\mathcal{J}^{\nu} \models \bigwedge_{C \sqsubseteq D \in \mathcal{S}} C^{\nu} \sqsubseteq D^{\nu}.$$

For all concept, role and object names X mentioned in $\mathcal{S}$ it holds that $X^{\mathcal{J}^{\nu}} = X^{\mathcal{I}}$, because the execution of σ^{ν} in $\mathcal{I}$ changes only the interpretation of the names $A[x_1], \dots, A[x_n]$ and *AllGrounded* that are not occurring in $\mathcal{S}$. It follows that

$$\mathcal{I} \models \bigwedge_{C \sqsubseteq D \in \mathcal{S}} C^{\nu} \sqsubseteq D^{\nu}.$$

Since the grounding ν with $\nu(x_i) \in \mathsf{Obj}$ for all $i = 1, \dots, n$ was arbitrarily chosen, we obtain $\mathcal{I} \models \mathsf{ground}(\mathcal{S}, \mathsf{Obj})$. □

The reduction of reasoning with nominal schemas to the satisfiability problem leads to the matching lower bounds for the verification problem.

Theorem 4.26. *Verifying $\mathcal{L}$-CTL* properties of an $\mathcal{L}$-ConGolog program over local effect actions that are specified in an local effect $\mathcal{L}$-action theory is*

1. 2ExpTime-*complete if $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$, and*
2. co-N2ExpTime-*complete if $\mathcal{L} = \mathcal{ALCQIO}$.*

Proof. Checking whether an $\mathcal{L}$-CTL* state formula Φ is satisfiable in an $\mathcal{L}$-ConGolog program over local effect actions of the form $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$, where Σ is a local effect $\mathcal{L}$-action theory is 2ExpTime-hard if $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$, and N2ExpTime-hard if $\mathcal{L} = \mathcal{ALCQIO}$. This is a consequence of Lemma 4.24 and 4.25 and Theorem 4.22. It holds that Φ is valid in $\mathcal{P}$ iff $\neg\Phi$ is not satisfiable. Therefore, the validity problem has the same complexity as the complement problem of the satisfiability problem. Thus, validity is 2ExpTime-hard if $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$, and co-N2ExpTime-hard if $\mathcal{L} = \mathcal{ALCQIO}$. Together with Lemma 4.19 we obtain completeness for the respective complexity classes. $\square$

The hardness result already holds for $\mathcal{L}$-CTL properties, and in case of unconditional actions, where executability and action effects do not depend on the context.

4.5 Summary

In this chapter, we have obtained a fragment of ConGolog programs and temporal properties for which the verification problem is decidable. In this fragment, we have restricted the base logic to a DL $\mathcal{L}$ ranging from $\mathcal{ALCO}$ to $\mathcal{ALCQIO}$, primitive actions are restricted to local effect actions, the pick-operator is dropped and as the verification logic we have used CTL* over $\mathcal{L}$-axioms. Decidability was shown by introducing an abstraction technique that allows us to reduce the verification problem to a decidable model checking problem. Moreover, we have obtained tight complexity results in case local effect $\mathcal{L}$-action theories are used for representing the underlying first-order dynamical system. The complexity upper bounds we have obtained are the same as for the verification problems studied in [BLM10] and [Lip14]. However, in [BLM10] and [Lip14] no matching lower bounds are given.

Chapter 5

Limits of Decidable Verification with Non-Local Effect Actions

The goal in this chapter is to further push the decidability border for the verification problem towards programs over actions with non-local effects. The focus is on DL-ConGolog programs (see Definition 3.16), i.e. pick-free programs over a finite set of DL-definable ground actions, and DL-CTL* specifications.

We have seen in the previous chapter that restricting action effects to local effects leads to a decidable fragment of DL-ConGolog. Allowing only actions with local effects is a quite strong restriction and leads to a domain where only a fixed finite set of named objects is affected by action executions. In some domains it is more natural to also take non-local effects of actions into account, especially under the open-world assumption. We call an effect *non-local* if the changes affect also unnamed objects. For instance in a transportation domain the action of moving a box from one location to another location affects not only the box itself but also the (unboundedly many, unmentioned) items that are currently contained in the box. In a DL-action theory it is possible to define actions with non-local effects (see Example 2.32). However, if we use such actions inside a loop of a program the verification problem easily becomes undecidable (Section 5.1). Thus, we observe that the restriction to pick-free programs over a finite set of DL-definable ground actions does not guarantee decidability. To investigate the sources of undecidability in more detail we first generalize our notion of dynamic types from local effect actions to ground actions defined in an arbitrary DL-admissible representation of a dynamical system (Section 5.2). In presence of non-local effects there are in general infinitely many dynamic types. In Section 5.3 we define two incomparable syntactic restrictions on the effect representation of actions that allow us to partition the state space into sets of interpretations of only finitely many dynamic types. In these restricted representations actions still might have non-local effects but we can show that verifying DL-ConGolog programs based on these restricted DL-admissible representations is decidable via a reduction to propositional model checking. In Section 5.4, we summarize the results and briefly review related work.

5.1 Undecidability due to Non-Local Effects

In this section, we consider the verification problem with an input that consists of the following components:

- an $\mathcal{L}$-action theory $\Sigma = (\mathcal{K}, \boldsymbol{A}, \mathsf{eff}, \mathsf{pre})$, where $\boldsymbol{A}$ is a finite set of ground action terms;

$$\begin{tikzcd}
\mathsf{home}^{\mathcal{I}} \arrow[r, "\mathit{Left}^{\mathcal{I}}"] & r_1 \arrow[r, "\mathit{Left}^{\mathcal{I}}"] & r_2 \arrow[r, "\mathit{Left}^{\mathcal{I}}"] & r_3 \\
\in & \in & \in & \in \\
(\mathit{CurrentRoom})^{\mathcal{I}} & (\mathit{Room})^{\mathcal{I}}, & (\mathit{Room})^{\mathcal{I}} & (\mathit{Room})^{\mathcal{I}} \\
 & (\exists \mathit{Right}.\mathit{CurrentRoom})^{\mathcal{I}} & &
\end{tikzcd}$$

Figure 5.1: Example model with rooms

- a pick-free program expression δ, where all actions terms in δ are contained in $\boldsymbol{A}$ and all tests in δ consists of ground Boolean $\mathcal{L}$-KBs;
- an $\mathcal{L}$-CTL* state formula Φ.

We assume $\mathcal{L} \subseteq \mathcal{DL}$. Note that the programs $\mathcal{P} = (\mathfrak{D}(\Sigma), \delta)$ described above are $\mathcal{L}$-ConGolog programs (according to Definition 3.16). The only difference compared to the setting in the previous chapter is that now the actions in the set $\boldsymbol{A}$ are allowed to have non-local effects. The add-sets and delete-sets of actions may contain also an unbounded number of objects. Even though in this restricted setting the domain only consists of finitely many ground actions, DL-ConGolog programs are still Turing complete. The verification problem is undecidable. Before we present the proof, we consider an example to illustrate the expressive power of programs over actions with non-local effects.

Example 5.1. We model a domain of a mobile robot moving along a corridor of rooms. Initially only the starting point is named in the knowledge base. The robot can move in one step to the room that is left or right adjacent to the room the robot is currently located in. Furthermore, the robot can change its direction. Some static global properties of the relevant fluents are described using a TBox with the following axioms:

$$\mathit{Left} \equiv \mathsf{inv}(\mathit{Right}) \tag{5.1}$$

$$\mathit{Room} \sqsubseteq {\leq}1\, \mathit{Right}.\mathit{Room} \sqcap {\leq}1\, \mathit{Left}.\mathit{Room} \tag{5.2}$$

$$\mathit{CurrentRoom} \sqsubseteq \mathit{Room} \tag{5.3}$$

$$\top \sqsubseteq {\leq}1\, U.\mathit{CurrentRoom} \sqcap {\leq}1\, U.\mathit{Dir} \tag{5.4}$$

$$\mathit{Dir} \sqsubseteq \{\mathsf{left}\} \sqcup \{\mathsf{right}\} \tag{5.5}$$

$$\{\mathsf{home}\} \sqsubseteq \mathit{Room} \sqcap \exists \mathit{Left}.\mathit{Room} \tag{5.6}$$

$$\mathit{Room} \sqcap \exists \mathit{Left}.\{\mathsf{home}\} \sqsubseteq \bot \tag{5.7}$$

Right and *Left* are role names representing the adjacency relation among rooms. According to (5.2) there is at most one room to the right of each room and at most one to the left. The concept name *CurrentRoom* is supposed to describe the room the robot is currently located in and *Dir* captures the direction the robot is currently facing to. The CI (5.4) ensures that both names are interpreted as singleton sets. Note that U is the universal role. The robot is either facing to the left or to the right according to (5.5). There is a room named home that is located to the left of some room (5.6). The inclusion (5.7) says that there is no room to the left of home. In other words, home is the leftmost room on the corridor. Initially, the robot is at home and is facing to the right. The ABox of the initial KB consists of the following ABox literals:

$$(\mathsf{home} \in \mathit{CurrentRoom}), (\mathsf{right} \in \mathit{Dir}).$$

There are two ground actions denoted by `move-fwd` and `turn`. Consider an interpretation $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ that is a model of the initial KB and satisfies

$$\begin{aligned}&\mathit{CurrentRoom}^{\mathcal{I}} = \{\mathsf{home}^{\mathcal{I}}\},\\ &\mathit{Dir}^{\mathcal{I}} = \{\mathsf{right}^{\mathcal{I}}\} \text{ and}\\ &(r_1, \mathsf{home}^{\mathcal{I}}) \in \mathit{Right}^{\mathcal{I}} \text{ for some } r_1 \in \mathit{Room}^{\mathcal{I}}.\end{aligned}$$

The domain element r_1 represents the room that is located to the right of home in $\mathcal{I}$ and home is to the left of r_1. Figure 5.1 shows a sketch of $\mathcal{I}$. Since $\mathcal{I}$ satisfies the CI (5.2), we have

$$(\mathit{Room} \sqcap \exists \mathit{Right}.\mathit{CurrentRoom})^{\mathcal{I}} = \{r_1\}.$$

The robot is currently in home and is facing into the direction of the room r_1. The execution of `move-fwd` in $\mathcal{I}$ should result in an interpretation $\mathcal{J}$, where

$$\mathit{CurrentRoom}^{\mathcal{J}} = \{r_1\}$$

is true. The effects of `move-fwd` are therefore defined as follows:

$$\begin{aligned}\mathsf{eff}(\texttt{move-fwd}) := \{\, &\langle \mathit{CurrentRoom}, \mathit{CurrentRoom}\rangle^-,\\ &(\mathsf{right} \in \mathit{Dir}) \rhd \langle \mathit{CurrentRoom}, \mathit{Room} \sqcap \exists \mathit{Right}.\mathit{CurrentRoom}\rangle^+,\\ &(\mathsf{left} \in \mathit{Dir}) \rhd \langle \mathit{CurrentRoom}, \mathit{Room} \sqcap \exists \mathit{Left}.\mathit{CurrentRoom}\rangle^+\}.\end{aligned}$$

Moving forward to the next room is only possible if there actually is a room next to the current one. We have $\mathsf{pre}(\texttt{move-fwd}) := \{\psi_{\mathsf{fwd}}\}$, where ψ_{fwd} is a Boolean KB given by

$$\begin{aligned}\psi_{\mathsf{fwd}} := \;&(\mathsf{right} \in \mathit{Dir}) \rightarrow (\mathit{CurrentRoom} \sqsubseteq \exists \mathit{Left}.\mathit{Room}) \wedge\\ &(\mathsf{left} \in \mathit{Dir}) \rightarrow (\mathit{CurrentRoom} \sqsubseteq \exists \mathit{Right}.\mathit{Room}).\end{aligned}$$

We assume that making turns is always possible: $\mathsf{pre}(\texttt{turn}) := \emptyset$. The effects are given as follows

$$\begin{aligned}\mathsf{eff}(\texttt{turn}) := \{\, &(\mathsf{right} \in \mathit{Dir}) \rhd \langle \mathit{Dir}, \{\mathsf{left}\}\rangle^+, (\mathsf{right} \in \mathit{Dir}) \rhd \langle \mathit{Dir}, \{\mathsf{right}\}\rangle^-,\\ &(\mathsf{left} \in \mathit{Dir}) \rhd \langle \mathit{Dir}, \{\mathsf{right}\}\rangle^+, (\mathsf{left} \in \mathit{Dir}) \rhd \langle \mathit{Dir}, \{\mathsf{left}\}\rangle^-\}.\end{aligned}$$

We can now write a control program for a robot that moves back and forth and visits all rooms along the corridor:

$$\textbf{while } \textsc{True} \textbf{ do } (\neg\psi_{\mathsf{fwd}}?; \texttt{turn})^*; \psi_{\mathsf{fwd}}?; \texttt{move-fwd} \textbf{ end}.$$

▲

The idea we have used to implement the moves of the robot in the example above can be reused to simulate a two-counter machine. In the TBox a (static) chain of unbounded length with a fixed origin is axiomatized. The origin represents the zero value. For each counter a concept name is introduced, and pushing these concept names back and forth in the chain corresponds to a decrement and increment operation, respectively. Undecidability is shown by reducing the halting problem of two-counter machines to the verification problem.

Theorem 5.2. *Verifying $\mathcal{ALCQIO}$-CTL* properties of $\mathcal{ALCQIO}$-ConGolog programs based on $\mathcal{ALCQIO}$-action theories is undecidable.*

Proof. For the reduction we use the same machine model as in Definition 3.3. We formulate a program simulating a two-counter machine with two counters c_0 and c_1. The program of the counter machine consists a list of instructions given by $\mathsf{M} = \mathsf{J}_0; \cdots ; \mathsf{J}_m$ as described in Def. 3.3.

The following concept, role and object names are introduced:

- two concept names C_0 and C_1, one for each counter;
- concept names *Halt* and $J_0, \ldots, J_m$ (one name for each instruction);
- a role name *Succ* and object names $\mathbf{0}$ and p.
- We use the name *Pred* to abbreviate the role $\mathsf{inv}(Succ)$, i.e. the inverse of *Succ*.

To represent the values of the counters in an interpretation we define an infinite chain of objects starting in $\mathbf{0}$ using the role name *Succ*. We ensure that in each interpretation $\mathcal{I}$, $d \in C_\ell^{\mathcal{I}}$ with $\ell = 0, 1$ is true for exactly one domain element d in this chain. The distance of this element d from $\mathbf{0}$ in the *Succ*-chain represents the value of the counter c_ℓ.

M is in a halting configuration if $p \in Halt$ is true and J_i is the currently executed instruction iff the corresponding literal $p \in J_i$ is true.

An $\mathcal{ALCQIO}$-action theory Σ_M is defined. The initial KB is denoted by $\mathcal{K}_\mathsf{M} = (\mathcal{T}_\mathsf{M}, \mathcal{A}_\mathsf{M})$. The ABox for describing the initial situation is given as follows:

$$\mathcal{A}_\mathsf{M} = \{p \in (\neg Halt \sqcap J_0 \sqcap \neg J_1 \sqcap \cdots \sqcap \neg J_m)\} \tag{5.8}$$

The machine is not in its halting state and J_0 is the first instruction to be executed. In the TBox an infinite *Succ*-chain is defined:

$$\begin{aligned} \mathcal{T}_\mathsf{M} = \{ \quad \top &\sqsubseteq {\leq}1\, Succ.\neg\{\mathbf{0}\} \sqcap {\geq}1\, Succ.\neg\{\mathbf{0}\}, \\ \neg\{\mathbf{0}\} &\sqsubseteq {\leq}1\, Pred.\top \sqcap {\geq}1\, Pred.\top\}. \end{aligned}$$

Thus, each element has exactly one successor (via the role *Succ*) that is not equal to $\mathbf{0}$ and each element except for $\mathbf{0}$ has exactly one predecessor (with role *Pred* referring to the inverse of *Succ*). The concepts in the sequence

$$\{\mathbf{0}\}, \exists Pred.\{\mathbf{0}\}, \exists Pred.(\exists Pred.\{\mathbf{0}\}), \ldots, \tag{5.9}$$

where $Pred := \mathsf{inv}(Succ)$, are interpreted as pairwise disjoint singleton sets in all models of $\mathcal{T}_\mathsf{M}$. The sequence of their extensions under a model of $\mathcal{T}_\mathsf{M}$ represents the natural numbers $0, 1, 2, \ldots$. This structure is enforced by $\mathcal{T}_\mathsf{M}$ and will remain unchanged. The extension of the role *Succ* won't be affected by any action.

There is one action term for each instruction J_i of M and two actions for initializing the two counters. The finite set of relevant ground action terms is given by

$$\mathsf{Act}_\mathsf{M} := \{\mathtt{inst}_0(\mathbf{0},p),\dots,\mathtt{inst}_m(\mathbf{0},p),\mathtt{init}_1,\mathtt{init}_2(\mathbf{0})\}.$$

The two initialization actions have the following preconditions and effects:

$$\begin{aligned}
&\mathsf{pre}(\mathtt{init}_1) := \emptyset, && \mathsf{eff}(\mathtt{init}_1) := \{\langle C_0,\top\rangle^-,\langle C_1,\top\rangle^-\},\\
&\mathsf{pre}(\mathtt{init}_2(\mathbf{0})) := \emptyset, && \mathsf{eff}(\mathtt{init}_2(\mathbf{0})) := \{\langle C_0,\{\mathbf{0}\}\rangle^+,\langle C_1,\{\mathbf{0}\}\rangle^+\}.
\end{aligned}$$

We make sure that the extensions of C_0 and C_1 are singleton sets containing $\mathbf{0}$.

The kth instruction is possible iff the literal $p \sqsubseteq J_k$ is true. For all $k \in \{1,\dots,m\}$ we have

$$\mathsf{pre}(\mathtt{inst}_k(\mathbf{0},p)) := \{p \sqsubseteq J_k\}.$$

If the kth instruction J_k of M is an increment instruction of the form $\mathsf{Inc}(\ell,i)$ with $k,i \in \{0,\dots,m\}$ and $\ell \in \{0,1\}$, then the effects of $\mathtt{inst}_k(\mathbf{0},p)$ are defined as follows:

$$\begin{aligned}
\mathsf{eff}(\mathtt{inst}_k(\mathbf{0},p)) := \{&\langle C_\ell,\exists \mathit{Pred}.C_\ell\rangle^+,\langle C_\ell,C_\ell\rangle^-,\\
&\langle J_k,\{p\}\rangle^-,\langle J_i,\{p\}\rangle^+\}.
\end{aligned} \tag{5.10}$$

The concept $\exists \mathit{Pred}.C_\ell$ refers to the element in the *Succ*-chain whose predecessor in this chain is the current instance of C_ℓ. This instance of $\exists \mathit{Pred}.C_\ell$ is added to C_ℓ and the old instance of C_ℓ is deleted. Intuitively, C_ℓ is shifted to the right in the *Succ*-chain. With the last two (local) effects the program counter is set to the subsequent ith instruction.

If the kth instruction is a decrement of the form $\mathsf{Dec}(\ell,i,j)$ with $i,j \in \{0,\dots,m\}$ and $\ell = \{0,1\}$, then the conditional effects are defined as follows:

$$\begin{aligned}
\mathsf{eff}(\mathtt{inst}_k(\mathbf{0},p)) := \{&(\mathbf{0} \sqsubseteq \neg C_\ell) \rhd \langle C_\ell,\exists \mathit{Succ}.C_\ell\rangle^+,\\
&(\mathbf{0} \sqsubseteq \neg C_\ell) \rhd \langle C_\ell,C_\ell\rangle^-,\\
&\langle J_k,\{p\}\rangle^-,\\
&(\mathbf{0} \sqsubseteq C_\ell) \rhd \langle J_i,\{p\}\rangle^+,\\
&(\mathbf{0} \sqsubseteq \neg C_\ell) \rhd \left\langle J_j,\{p\}\right\rangle^+\}.
\end{aligned} \tag{5.11}$$

The condition $(\mathbf{0} \sqsubseteq \neg C_\ell)$ indicates that the value of the counter c_ℓ is greater than zero. The decrement is realized by moving C_ℓ to the left in the *Succ*-chain. The new instance of C_ℓ is exactly the element whose successor is in C_ℓ and the old instance of C_ℓ is deleted from C_ℓ. In case the value of the ℓth counter is zero the program counter is set to the ith instruction and nothing else is changed. Otherwise, the program counter is set to the jth instruction.

The effects of $\mathtt{inst}_k(\mathbf{0},p)$ in case J_k is the halting instruction are given by

$$\mathsf{eff}(\mathtt{inst}_k(\mathbf{0},p)) := \{\langle \mathit{Halt},\{p\}\rangle^+\}. \tag{5.12}$$

The $\mathcal{ALCQIO}$-ConGolog program $\mathcal{P}_\mathsf{M} = (\mathfrak{D}(\Sigma_\mathsf{M}),\delta_\mathsf{M})$ consists of the FO-DS $\mathfrak{D}(\Sigma_\mathsf{M})$ induced by Σ_M and the following program expression with one while-loop and cascading if-then-else statements:

$$\begin{aligned}\delta_{\mathsf{M}} := {} & \texttt{init}_1;\texttt{init}_2(\mathbf{0}); \\ & \textbf{while } (p \sqsubseteq \neg \mathit{Halt}) \textbf{ do} \\ & \quad ((p \sqsubseteq J_0)?;\texttt{inst}_0(\mathbf{0},p);) \mid \cdots \mid ((p \sqsubseteq J_m)?;\texttt{inst}_m(\mathbf{0},p)) \\ & \textbf{end}.\end{aligned} \tag{5.13}$$

Validity of the following state formula in $\mathcal{P}_{\mathsf{M}}$ says that M eventually halts on all execution paths:

$$\Phi_{\mathsf{M}} := \mathsf{AF}(p \sqsubseteq \mathit{Halt}).$$

It has to be shown that Φ_{M} is valid in $\mathcal{P}_{\mathsf{M}}$ iff M halts. We only give an outline of the proof:

First of all, we observe that each state in the transition system of $\mathcal{P}_{\mathsf{M}}$ has exactly one unique successor state. The program is deterministic. Next, we define a function that maps each reachable state (after initialization) in the transition system of $\mathcal{P}_{\mathsf{M}}$ to a configuration of M. To abbreviate the concepts in the sequence (5.9) we use the following notation:

$$\begin{aligned}&\exists \mathit{Pred}^0.\{\mathbf{0}\} := \{\mathbf{0}\} \\ &\exists \mathit{Pred}^n.\{\mathbf{0}\} := \exists \mathit{Pred}.(\exists \mathit{Pred}^{n-1}.\{\mathbf{0}\}) \text{ for all } n \geq 1.\end{aligned} \tag{5.14}$$

Let

$$\mathfrak{I}_{\mathsf{M}} = (Q_{\mathsf{M}}, I_{\mathsf{M}}, \hookrightarrow_{\mathsf{M}}, \lambda_{\mathsf{M}})$$

be the transition system induced by $\mathcal{P}_{\mathsf{M}}$. With $\overline{Q_{\mathsf{M}}}$ we denote the set of all states $\langle \mathcal{J}, \sigma, \rho \rangle \in Q_{\mathsf{M}}$ satisfying the following properties:

- $\sigma = \texttt{init}_1\texttt{init}_2(\mathbf{0})\sigma'$ for some sequence $\sigma' \in \mathsf{Act_M}^*$, and
- $\langle \mathcal{I}, \langle\rangle, \delta_{\mathsf{M}} \rangle \hookrightarrow_{\mathsf{M}}{}^* \langle \mathcal{J}, \sigma, \rho \rangle$ for some initial state $\langle \mathcal{I}, \langle\rangle, \delta_{\mathsf{M}} \rangle \in I_{\mathsf{M}}$.

There exists a function conf that maps those states to configurations of M with the following property: For all states $\langle \mathcal{J}, \sigma, \rho \rangle \in \overline{Q_{\mathsf{M}}}$ it holds that

$$\begin{aligned}\mathsf{conf}(\langle \mathcal{J}, \sigma, \rho \rangle) = (i, n_0, n_1) \text{ iff } & \mathcal{J} \models (p \sqsubseteq J_i) \wedge \\ & (C_0)^{\mathcal{J}} = (\exists \mathit{Pred}^{n_0}.\{\mathbf{0}\})^{\mathcal{J}} \wedge \\ & (C_1)^{\mathcal{J}} = (\exists \mathit{Pred}^{n_1}.\{\mathbf{0}\})^{\mathcal{J}}.\end{aligned}$$

By induction on the length of σ it can be shown that such a unique function conf exists. Next, we can show that for all states $\langle \mathcal{J}, \sigma, \rho \rangle \in \overline{Q_{\mathsf{M}}}$ with $\mathsf{conf}(\langle \mathcal{J}, \sigma, \rho \rangle) = (i, n_0, n_1)$ the following holds:

- If $\mathsf{J}_i \neq \mathsf{Halt}$, then there exists exactly one successor state $\langle \mathcal{J}', \sigma', \rho' \rangle \in Q_{\mathsf{M}}$ with

  $$\langle \mathcal{J}, \sigma, \rho \rangle \hookrightarrow_{\mathsf{M}} \langle \mathcal{J}', \sigma', \rho' \rangle,$$

  and $\mathsf{conf}(\langle \mathcal{J}, \sigma, \rho \rangle) \vdash_{\mathsf{M}} \mathsf{conf}(\langle \mathcal{J}', \sigma', \rho' \rangle)$.
- Otherwise, if $\mathsf{J}_i = \mathsf{Halt}$, then either $\mathcal{J} \models p \sqsubseteq \mathit{Halt}$ or there exists exactly one successor state $\langle \mathcal{J}', \sigma', \rho' \rangle$ with $\mathcal{J}' \models p \sqsubseteq \mathit{Halt}$.

One also has to show that each execution of M leads to a corresponding path in the transition system of $\mathcal{P}_\mathsf{M}$. For each execution of M

$$(i_0, n_0^0, n_1^0) \vdash_\mathsf{M} (i_1, n_0^1, n_1^1) \vdash_\mathsf{M} \cdots \vdash_\mathsf{M} (i_k, n_0^k, n_1^k)$$

with $(i_0, n_0^0, n_1^0) = (0,0,0)$. and all initial states $\langle \mathcal{I}, \langle\rangle, \delta_\mathsf{M}\rangle \in I_\mathsf{M}$ there exists an initial path fragment

$$\langle \mathcal{I}_0, \sigma_0, \delta_0\rangle \hookrightarrow_\mathsf{M} \langle \mathcal{I}_1, \sigma_1, \delta_1\rangle \hookrightarrow_\mathsf{M} \cdots \hookrightarrow_\mathsf{M} \langle \mathcal{I}_{k+2}, \sigma_{k+2}, \delta_{k+2}\rangle$$

such that $\langle \mathcal{I}_0, \sigma_0, \delta_0\rangle = \langle \mathcal{I}, \langle\rangle, \delta_\mathsf{M}\rangle$, $\sigma_1 = \mathtt{init}_1$ and $\sigma_2 = \mathtt{init}_1\mathtt{init}_2(\mathbf{0})$ and

$$\mathsf{conf}(\langle \mathcal{I}_{z+2}, \sigma_{z+2}, \delta_{z+2}\rangle) = (i_z, n_0^z, n_1^z) \text{ for all } z = 0, \ldots, k.$$

Thus, for each execution of M there is matching execution of $\mathcal{P}_\mathsf{M}$ and vice versa. It follows that Φ_M is valid in $\mathcal{P}_\mathsf{M}$ iff M halts. □

It is possible to strengthen the undecidability result by further restricting the expressive power of the base logic to the DL $\mathcal{ELI}_\bot$ that does not offer number restrictions, nominals, negation and disjunction. *$\mathcal{ELI}_\bot$-concepts C* and *$\mathcal{ELI}_\bot$-roles R* are built according to the following syntax rules:

$$\begin{aligned} C &::= \top \mid \bot \mid A \mid C \sqcap C \mid \exists R.C, \\ R &::= P \mid \mathsf{inv}(P), \end{aligned}$$

where $A \in \mathsf{N_C}$ and $P \in \mathsf{N_R}$. $\mathcal{ELI}_\bot$ is rather inexpressive compared to $\mathcal{ALCQIO}$, but a simulation of a two counter machine with an $\mathcal{ELI}_\bot$-ConGolog program is still possible.

Corollary 5.3. *Undecidability already holds for programs based on an $\mathcal{ELI}_\bot$-action theory.*

Proof (sketch). We modify the program $\mathcal{P}_\mathsf{M} = (\mathfrak{D}(\Sigma_\mathsf{M}), \delta_\mathsf{M})$ and the property Φ_M used in the previous reduction. We reuse the concept names: C_0 and C_1 for the two counters, *Halt* for the halting state, $J_0, \ldots, J_m$ for the different instructions and the roles *Succ* and

$$\mathit{Pred} := \mathsf{inv}(\mathit{Succ})$$

to represent the chain. Now, we do not axiomatize the chain in the TBox as before but create it “on-the-fly” using actions. For this purpose the concept name V captures the already created finite portion of the chain and the concept name *Max* captures the current rightmost elements in the chain. The extension of V will be expanded stepwise and we will ensure that the successors of elements in *Max* are not in V. Instead of the object names $\mathbf{0}$ and p we now use two rigid concept names *Zero* and *State*. The CI $\mathit{Zero} \sqsubseteq C_\ell$ is supposed to indicate that the value of the ℓth counter is equal to zero, $\mathit{State} \sqsubseteq J_i$ says that the ith instruction is executed next, and $\mathit{State} \sqsubseteq \mathit{Halt}$ indicates the halting state. We require that *Zero* and *State* are interpreted as non-empty sets. We use an object name o and the following initial ABox:

$$\mathcal{A}_\mathsf{M} := \{o \sqsubseteq \mathit{Zero} \sqcap \mathit{State}\}.$$

The TBox $\mathcal{T}_\mathsf{M}$ is empty. To initialize the machine there are two actions $\mathtt{init}_1$ and $\mathtt{init}_2$ with the following preconditions and effects:

$$\begin{aligned}&\mathsf{pre}(\mathtt{init}_1) := \emptyset,\\ &\mathsf{eff}(\mathtt{init}_1) := \{\langle C_0, \top\rangle^-, \langle C_1, \top\rangle^-, \langle V, \top\rangle^-, \langle \mathit{Max}, \top\rangle^-, \langle \mathit{Halt}, \top\rangle^-, \langle J_0, \top\rangle^-, \ldots, \langle J_m, \top\rangle^-\};\end{aligned}$$

The extension of all concept names are emptied except for *Zero* and *State*. Next, we initialize the two counters and the label for the first instruction.

$$\begin{aligned}&\mathsf{pre}(\mathtt{init}_2) := \emptyset\\ &\mathsf{eff}(\mathtt{init}_2) := \{\langle C_0, \mathit{Zero}\rangle^+, \langle C_1, \mathit{Zero}\rangle^+, \langle V, \mathit{Zero}\rangle^+, \langle \mathit{Max}, \mathit{Zero}\rangle^+, \langle J_0, \mathit{State}\rangle^+\}.\end{aligned}$$

Let n be the maximum value one of the two counters has reached so far during the execution of M. The representation of the counter values is done by partitioning the extension of V into n different pairwise disjoint sets given by the extensions of the following concepts:

$$\mathit{Zero}, \exists \mathit{Pred}.\mathit{Zero}, \exists \mathit{Pred}.(\exists \mathit{Pred}.\mathit{Zero}), \ldots, \exists \mathit{Pred}^n.\mathit{Zero}$$

We call this sets *layers* in the following. We enforce this structure in all interpretations that are reachable from an initial state. This is done by using appropriate preconditions of actions. The concept name *Max* always stores the maximum layer. Initially, after doing $\mathtt{init}_2$ the extension of V only consists of the extension of *Zero*, which at the same time also represents the maximum layer.

Next, we define the preconditions and effects of the ground actions $\mathtt{inst}_0, \ldots, \mathtt{inst}_m$. In case the kth instruction is an increment of the ℓth counter of the form $\mathsf{Inc}(\ell, i)$ we have

$$\begin{aligned}\mathsf{pre}(\mathtt{inst}_k) := \{&\mathit{State} \sqsubseteq J_k,\\ &\mathit{Max} \sqsubseteq \exists \mathit{Succ}.\top,\\ &\exists \mathit{Pred}.\mathit{Max} \sqcap V \sqsubseteq \bot\}\end{aligned}$$

$$\begin{aligned}\mathsf{eff}(\mathtt{inst}_k) := \{\,&\langle C_\ell, \exists \mathit{Pred}.C_\ell\rangle^+, \langle C_\ell, C_\ell\rangle^-,\\ &C_\ell \sqsubseteq \mathit{Max} \rhd \langle \mathit{Max}, \exists \mathit{Pred}.C_\ell\rangle^+,\\ &C_\ell \sqsubseteq \mathit{Max} \rhd \langle \mathit{Max}, \mathit{Max}\rangle^-,\\ &C_\ell \sqsubseteq \mathit{Max} \rhd \langle V, \exists \mathit{Pred}.C_\ell\rangle^+,\\ &\langle J_k, \mathit{State}\rangle^-,\\ &\langle J_i, \mathit{State}\rangle^+\},\end{aligned}$$

The two preconditions $\mathit{Max} \sqsubseteq \exists \mathit{Succ}.\top$ and $\exists \mathit{Pred}.\mathit{Max} \sqcap V \sqsubseteq \bot$ are needed in case the increment exceeds the previous upper bound of both counters. $\mathit{Max} \sqsubseteq \exists \mathit{Succ}.\top$ ensures that a new non-empty layer (consisting of the successors of the elements in *Max*) exists. The new layer $\exists \mathit{Pred}.\mathit{Max}$ has to be disjoint with all previous ones ($\exists \mathit{Pred}.\mathit{Max} \sqcap V \sqsubseteq \bot$). The effects $\langle C_\ell, \exists \mathit{Pred}.C_\ell\rangle^+$ and $\langle C_\ell, C_\ell\rangle^-$ shift C_ℓ to the next layer. In all legal states it is ensured that the extensions of C_ℓ and $\exists \mathit{Pred}.C_\ell$ represent two consecutive disjoint layers in the extension of V. The three conditional effects are triggered in case the previous upper bound needs to be incremented. The effects shift the maximum layer and expand the extension of V with the new layer. The effect descriptions on J_i and J_k implement the jump to the next instruction.

In case the kth instruction is a decrement of the form $\mathsf{Dec}(\ell, i, j)$ we have

$$\begin{aligned}\mathsf{pre}(\mathtt{inst}_k) &:= \{\mathit{State} \sqsubseteq J_k\}\\ \mathsf{eff}(\mathtt{inst}_k) &:= \{\, C_\ell \sqcap \mathit{Zero} \sqsubseteq \bot \rhd \langle C_\ell, V \sqcap \exists \mathit{Succ}.C_\ell\rangle^+,\\ &\qquad C_\ell \sqcap \mathit{Zero} \sqsubseteq \bot \rhd \langle C_\ell, C_\ell\rangle^-,\\ &\qquad C_\ell \sqcap \mathit{Zero} \sqsubseteq \bot \rhd \langle J_j, \mathit{State}\rangle^+,\\ &\qquad \langle J_k, \mathit{State}\rangle^-,\\ &\qquad \mathit{Zero} \sqsubseteq C_\ell \rhd \langle J_i, \mathit{State}\rangle^+\}.\end{aligned}$$

The extensions of V and Max remain unchanged. The only precondition ensures that we are in the correct state. The CI $\mathit{Zero} \sqsubseteq C_\ell$ represents the zero test. Since the concepts C_ℓ and Zero always consist of exactly one whole layer in V, the extensions of both concepts are either identical or disjoint. Thus, the disjointness axiom $C_\ell \sqcap \mathit{Zero} \sqsubseteq \bot$ expresses a proper non-zero test. The shift of C_ℓ to do the decrement is implemented as expected. The concept $V \sqcap \exists \mathit{Succ}.C_\ell$ represents the layer in V, where all elements have successors in C_ℓ. V is needed as a conjunct, because there can be pair $(d, d') \in \mathit{Succ}^{\mathcal{I}}$ with $d \notin V^{\mathcal{I}}$ and $d \in (C_\ell)^{\mathcal{I}}$.

In case the kth instruction is the halting instruction we have

$$\begin{aligned}\mathsf{pre}(\mathtt{inst}_k) &:= \{\mathit{State} \sqsubseteq J_k\},\\ \mathsf{eff}(\mathtt{inst}_k) &:= \{\langle \mathit{Halt}, \mathit{State}\rangle^+\}.\end{aligned}$$

The program expression is given by

$$\delta_{\mathsf{M}} := \mathtt{init}_1; \mathtt{init}_2; (\mathtt{inst}_0 \mid \mathtt{inst}_1 \mid \cdots \mid \mathtt{inst}_m)^*; (\mathit{State} \sqsubseteq \mathit{Halt})?$$

We claim that M reaches a halting configuration iff the $\mathcal{ELI}_\bot$-CTL* state formula

$$\Phi_{\mathsf{M}} := \mathsf{AX}(\mathsf{AX}(\mathsf{AG}(\mathit{State} \sqcap \mathit{Halt} \sqsubseteq \bot)))$$

is not valid in $\mathcal{P}_{\mathsf{M}}$. The two next-operators are needed to skip the states before initialization. We omit a detailed proof. □

5.2 General Dynamic Types and Regression

In this section, an abstraction technique for (unrestricted) DL-ConGolog programs is introduced. Due to the undecidability of the verification problem the technique does not lead to a decision procedure. In the general case the abstraction we introduce is a possibly infinite context-bisimilar propositional transition system. Two examples of decidable classes where the abstraction technique actually leads to a finite state system are introduced later in the subsequent sections. Here, we only introduce the basic constructions for the general case.

The abstraction technique for programs over local effect actions is based on the notion of a dynamic type of an interpretation. The execution of actions and programs in interpretations of the same dynamic type leads to an indistinguishable behavior w.r.t. the set of all relevant axioms (context). This is achieved by encoding not only the static type of an interpretation but also the static types of all possible updates in the dynamic type. In the local effect case we can describe the changes caused by an arbitrary action sequence as a finite set of effect descriptions taken from a finite set of relevant effects. This leads to finitely many possible changes and to finitely many dynamic types which in turn allows the construction of a finite bisimilar abstraction of the transition system of the program.

In this section we generalize the notion of dynamic types to a domain with arbitrary DL-definable actions. We basically use the same idea as for the local effect case. However, now we have to deal with effects of the form $\langle F, X\rangle^{\pm}$, where X is a complex concept or role and possibly mentions other concept and role names. In this general setting, it is no longer sufficient to only consider finitely many of such effects. Therefore, the generalized definition leads to a possibly infinite set of dynamic types.

5.2.1 Dynamic Types in Presence of Non-Local Effects

In the sequel, we first collect a couple of preliminary notions needed for the definition of dynamic types.

First, we represent the effects of $\mathcal{DL}$-definable actions as a set of effect descriptions.

Definition 5.4. Let $\boldsymbol{A}$ be a finite set of ground actions and $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$. For an action $\boldsymbol{\alpha} \in \boldsymbol{A}$ and static type $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ *the effects of executing* $\boldsymbol{\alpha}$ *in* $\mathfrak{s}$ are defined as follows:

$$\Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha}) := \left\{ \left\langle F, \mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F] \right\rangle^+ \,\middle|\, F \in \mathcal{F} \right\} \cup \left\{ \left\langle F, \mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F] \right\rangle^- \,\middle|\, F \in \mathcal{F} \right\}.$$

Since the concepts or roles $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$ and $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$ are computable for any given tuple $(\mathfrak{s}, \boldsymbol{\alpha}, F) \in \mathfrak{S}_{\mathcal{C}} \times \boldsymbol{A} \times \mathcal{F}$, the sets $\Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$ are also computable. ▲

Executing an action $\boldsymbol{\alpha} \in \boldsymbol{A}$ in an interpretation $\mathcal{I}$ means updating $\mathcal{I}$ with $\Sigma_{\boldsymbol{A}}(\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}), \boldsymbol{\alpha})$.

Lemma 5.5. *Let $\boldsymbol{A}$ be a finite set of ground actions, $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ and $\mathfrak{D}(\Sigma_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K})\mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ the induced FO-DS. For all $\mathcal{I} \in \mathbb{I}$ and all $\boldsymbol{\alpha} \in \boldsymbol{A}$ it holds that $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{I}^{\mathsf{E}}$ with $\mathsf{E} = \Sigma_{\boldsymbol{A}}(\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}), \boldsymbol{\alpha})$.*

Next, we show that also the changes caused by a sequence of ground actions can be defined as a set of $\mathcal{DL}$-effects. The representation $\Sigma_{\boldsymbol{A}}$ provides definitions of the add-sets and delete-sets for each individual ground action in form of $\mathcal{DL}$-concepts or $\mathcal{DL}$-roles. The problem is how to obtain the corresponding definitions for sequences of actions. Consider the FO-DS

$$\mathfrak{D}(\Sigma_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K})\mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$$

induced by $\Sigma_{\boldsymbol{A}}$, a fluent $F \in \mathcal{F}$, an action sequence $\sigma = \boldsymbol{\alpha}_0\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ and an interpretation $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$. The execution of σ in $\mathcal{I}_0$ generates a sequence of interpretations:

$$\mathcal{I}_0 \Rightarrow^{\boldsymbol{\alpha}_0}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{I}_1 \Rightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \cdots \Rightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{I}_{n+1}.$$

Clearly, there are sets $X^+_\sigma \subseteq \Delta^{\mathsf{ar}(F)}_{\mathcal{I}_0}$ and $X^-_\sigma \subseteq \Delta^{\mathsf{ar}(F)}_{\mathcal{I}_0}$ such that

$$F^{\mathcal{I}_{n+1}} = (F^{\mathcal{I}_0} \setminus X^-_\sigma) \cup X^+_\sigma,$$

where X_σ^+ is the set the execution of σ in $\mathcal{I}_0$ adds to $F^{\mathcal{I}_0}$ and X_σ^- the set that is deleted. By assumption on Σ_A we can compute for each $i \in \{0,\ldots,n\}$ two concepts (or roles) that provide a definition of the corresponding add-set $\mathsf{add}(\mathcal{I}_i, \boldsymbol{\alpha}_i, F)$ and delete-set $\mathsf{del}(\mathcal{I}_i, \boldsymbol{\alpha}_i, F)$ in $\mathcal{I}_i$. The question is how to obtain definitions of the sets X_σ^+ and X_σ^- in $\mathcal{I}_0$ (given the definitions of the inidividual add- an delete-sets). Note that for all $i \in \{0,\ldots,n\}$ we have

$$\mathsf{add}(\mathcal{I}_i, \boldsymbol{\alpha}_i, F) = \left(\mathsf{E}^+[\text{s-type}_\mathcal{C}(\mathcal{I}_i), \boldsymbol{\alpha}_i, F]\right)^{\mathcal{I}_i} \text{ and } \mathsf{del}(\mathcal{I}_i, \boldsymbol{\alpha}_i, F) = \left(\mathsf{E}^-[\text{s-type}_\mathcal{C}(\mathcal{I}_i), \boldsymbol{\alpha}_i, F]\right)^{\mathcal{I}_i}.$$

Thus, the definitions given by $\mathsf{E}^+[\cdot]$ and $\mathsf{E}^-[\cdot]$ for the ith action in the sequence refer to the interpretation $\mathcal{I}_i$, but to define the sets X_σ^+ and X_σ^- in $\mathcal{I}_0$ we need a definition that only refers to the initial interpretation $\mathcal{I}_0$.

To obtain such definitions in form of $\mathcal{DL}$-concepts and roles we use a well-known technique called regression. In Reiter's basic action theories it can be used for solving the projection problem [Rei91]: given a sequence of actions σ and a formula φ the *regression of φ through σ* is a formula that is true *before* doing σ iff φ is true *after* doing σ. We define a single-step regression operator for concepts, roles and Boolean KBs through a given set of effect descriptions.

Definition 5.6. Let E be a set of unconditional effect descriptions. For a ground $\mathcal{DL}$-concept C and $\mathcal{DL}$-role R the *regression of C and R, respectively, through* E, denoted by $\mathfrak{R}[C,\mathsf{E}]$ and $\mathfrak{R}[R,\mathsf{E}]$, respectively, is a $\mathcal{DL}$-concept and $\mathcal{DL}$-role, respectively, that is defined inductively as given in Figure 5.2. The regression operator is also defined for a Boolean $\mathcal{DL}$-KB. ▲

The operator materializes all effects on the syntax level. The next lemma states that the regression result correctly captures the extension of the concept or role after an update with the corresponding set of effects.

Lemma 5.7. *Let $\mathcal{I}$ be an interpretation over* $\mathsf{N_C}$,$\mathsf{N_R}$ *and* $\mathsf{N_O}$, E *a set of effects, C a $\mathcal{DL}$-concept, R a $\mathcal{DL}$-role and ψ a Boolean $\mathcal{DL}$-KB. It holds that*

1. $(\mathfrak{R}[C,\mathsf{E}])^\mathcal{I} = C^{\mathcal{I}^\mathsf{E}}$*;*
2. $(\mathfrak{R}[R,\mathsf{E}])^\mathcal{I} = R^{\mathcal{I}^\mathsf{E}}$*;*
3. $\mathcal{I} \models \mathfrak{R}[\psi,\mathsf{E}]$ *iff* $\mathcal{I}^\mathsf{E} \models \psi$*.*

Proof. We prove the first claim for concept names. Let $A \in \mathsf{N_C}$. We have

$$\begin{aligned}
(\mathfrak{R}[A,\mathsf{E}])^\mathcal{I} &= \Big(\big(A \sqcap \bigsqcap_{\langle A,C\rangle^- \in \mathsf{E}} \neg C\big) \sqcup \bigsqcup_{\langle A,C\rangle^+ \in \mathsf{E}} C\Big)^\mathcal{I} \\
&= \big(A^\mathcal{I} \cap \bigcap_{\langle A,C\rangle^- \in \mathsf{E}} (\Delta_\mathcal{I} \setminus C^\mathcal{I})\big) \cup \bigcup_{\langle A,C\rangle^+ \in \mathsf{E}} C^\mathcal{I} \\
&= \Big(A^\mathcal{I} \setminus \Big(\bigcup_{\langle A,C\rangle^- \in \mathsf{E}} C^\mathcal{I}\Big)\Big) \cup \bigcup_{\langle A,C\rangle^+ \in \mathsf{E}} C^\mathcal{I} \\
&= A^{\mathcal{I}^\mathsf{E}}.
\end{aligned}$$

The proof for role names is analogous. For complex concepts, complex roles and Boolean KBs the proof is by structural induction. □

$$
\begin{aligned}
&\mathfrak{R}[A,\mathsf{E}] && := \big(A \sqcap \bigsqcap_{\langle A,C\rangle^- \in \mathsf{E}} \neg C\big) \sqcup \bigsqcup_{\langle A,C\rangle^+ \in \mathsf{E}} C, \text{ with } A \in \mathsf{N_C};\\
&\mathfrak{R}[\top,\mathsf{E}] && := \top\\
&\mathfrak{R}[\bot,\mathsf{E}] && := \bot\\
&\mathfrak{R}[\{o\},\mathsf{E}] && := \{o\};\\
&\mathfrak{R}[\neg C,\mathsf{E}] && := \neg\mathfrak{R}[C,\mathsf{E}];\\
&\mathfrak{R}[C \sqcap D,\mathsf{E}] && := \mathfrak{R}[C,\mathsf{E}] \sqcap \mathfrak{R}[D,\mathsf{E}];\\
&\mathfrak{R}[\Xi\, R.C,\mathsf{E}] && := \Xi\mathfrak{R}[R,\mathsf{E}].\mathfrak{R}[C,\mathsf{E}] \text{ with } \Xi \in \{\exists,\forall,\geq n\};
\end{aligned}
$$

For a role R, the regression $\mathfrak{R}[R,\mathsf{E}]$ is defined as follows:

$$
\begin{aligned}
&\mathfrak{R}[P,\mathsf{E}] && := \Big(P \setminus \Big(\bigsqcup_{\langle P,R\rangle^- \in \mathsf{E}} R\Big)\Big) \sqcup \bigsqcup_{\langle P,R\rangle^+ \in \mathsf{E}} R, \text{ with } P \in \mathsf{N_R};\\
&\mathfrak{R}[\mathsf{inv}(R),\mathsf{E}] && := \mathsf{inv}(\mathfrak{R}[R,\mathsf{E}])\\
&\mathfrak{R}[\mathsf{id},\mathsf{E}] && := \mathsf{id};\\
&\mathfrak{R}[\{(o,o')\},\mathsf{E}] && := \{(o,o')\};\\
&\mathfrak{R}[\neg R,\mathsf{E}] && := \neg\mathfrak{R}[R,\mathsf{E}];\\
&\mathfrak{R}[R \sqcap S,\mathsf{E}] && := \mathfrak{R}[R,\mathsf{E}] \sqcap \mathfrak{R}[S,\mathsf{E}];\\
&\mathfrak{R}[C \times D,\mathsf{E}] && := \mathfrak{R}[C,\mathsf{E}] \times \mathfrak{R}[D,\mathsf{E}];
\end{aligned}
$$

For a Boolean KB ψ we have:

$$
\begin{aligned}
&\mathfrak{R}[C \sqsubseteq D,\mathsf{E}] && := \mathfrak{R}[C,\mathsf{E}] \sqsubseteq \mathfrak{R}[D,\mathsf{E}];\\
&\mathfrak{R}[\neg\psi,\mathsf{E}] && := \neg\mathfrak{R}[\psi,\mathsf{E}]\\
&\mathfrak{R}[\psi_1 \wedge \psi_2,\mathsf{E}] && := \mathfrak{R}[\psi_1,\mathsf{E}] \wedge \mathfrak{R}[\psi_2,\mathsf{E}].
\end{aligned}
$$

Figure 5.2: Regression in $\mathcal{DL}$ w.r.t. a set of effects

Using the regression operator sets of effect descriptions can be accumulated.

Definition 5.8. Let E_0 and E_1 be two sets of unconditional $\mathcal{DL}$-effects. We define the accumulation of E_0 and E_1 as the union of three sets. With $\mathsf{Regr}(\mathsf{E}_1, \mathsf{E}_0)$ we denote the set of effects obtained by regressing the effect descriptors of all effects in E_1 through E_0:

$$\mathsf{Regr}(\mathsf{E}_1, \mathsf{E}_0) := \{\langle F, \mathfrak{R}[Y, \mathsf{E}_0]\rangle^{\pm} \mid \langle F, Y\rangle^{\pm} \in \mathsf{E}_1\}.$$

The next set of effects, denoted by $\mathsf{Diff}^+(\mathsf{E}_0, \mathsf{E}_1)$, is obtained by subtracting the effect descriptors of negative effects in E_1 from the positive ones in E_0:

$$\mathsf{Diff}^+(\mathsf{E}_0, \mathsf{E}_1) := \left\{ \left\langle F, \big(X \sqcap \bigsqcap_{\langle F,Y\rangle^- \in \mathsf{E}_1} \neg\mathfrak{R}[Y, \mathsf{E}_0]\big) \right\rangle^+ \,\middle|\, \langle F,X\rangle^+ \in \mathsf{E}_0 \right\}$$

The *accumulation of* E_0 *and* E_1, denoted by $\mathsf{E}_0 \bowtie \mathsf{E}_1$, is a set of effects defined as the union:

$$\mathsf{E}_0 \bowtie \mathsf{E}_1 := \mathsf{Regr}(\mathsf{E}_1, \mathsf{E}_0) \cup \mathsf{Diff}^+(\mathsf{E}_0, \mathsf{E}_1) \cup \{\langle F,X\rangle^- \in \mathsf{E}_0\}.$$

▲

What we want to achieve is that first updating an interpretation $\mathcal{I}$ with E_0 and than afterwards with E_1 gives the same result as a single update of $\mathcal{I}$ with the combined set of effects denoted by $\mathsf{E}_0 \bowtie \mathsf{E}_1$. Therefore, we need to regress the effect descriptors of the effects in E_1 through E_0 to take into account that the update with E_0 occurs before the update with E_1. This is captured by including $\mathsf{Regr}(\mathsf{E}_1, \mathsf{E}_0)$. The set $\mathsf{Diff}^+(\mathsf{E}_0, \mathsf{E}_1)$ is needed when an addition in E_0 is afterwards canceled by a deletion effect in E_1. The negative effects in E_0 are kept without any rewriting. It might be the case that a positive effect in E_1 overwrites a negative one in E_0 but due to the add-after-delete semantics of updates it is not necessary to make this explicit.

Lemma 5.9. *Let $\mathcal{I}$ be an interpretation over $\mathsf{N_C}$, $\mathsf{N_R}$ and $\mathsf{N_O}$ and E_0 and E_1 two sets of $\mathcal{DL}$-effects. It holds that*

$$(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1} = \mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}.$$

Proof. The semantics of updates with sets of effects in Definition 2.28 implies that

$$\Delta_{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}} = \Delta_{\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}} \text{ and } o^{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}} = o^{\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}} \text{ for all } o \in \mathsf{N_O}.$$

Let $F \in \mathsf{N_C} \cup \mathsf{N_R}$. We show that also

$$F^{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}} = F^{\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}} \text{ holds.}$$

We have that the interpretation $(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}$ is the update of $\mathcal{I}^{\mathsf{E}_0}$ with E_1 and $\mathcal{I}^{\mathsf{E}_0}$ is the update of $\mathcal{I}$ with E_0. Using the definition of updates we obtain

$$F^{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}} = \big(F^{\mathcal{I}^{\mathsf{E}_0}} \setminus \big(\bigcup_{\langle F,Y\rangle^- \in \mathsf{E}_1} Y^{\mathcal{I}^{\mathsf{E}_0}}\big)\big) \cup \bigcup_{\langle F,Y\rangle^+ \in \mathsf{E}_1} Y^{\mathcal{I}^{\mathsf{E}_0}}.$$

In the next steps we rewrite the expression on the right-hand side of the equation to obtain a definition of the set $F^{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}}$ that refers only to the interpretation $\mathcal{I}$. For better readability we introduce some abbreviations:

$$M^+ := \bigcup_{\langle F,Y\rangle^+\in\mathsf{E}_1} Y^{\mathcal{I}^{\mathsf{E}_0}} \text{ and } M^- := \bigcup_{\langle F,Y\rangle^-\in\mathsf{E}_1} Y^{\mathcal{I}^{\mathsf{E}_0}}$$

With Lemma 5.7 it follows that

$$M^+ = \bigcup_{\langle F,Y\rangle^+\in\mathsf{E}} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}} \text{ and } M^- = \bigcup_{\langle F,Y\rangle^-\in\mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}. \tag{5.15}$$

The definition of $F^{\mathcal{I}^{\mathsf{E}_0}}$ yields (we assume that "\" binds stronger than "∪")

$$\begin{aligned}
&F^{\mathcal{I}^{\mathsf{E}_0}} \setminus M^- \cup M^+\\
=&\Big(F^{\mathcal{I}} \setminus \big(\bigcup_{\langle F,X\rangle^-\in\mathsf{E}_0} X^{\mathcal{I}}\big) \cup \bigcup_{\langle F,X\rangle^+\in\mathsf{E}_0} X^{\mathcal{I}}\Big) \setminus M^- \cup M^+\\
=&\Big(F^{\mathcal{I}} \setminus \big(\bigcup_{\langle F,X\rangle^-\in\mathsf{E}_0} X^{\mathcal{I}}\big)\Big) \setminus M^- \cup \bigcup_{\langle F,X\rangle^+\in\mathsf{E}_0} (X^{\mathcal{I}} \setminus M^-) \cup M^+\\
=&\; F^{\mathcal{I}} \setminus \big(\big(\bigcup_{\langle F,X\rangle^-\in\mathsf{E}_0} X^{\mathcal{I}}\big) \cup M^-\big) \cup \bigcup_{\langle F,X\rangle^+\in\mathsf{E}_0} (X^{\mathcal{I}} \setminus M^-) \cup M^+
\end{aligned}$$

With (5.15) it follows that

$$\begin{aligned}
F^{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}} = &\Big(F^{\mathcal{I}} \setminus \Big(\bigcup_{\langle F,X\rangle^-\in\mathsf{E}_0} X^{\mathcal{I}} \cup \bigcup_{\langle F,Y\rangle^-\in\mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}\Big)\Big) \cup\\
&\bigcup_{\langle F,X\rangle^+\in\mathsf{E}_0} \Big(X^{\mathcal{I}} \setminus \big(\bigcup_{\langle F,Y\rangle^-\in\mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}\big)\Big) \cup\\
&\bigcup_{\langle F,Y\rangle^+\in\mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}.
\end{aligned}$$

Obviously, it holds that

$$X^{\mathcal{I}} \setminus \big(\bigcup_{\langle F,Y\rangle^-\in\mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}\big) = \Big(X \sqcap \bigsqcap_{\langle F,Y\rangle^-\in\mathsf{E}_1} \neg\mathfrak{R}[Y,\mathsf{E}_0]\Big)^{\mathcal{I}}.$$

With the equations above and the definition of $\mathsf{E}_0 \bowtie \mathsf{E}_1$ we obtain

$$
\begin{aligned}
F^{(\mathcal{I}^{\mathsf{E}_0})^{\mathsf{E}_1}} &= \Big(F^{\mathcal{I}} \setminus \Big(\bigcup_{\langle F,X\rangle^- \in \mathsf{E}_0} X^{\mathcal{I}} \cup \bigcup_{\langle F,Y\rangle^- \in \mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}} \Big)\Big) \cup \\
&\quad \bigcup_{\langle F,X\rangle^+ \in \mathsf{E}_0} \Big(X \sqcap \bigsqcap_{\langle F,Y\rangle^- \in \mathsf{E}_1} \neg \mathfrak{R}[Y,\mathsf{E}_0] \Big)^{\mathcal{I}} \cup \\
&\quad \bigcup_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}. \\
&= F^{\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}}.
\end{aligned}
$$

□

The lemma above is a generalization of Lemma 4.6 to arbitrary sets of effects. The definition of $\mathsf{E}_0 \bowtie \mathsf{E}_1$ possibly leads to an overlap of positive and negative effects because the negative effects in E_0 are not rewritten.

Definition 5.10. Let E be a finite set of unconditional effect descriptions. We say that E is *coherent* iff for all pairs of effects of the form $\{\langle F,X\rangle^+, \langle F,Y\rangle^-\} \subseteq \mathsf{E}$ for some fluent $F \in \mathsf{N_F}$ it is implied that $X^{\mathcal{I}} \cap Y^{\mathcal{I}} = \emptyset$ for all interpretations $\mathcal{I}$. ▲

A coherence-preserving version of the accumulation operator can be easily obtained: let E_0 and E_1 be two set of unconditional $\mathcal{DL}$-effects. The negative counterpart of $\mathsf{Diff}^+(\mathsf{E}_0,\mathsf{E}_1)$ is defined by

$$
\mathsf{Diff}^-(\mathsf{E}_0,\mathsf{E}_1) := \Big\{ \Big\langle F, \big(X \sqcap \bigsqcap_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \neg \mathfrak{R}[Y,\mathsf{E}_0]\big) \Big\rangle^- \,\Big|\, \langle F,X\rangle^- \in \mathsf{E}_0 \Big\}.
$$

The *coherence-preserving accumulation* of E_0 and E_1, denoted by $\mathsf{E}_0 \bowtie_{\mathsf{c}} \mathsf{E}_1$, is defined as follows

$$
\mathsf{E}_0 \bowtie_{\mathsf{c}} \mathsf{E}_1 := \mathsf{Regr}(\mathsf{E}_1,\mathsf{E}_0) \cup \mathsf{Diff}^+(\mathsf{E}_0,\mathsf{E}_1) \cup \mathsf{Diff}^-(\mathsf{E}_0,\mathsf{E}_1).
$$

Note that interpretation updates with sets of effect descriptions are defined with an add-after-delete semantics. In case of an overlap precedence is given to the corresponding add-effect. Due to this semantics both accumulations lead to exactly the same update result.

Lemma 5.11. *Let E_0 and E_1 be two sets of unconditional $\mathcal{DL}$-effects. The coherence-preserving accumulation of E_0 and E_1 has the following properties.*

1. *If E_0 and E_1 are coherent, then also $\mathsf{E}_0 \bowtie_{\mathsf{c}} \mathsf{E}_1$ is coherent.*

2. *Let $\mathcal{I}$ be an interpretation over $\mathsf{N_C}, \mathsf{N_R}$ and $\mathsf{N_O}$. It holds that $\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1} = \mathcal{I}^{\mathsf{E}_0 \bowtie_{\mathsf{c}} \mathsf{E}_1}$.*

Proof. We have

$$
\mathsf{E}_0 \bowtie_{\mathsf{c}} \mathsf{E}_1 = \mathsf{Regr}(\mathsf{E}_1,\mathsf{E}_0) \cup \mathsf{Diff}^+(\mathsf{E}_0,\mathsf{E}_1) \cup \mathsf{Diff}^-(\mathsf{E}_0,\mathsf{E}_1).
$$

Assume E_0 and E_1 are coherent. We prove that $\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1$ is coherent. Let

$$\{\langle F,X\rangle^+, \langle F,Y\rangle^-\} \subseteq \mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1.$$

We have to show that $X^{\mathcal{I}} \cap Y^{\mathcal{I}} = \emptyset$ for all interpretations $\mathcal{I}$. We distinguish three cases.

(i) Assume $\{\langle F,X\rangle^+, \langle F,Y\rangle^-\} \subseteq \mathsf{Regr}(\mathsf{E}_1,\mathsf{E}_0)$. Thus, there are effects $\langle F,X_1\rangle^+ \in \mathsf{E}_1$ and $\langle F,Y_1\rangle^- \in \mathsf{E}_1$ such that $X = \mathfrak{R}[X_1,\mathsf{E}_0]$ and $Y = \mathfrak{R}[Y_1,\mathsf{E}_0]$. Let $\mathcal{I}$ be an interpretation. It holds that

$$X^{\mathcal{I}} \cap Y^{\mathcal{I}} = (\mathfrak{R}[X_1,\mathsf{E}_0])^{\mathcal{I}} \cap (\mathfrak{R}[Y_1,\mathsf{E}_0])^{\mathcal{I}} \overset{\text{(Lemma 5.7)}}{=} X_1^{\mathcal{I}^{\mathsf{E}_0}} \cap Y_1^{\mathcal{I}^{\mathsf{E}_0}} \overset{(\mathsf{E}_1 \text{ is coherent})}{=} \emptyset.$$

(ii) Assume $\langle F,X\rangle^+ \in \mathsf{Diff}^+(\mathsf{E}_0,\mathsf{E}_1)$ and $\langle F,Y\rangle^- \in \mathsf{Diff}^-(\mathsf{E}_0,\mathsf{E}_1)$. Thus, there are effects $\langle F,X_0\rangle^+ \in \mathsf{E}_0$ and $\langle F,Y_0\rangle^- \in \mathsf{E}_0$ such that

$$X = X_0 \sqcap \bigsqcap_{\langle F,Z\rangle^- \in \mathsf{E}_1} \neg\mathfrak{R}[Z,\mathsf{E}_0] \text{ and } Y = Y_0 \sqcap \bigsqcap_{\langle F,W\rangle^+ \in \mathsf{E}_1} \neg\mathfrak{R}[W,\mathsf{E}_0].$$

Let $\mathcal{I}$ be an interpretation. It holds that

$$X^{\mathcal{I}} \cap Y^{\mathcal{I}} = \Big(X_0 \sqcap \bigsqcap_{\langle F,Z\rangle^- \in \mathsf{E}_1} \neg\mathfrak{R}[Z,\mathsf{E}_0]\Big)^{\mathcal{I}} \cap \Big(Y_0 \sqcap \bigsqcap_{\langle F,W\rangle^+ \in \mathsf{E}_1} \neg\mathfrak{R}[W,\mathsf{E}_0]\Big)^{\mathcal{I}}.$$

It follows that $X^{\mathcal{I}} \subseteq X_0^{\mathcal{I}}$ and $Y^{\mathcal{I}} \subseteq Y_0^{\mathcal{I}}$. $X_0^{\mathcal{I}}$ and $Y_0^{\mathcal{I}}$ are disjoint, because E_0 is coherent. Obviously, it follows that $X^{\mathcal{I}}$ and $Y^{\mathcal{I}}$ are disjoint.

(iii) Assume $\langle F,X\rangle^+ \in \mathsf{Regr}(\mathsf{E}_1,\mathsf{E}_0)$ and $\langle F,Y\rangle^- \in \mathsf{Diff}^-(\mathsf{E}_0,\mathsf{E}_1)$. Thus, there are effects $\langle F,X_1\rangle^+ \in \mathsf{E}_1$ and and $\langle F,Y_0\rangle^- \in \mathsf{E}_0$ such that

$$X = \mathfrak{R}[X_1,\mathsf{E}_0] \text{ and } Y = Y_0 \sqcap \bigsqcap_{\langle F,W\rangle^+ \in \mathsf{E}_1} \neg\mathfrak{R}[W,\mathsf{E}_0].$$

Let $\mathcal{I}$ be an interpretation. It follows that

$$Y^{\mathcal{I}} = Y_0^{\mathcal{I}} \setminus \Big(\bigcup_{\langle F,W\rangle^+ \in \mathsf{E}_1} (\mathfrak{R}[W,\mathsf{E}_0])^{\mathcal{I}}\Big).$$

We have

$$X^{\mathcal{I}} = (\mathfrak{R}[X_1,\mathsf{E}_0])^{\mathcal{I}} \subseteq \Big(\bigcup_{\langle F,W\rangle^+ \in \mathsf{E}_1} (\mathfrak{R}[W,\mathsf{E}_0])^{\mathcal{I}}\Big).$$

Consequently, $X^{\mathcal{I}} \cap Y^{\mathcal{I}} = \emptyset$.

(iv) The case with $\langle F,X\rangle^+ \in \mathsf{Diff}^+(\mathsf{E}_0,\mathsf{E}_1)$ and $\langle F,Y\rangle^- \in \mathsf{Regr}(\mathsf{E}_1,\mathsf{E}_0)$ is analogous to the previous case.

It follows that $\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1$ is coherent. For the proof of the second part of the lemma consider an interpretation $\mathcal{I}$ over $\mathsf{N_C}$,$\mathsf{N_R}$ and $\mathsf{N_O}$. We show that

$$\mathcal{I}^{\mathsf{E}_0 \ltimes \mathsf{E}_1} = \mathcal{I}^{\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1}.$$

Both updated interpretations have the same domain and interpret object names in the same way. Let $F \in \mathsf{N_C} \cup \mathsf{N_R}$. We have

$$F^{\mathcal{I}^{\mathsf{E}_0 \ltimes \mathsf{E}_1}} = F^{\mathcal{I}} \setminus M^- \cup M^+$$

where

$$M^- = \Big(\bigcup_{\langle F,Z\rangle^- \in\, (\mathsf{E}_0 \ltimes \mathsf{E}_1)} Z^{\mathcal{I}}\Big) = \bigcup_{\langle F,X\rangle^- \in \mathsf{E}_0} X^{\mathcal{I}} \cup \bigcup_{\langle F,Y\rangle^- \in \mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}} \text{ and}$$

$$M^+ = \Big(\bigcup_{\langle F,Z\rangle^+ \in\, (\mathsf{E}_0 \ltimes \mathsf{E}_1)} Z^{\mathcal{I}}\Big) = \bigcup_{\langle F,X\rangle^+ \in \mathsf{E}_0} \Big(X \sqcap \bigsqcap_{\langle F,Y\rangle^- \in \mathsf{E}_1} \neg \mathfrak{R}[Y,\mathsf{E}_0]\Big)^{\mathcal{I}} \cup \bigcup_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}.$$

With $\ltimes_{\mathsf{c}}$ we have

$$F^{\mathcal{I}^{\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1}} = F^{\mathcal{I}} \setminus M_{\mathsf{c}}^- \cup M_{\mathsf{c}}^+$$

with

$$M_{\mathsf{c}}^- = \Big(\bigcup_{\langle F,Z\rangle^- \in\, (\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1)} Z^{\mathcal{I}}\Big) = \bigcup_{\langle F,X\rangle^- \in \mathsf{E}_0} \Big(X \sqcap \bigsqcap_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \neg \mathfrak{R}[Y,\mathsf{E}_0]\Big)^{\mathcal{I}} \cup \bigcup_{\langle F,Y\rangle^- \in \mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}} \text{ and}$$

$$M_{\mathsf{c}}^+ = \Big(\bigcup_{\langle F,Z\rangle^+ \in\, (\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1)} Z^{\mathcal{I}}\Big) = \bigcup_{\langle F,X\rangle^+ \in \mathsf{E}_0} \Big(X \sqcap \bigsqcap_{\langle F,Y\rangle^- \in \mathsf{E}_1} \neg \mathfrak{R}[Y,\mathsf{E}_0]\Big)^{\mathcal{I}} \cup \bigcup_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \mathfrak{R}[Y,\mathsf{E}_0]^{\mathcal{I}}.$$

It follows that

$$M^+ = M_{\mathsf{c}}^+ \text{ and } M_{\mathsf{c}}^- \subseteq M^-.$$

Hence,

$$(F^{\mathcal{I}} \setminus M^-) \subseteq (F^{\mathcal{I}} \setminus M_{\mathsf{c}}^-) \text{ and } F^{\mathcal{I}^{\mathsf{E}_0 \ltimes \mathsf{E}_1}} \subseteq F^{\mathcal{I}^{\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1}}.$$

It remains to be shown that $F^{\mathcal{I}^{\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1}} \subseteq F^{\mathcal{I}^{\mathsf{E}_0 \ltimes \mathsf{E}_1}}$ also holds. Assume to the contrary that there exists $\bar{d} \in (\Delta_{\mathcal{I}})^{\mathsf{ar}(F)}$ with $\bar{d} \in F^{\mathcal{I}^{\mathsf{E}_0 \ltimes_{\mathsf{c}} \mathsf{E}_1}}$ and $\bar{d} \notin F^{\mathcal{I}^{\mathsf{E}_0 \ltimes \mathsf{E}_1}}$. With $M^+ = M_{\mathsf{c}}^+$ it follows that $\bar{d} \in (F^{\mathcal{I}} \setminus M_{\mathsf{c}}^-)$ and $\bar{d} \notin (F^{\mathcal{I}} \setminus M^-)$. Therefore, $\bar{d} \in M^-$ and $\bar{d} \notin M_{\mathsf{c}}^-$. It follows that

$$\bar{d} \notin \bigcup_{\langle F,Y\rangle^- \in \mathsf{E}_1} (\mathfrak{R}[Y,\mathsf{E}_0])^{\mathcal{I}} \text{ and } \bar{d} \in X^{\mathcal{I}} \text{ for some } \langle F,X\rangle^- \in \mathsf{E}_0.$$

Furthermore, $\bar{d} \notin M_{\mathsf{c}}^-$ implies

$$\bar{d} \notin \bigcup_{\langle F,X\rangle^- \in \mathsf{E}_0} \Big(X \sqcap \bigsqcap_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \neg \mathfrak{R}[Y,\mathsf{E}_0]\Big)^{\mathcal{I}}.$$

Since $\bar{d} \in X^{\mathcal{I}}$ for some $\langle F, X\rangle^- \in \mathsf{E}_0$, it follows that

$$\bar{d} \notin \Big(\prod_{\langle F,Y\rangle^+ \in \mathsf{E}_1} \neg\mathfrak{R}[Y, \mathsf{E}_0] \Big)^{\mathcal{I}}.$$

Thus, there exists an effect $\langle F, Y\rangle^+ \in \mathsf{E}_1$ such that $\bar{d} \in (\mathfrak{R}[Y, \mathsf{E}_0])^{\mathcal{I}}$. Consequently, $\bar{d} \in M^+$ which yields $\bar{d} \in F^{\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}}$ which is a contradiction to our initial assumption. Therefore, we have $F^{\mathcal{I}^{\mathsf{E}_0 \bowtie_c \mathsf{E}_1}} = F^{\mathcal{I}^{\mathsf{E}_0 \bowtie \mathsf{E}_1}}$. □

In the following we do not care about coherence and use $\bowtie$ instead of $\bowtie_c$ because it simplifies the technical treatment. As above, consider the execution of a ground action sequence $\sigma = \boldsymbol{\alpha}_0 \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ in an interpretation $\mathcal{I}_0$, where $\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ is a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$:

$$\mathcal{I}_0 \Rightarrow^{\boldsymbol{\alpha}_0}_{\mathfrak{D}(\Sigma_A)} \mathcal{I}_1 \Rightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}(\Sigma_A)} \cdots \Rightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}(\Sigma_A)} \mathcal{I}_{n+1}.$$

The changes in $\mathcal{I}_0$ caused by the execution of σ can be described by the following accumulated set of $\mathcal{DL}$-effects:

$$\Sigma_A(\mathfrak{s}_0, \boldsymbol{\alpha}_0) \bowtie \Sigma_A(\mathfrak{s}_1, \boldsymbol{\alpha}_1) \bowtie \cdots \bowtie \Sigma_A(\mathfrak{s}_n, \boldsymbol{\alpha}_n)$$

with $\mathfrak{s}_i = \textsf{s-type}_{\mathcal{C}}(\mathcal{I}_i)$ for all $i \in \{0, \ldots, n\}$.

Example 5.12. Consider the action `move-fwd` defined in Example 5.1. The execution affects the extension of the concept name *CurrentRoom* (abbreviated with *CR* in the following). In case the robot is facing right ($\mathsf{right} \sqsubseteq \mathit{Direction}$ is true) `move-fwd` has the effects:

$$\mathsf{E}_{\mathrm{mv}} = \{\langle CR, CR\rangle^-, \langle CR, \exists Right.CR\rangle^+\}.$$

The effects of two consecutive moves can be computed using regression and the accumulation operator. The regression of *CR* through E_{mv} is given by

$$\mathfrak{R}[CR, \mathsf{E}_{\mathrm{mv}}] = (CR \sqcap \neg CR \sqcup \exists Right.CR) \equiv \exists Right.CR.$$

For two consecutive moves we obtain

$$\mathsf{Regr}(\mathsf{E}_{\mathrm{mv}}, \mathsf{E}_{\mathrm{mv}}) = \Big\{\langle CR, \exists Right.CR\rangle^-, \big\langle CR, \big(\exists Right.(\exists Right.CR)\big)\big\rangle^+\Big\}, \text{ and}$$
$$\mathsf{Diff}^+(\mathsf{E}_{\mathrm{mv}}, \mathsf{E}_{\mathrm{mv}}) = \Big\{\big\langle CR, \big(\exists Right.CR \sqcap \neg\exists Right.CR\big)\big\rangle^+\Big\}.$$

The overall accumulation is given by

$$\mathsf{E}_{\mathrm{mv}} \bowtie \mathsf{E}_{\mathrm{mv}} = \Big\{\langle CR, CR\rangle^-, \langle CR, \exists Right.CR\rangle^-, \big\langle CR, \big(\exists Right.(\exists Right.CR)\big)\big\rangle^+\Big\}.$$

The positive effect of two consecutive moves on *CR* is defined with two nested existential restrictions. Intuitively, the concept describes exactly the room that is two steps away from the initial current room. In case of n consecutive moves the accumulation produces a nested existential restriction of depth n pointing to the room n steps away from the initial position. ▲

For the construction of a dynamic type for Σ_A we consider all possible changes caused by action sequences over $\boldsymbol{A}$. This means all possible accumulations of effect sets of the form $\Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$ with $\mathfrak{s} \in \mathfrak{S}_\mathcal{C}$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$ are relevant. Before we define the set of all relevant effects based on the accumulation operator a few more auxiliary definitions are needed.

We define a containment relation on sets of effects modulo equivalence.

Definition 5.13. Let E and E' be two sets of effects. We write $\mathsf{E} \Subset \mathsf{E}'$ iff the following conditions are satisfied:

- for any positive effects $\langle F,X\rangle^+ \in \mathsf{E}$ there exists a positive effect $\langle F,Y\rangle^+ \in \mathsf{E}'$ such that $X \equiv Y$ is valid;
- for any negative effect $\langle F,X\rangle^- \in \mathsf{E}$ there exists a negative effect $\langle F,Y\rangle^- \in \mathsf{E}'$ such that $X \equiv Y$ is valid.

Furthermore, we write $\mathsf{E} \equiv \mathsf{E}'$ if both directions $\mathsf{E} \Subset \mathsf{E}'$ and $\mathsf{E} \Supset \mathsf{E}'$ are satisfied. ▲

Next, we define *closure under accumulations* as a property of a set of effects.

Definition 5.14. Let $\boldsymbol{A}$ be a finite set of ground action terms, $\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_\mathcal{C})$ a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$, and $\mathfrak{E}$ a (possibly infinite) set of $\mathcal{DL}$-effects over $\mathcal{F}$.

We say that $\mathfrak{E}$ is *closed under accumulations w.r.t.* Σ_A iff for all *finite* subsets $\mathsf{E} \subseteq \mathfrak{E}$ and for all $(\mathfrak{s}, \boldsymbol{\alpha}) \in \mathfrak{S}_\mathcal{C} \times \boldsymbol{A}$ it holds that $(\mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})) \Subset \mathfrak{E}$. ▲

Note that $(\emptyset \bowtie \mathsf{E}) = \mathsf{E}$. Therefore, we have $\Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}) \Subset \mathfrak{E}$ for all $(\mathfrak{s}, \boldsymbol{\alpha}) \in \mathfrak{S}_\mathcal{C} \times \boldsymbol{A}$ if $\mathfrak{E}$ is closed under accumulations w.r.t. Σ_A. For a $\mathcal{DL}$-admissible representation

$$\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_\mathcal{C})$$

a set of effects that is closed under accumulations w.r.t. Σ_A always exists. To prove this we inductively define a sequence of sets of effects $\mathfrak{E}_0, \mathfrak{E}_1, \mathfrak{E}_2, \ldots$ as follows:

$$\mathfrak{E}_0 := \bigcup_{(\mathfrak{s},\boldsymbol{\alpha}) \in \mathfrak{S}_\mathcal{C} \times \boldsymbol{A}} \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha});$$

$$\mathfrak{E}_i := \mathfrak{E}_{i-1} \cup \bigcup_{(\mathfrak{s},\boldsymbol{\alpha}) \in \mathfrak{S}_\mathcal{C} \times \boldsymbol{A}} \Bigg(\bigcup_{\substack{\mathsf{E} \subseteq \mathfrak{E}_{i-1}, \\ \mathsf{E} \text{ is finite}}} (\mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})) \Bigg) \text{ for all i} > 0.$$

It can be shown that the set of effects given by

$$\bigcup_{i=0}^{\infty} \mathfrak{E}_i$$

is closed under accumulations w.r.t. Σ_A. The construction of dynamic types is based on an arbitrary but fixed set of effects that is closed under accumulations w.r.t. the $\mathcal{DL}$-admissible representation under consideration.

Definition 5.15. Let $\boldsymbol{A}$ be a finite set of ground action terms,

$$\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_\mathcal{C})$$

a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ and $\mathfrak{E}$ a set of effects over $\mathcal{F}$ that is closed under accumulations w.r.t. $\Sigma_{\boldsymbol{A}}$. Furthermore, let $\mathfrak{D}(\Sigma_{\boldsymbol{A}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K})\mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ be the FO-DS induced by $\Sigma_{\boldsymbol{A}}$.

A *dynamic type* $\mathfrak{t}$ *w.r.t.* $\Sigma_{\boldsymbol{A}}$ *and* $\mathfrak{E}$ is a set

$$\mathfrak{t} \subseteq \mathcal{C} \times \{\mathsf{E} \mid \mathsf{E} \subseteq \mathfrak{E}, \mathsf{E} \text{ is finite }\}$$

satisfying the following two conditions

- $\mathfrak{t}$ is *complete*: for all $\psi \in \mathcal{C}$ and all finite sets $\mathsf{E} \subseteq \mathfrak{E}$ it holds that $(\psi, \mathsf{E}) \in \mathfrak{t}$ or $(\neg\psi, \mathsf{E}) \in \mathfrak{t}$ (modulo elimination of double negation).
- $\mathfrak{t}$ is *realizable*: there exists an interpretation $\mathcal{I} \in \mathbb{I}$ such that for all $(\psi, \mathsf{E}) \in \mathfrak{t}$ it holds that $\mathcal{I}^{\mathsf{E}} \models \psi$.

The *set of all dynamic types w.r.t.* $\Sigma_{\boldsymbol{A}}$ *and* $\mathfrak{E}$ is denoted by $\mathsf{D\text{-}Types}(\Sigma_{\boldsymbol{A}}, \mathfrak{E})$.

Let $\mathcal{I} \in \mathbb{I}$ be an interpretation. The *dynamic type of* $\mathcal{I}$ *w.r.t.* $\Sigma_{\boldsymbol{A}}$ *and* $\mathfrak{E}$ is defined by

$$\mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{I}) := \{(\psi, \mathsf{E}) \mid \psi \in \mathcal{C}, \mathsf{E} \subseteq \mathfrak{E}, \mathsf{E} \text{ is finite}, \mathcal{I}^{\mathsf{E}} \models \psi\}.$$

▲

The completeness and realizability conditions for dynamic types imply that

$$\mathsf{D\text{-}Types}(\Sigma_{\boldsymbol{A}}, \mathfrak{E}) = \{\mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{I}) \mid \mathcal{I} \in \mathbb{I}\}.$$

If interpretations have same dynamic type, then all possible respective future evolutions of them have the same static type. This is the key property ensuring that executing action sequences from $\boldsymbol{A}^*$ in interpretations of the same dynamic type leads to indistinguishable observable behavior w.r.t. the given context.

Lemma 5.16. *Let $\boldsymbol{A}$ be a finite set of ground action terms, $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$, $\mathfrak{D}(\Sigma_{\boldsymbol{A}}) = (\mathcal{F}, \mathcal{K}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ the induced FO-DS and $\mathfrak{E}$ a set of effects over $\mathcal{F}$ that is closed under accumulations w.r.t. $\Sigma_{\boldsymbol{A}}$.*

For two interpretations $\mathcal{I}, \mathcal{J} \in \mathbb{I}$ and an action sequence $\sigma \in \boldsymbol{A}^*$ it holds that

$$\mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{I}) = \mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{J}) \textit{ implies } \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}') = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J}'), \textit{ where}$$

$\mathcal{I} \Rightarrow^{\sigma}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{I}'$ *and* $\mathcal{J} \Rightarrow^{\sigma}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{J}'$.

Proof. Let $\sigma = \boldsymbol{\alpha}_1\boldsymbol{\alpha}_2 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$, $\mathcal{I}_0, \mathcal{J}_0 \in \mathbb{I}$ with $\mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{I}_0) = \mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{J}_0)$ and consider the execution of σ in $\mathcal{I}_0$ and $\mathcal{J}_0$:

$$\mathcal{I}_0 \Rightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{I}_1 \Rightarrow^{\boldsymbol{\alpha}_2}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \cdots \Rightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{I}_n \text{ and } \mathcal{J}_0 \Rightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{J}_1 \Rightarrow^{\boldsymbol{\alpha}_2}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \cdots \Rightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}(\Sigma_{\boldsymbol{A}})} \mathcal{J}_n.$$

We show $\mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{I}_0) = \mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{J}_0)$ implies $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}_n) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J}_n)$ by induction on n. Let $n = 0$. It holds that

$$\psi \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}_0) \text{ iff } (\psi, \emptyset) \in \mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{I}_0) \text{ iff } (\psi, \emptyset) \in \mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_{\boldsymbol{A}}}(\mathcal{J}_0) \text{ iff } \psi \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J}_0).$$

Let $n > 0$ and assume that there are static types $\mathfrak{s}_0, \ldots, \mathfrak{s}_{n-1} \in \mathfrak{S}_{\mathcal{C}}$ such that

$$\mathfrak{s}_i = \text{s-type}_{\mathcal{C}}(\mathcal{I}_i) = \text{s-type}_{\mathcal{C}}(\mathcal{J}_i) \text{ for all } i = 0, \ldots, n-1. \tag{5.16}$$

The effects of executing σ in $\mathcal{I}_0$ and $\mathcal{J}_0$ can be describes by the following set:

$$\mathsf{E}_\sigma = \Sigma_{\boldsymbol{A}}(\mathfrak{s}_0, \boldsymbol{\alpha}_1) \bowtie (\Sigma_{\boldsymbol{A}}(\mathfrak{s}_1, \boldsymbol{\alpha}_2) \bowtie (\cdots \bowtie \Sigma_{\boldsymbol{A}}(\mathfrak{s}_{n-1}, \boldsymbol{\alpha}_n)))$$

With the induction hypothesis (5.16) and Lemma 5.9 it follows that

$$\mathcal{I}_n = \mathcal{I}_0{}^{\mathsf{E}_\sigma} \text{ and } \mathcal{J}_n = \mathcal{J}_0{}^{\mathsf{E}_\sigma}.$$

Since $\mathfrak{E}$ is closed under accumulations, it can be shown that $\mathsf{E}_\sigma \Subset \mathfrak{E}$. Let $\widehat{\mathsf{E}} \subseteq \mathfrak{E}$ be a set of effects with $\mathsf{E}_\sigma \equiv \widehat{\mathsf{E}}$. For all $\psi \in \mathcal{C}$:

$\psi \in \text{s-type}_{\mathcal{C}}(\mathcal{I}_n)$

iff $\mathcal{I}_0{}^{\mathsf{E}_\sigma} \models \psi$ (with $\mathcal{I}_n = \mathcal{I}_0{}^{\mathsf{E}_\sigma}$ and the definition of static types)

iff $\mathcal{I}_0{}^{\widehat{\mathsf{E}}} \models \psi$ (with $\mathsf{E}_\sigma \equiv \widehat{\mathsf{E}}$)

iff $(\psi, \widehat{\mathsf{E}}) \in \text{d-type}^{\mathfrak{E}}_{\Sigma_A}(\mathcal{I}_0)$ (definition of dynamic types and $\widehat{\mathsf{E}} \subseteq \mathfrak{E}, \psi \in \mathcal{C}$)

iff $(\psi, \widehat{\mathsf{E}}) \in \text{d-type}^{\mathfrak{E}}_{\Sigma_A}(\mathcal{J}_0)$ (assumption $\text{d-type}^{\mathfrak{E}}_{\Sigma_A}(\mathcal{I}_0) = \text{d-type}^{\mathfrak{E}}_{\Sigma_A}(\mathcal{J}_0)$)

iff $\mathcal{J}_0{}^{\widehat{\mathsf{E}}} \models \psi$ (definition of dynamic types and $\widehat{\mathsf{E}} \subseteq \mathfrak{E}, \psi \in \mathcal{C}$)

iff $\mathcal{I}_0{}^{\mathsf{E}_\sigma} \models \psi$ ($\mathsf{E}_\sigma \equiv \widehat{\mathsf{E}}$)

iff $\mathcal{J}_n \models \psi$ (with $\mathcal{J}_n = \mathcal{J}_0{}^{\mathsf{E}_\sigma}$)

iff $\psi \in \text{s-type}_{\mathcal{C}}(\mathcal{J}_n)$ (definition of static types).

□

5.2.2 Propositional Abstraction of DL-ConGolog Programs

In this section we show that the dynamic types defined in the previous subsection allow us to define a bisimilar (possibly infinite) propositional abstraction of the transition system induced by a $\mathcal{DL}$-ConGolog program. The abstraction we are going to define is based on the following components:

- a finite set $\boldsymbol{A}$ of ground action terms;
- an $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ of the form $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$;
- a pick-free program expression δ, where all action terms are from $\boldsymbol{A}$ and for all tests of the form $\psi?$ in δ it holds that ψ is a Boolean combination of axioms from $\mathcal{C}$;
- a set $\mathfrak{E}$ of effects over $\mathcal{F}$ that closed under accumulations w.r.t. $\Sigma_{\boldsymbol{A}}$.

The termination and failure actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ are handled in the same way as in the local effect case (see Section 4.3). To avoid additional notation we make the following assumptions regarding termination and failure: the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ are contained in $\boldsymbol{A}$, the concept names *Final* and *Fail* are contained in $\mathcal{F}$, the assertions $(\mathsf{prog} \sqsubseteq \mathit{Final})$, $(\mathsf{prog} \sqsubseteq \mathit{Fail})$ and there negations are contained in $\mathcal{C}$, and $\neg(\mathsf{prog} \sqsubseteq \mathit{Final})$ and $\neg(\mathsf{prog} \sqsubseteq \mathit{Fail})$ are contained in $\mathcal{K}$. The symbols $\boldsymbol{\epsilon}$, $\mathfrak{f}$, *Final* and *Fail* are not mentioned in δ. For all $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ we have $\Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\epsilon}) = \{\langle \mathit{Final}, \{\mathsf{prog}\}\rangle^+\}$ and $\Sigma_{\boldsymbol{A}}(\mathfrak{s}, \mathfrak{f}) = \{\langle \mathit{Fail}, \{\mathsf{prog}\}\rangle^+\}$. The names *Final* and *Fail* are not affected by any other actions in $\boldsymbol{A}$. In the following we directly refer to $\mathfrak{D}(\Sigma_{\boldsymbol{A}})$ instead of $\mathfrak{D}(\Sigma_{\boldsymbol{A}}) \uplus \{\boldsymbol{\epsilon}, \mathfrak{f}\}$.

An abstract state consists of the dynamic type of the interpretation in which the execution of δ was started, the set of effects of the action sequence that has been executed so far and the program that remains to be executed. Using the accumulation operator the program transition relation can be lifted to a relation between abstract states. The definition generalizes the construction for the local effect case (see Definition 4.13).

Definition 5.17. Let $\boldsymbol{A}$, $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$, δ and $\mathfrak{E}$ be as described above. Furthermore, let AP be a finite set of atomic propositions with a bijection $\iota : \mathcal{C} \to \mathsf{AP}$.

The *abstraction of* $\mathcal{P} = (\mathfrak{D}(\Sigma_{\boldsymbol{A}}), \delta)$ *w.r.t.* $\mathfrak{E}$ is a propositional transition system

$$\mathfrak{T}_{\mathcal{P},\mathfrak{E}} = (Q_{\mathcal{P},\mathfrak{E}}, I_{\mathcal{P},\mathfrak{E}}, \hookrightarrow_{\mathcal{P},\mathfrak{E}}, \lambda_{\mathcal{P},\mathfrak{E}})$$

over AP, where

- $Q_{\mathcal{P},\mathfrak{E}} := \mathsf{D\text{-}Types}(\Sigma_{\boldsymbol{A}}, \mathfrak{E}) \times \{\mathsf{E} \mid \mathsf{E} \subseteq \mathfrak{E}, \mathsf{E} \text{ is finite}\} \times \mathsf{sub}(\delta)$;
- $I_{\mathcal{P},\mathfrak{E}} := \big\{(\mathfrak{t}, \emptyset, \delta) \in Q_{\mathcal{P},\mathfrak{E}} \,\big|\, (\varphi, \emptyset) \in \mathfrak{t} \text{ for all } \varphi \text{ occurring in } \mathcal{K}\big\}$;
- $\hookrightarrow_{\mathcal{P},\mathfrak{E}} := \Big\{\big((\mathfrak{t}, \mathsf{E}, \rho), (\mathfrak{t}', \mathsf{E}', \rho')\big) \in Q_{\mathcal{P},\mathfrak{E}} \times Q_{\mathcal{P},\mathfrak{E}} \,\Big|\, \mathfrak{t} = \mathfrak{t}',\ \text{(i) or (ii)}\Big\}$ with
  - (i) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ such that $\mathfrak{a}$ is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C} \mid (\psi, \mathsf{E}) \in \mathfrak{t}\}$, and
  $$\mathsf{E}' \equiv \mathsf{E} \bowtie \Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$$
  and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$;
  - (ii) there is *no* guarded action contained in $\mathsf{head}(\rho)$ that is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C} \mid (\psi, \mathsf{E}) \in \mathfrak{t}\}$ and we have
  $$\mathsf{E}' \equiv \mathsf{E} \bowtie \Sigma_{\boldsymbol{A}}(\mathfrak{s}, \mathfrak{f})$$
  and $\rho = \rho'$;
- $\lambda_{\mathcal{P},\mathfrak{E}} : (\mathfrak{t}, \mathsf{E}, \rho) \mapsto \{\iota(\psi) \mid (\psi, \mathsf{E}) \in \mathfrak{t}\}$ for all $(\mathfrak{t}, \mathsf{E}, \rho) \in Q_{\mathcal{P},\mathfrak{E}}$.

▲

In the sequel we prove that the transition system $\mathfrak{I}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ induced by $\mathcal{P} = (\mathfrak{D}(\Sigma_{\boldsymbol{A}}), \delta)$ and the abstraction $\mathfrak{T}_{\mathcal{P},\mathfrak{E}} = (Q_{\mathcal{P},\mathfrak{E}}, I_{\mathcal{P},\mathfrak{E}}, \hookrightarrow_{\mathcal{P},\mathfrak{E}}, \lambda_{\mathcal{P},\mathfrak{E}})$ are $\mathcal{C}$-bisimilar. A binary relation

$$\simeq_{\mathcal{C}} \subseteq Q_{\mathcal{P}} \times Q_{\mathcal{P},\mathfrak{E}}$$

is defined as follows: we have

$$\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}, \rho') \text{ iff the following conditions are satisfied}$$

- $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from some initial state $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \in I_{\mathcal{P}}$ such that $\mathfrak{t} = \text{d-type}^{\mathfrak{C}}_{\Sigma_A}(\mathcal{I}_0)$ and $\mathcal{I} = \mathcal{I}_0{}^{\mathsf{E}}$, and
- $\rho = \rho'$.

Lemma 5.18. *$\mathfrak{T}_{\mathcal{P},\mathfrak{C}}$ is a bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. $\mathcal{C}$.*

Proof. We follow the lines of the proof of Lemma 4.16. First, we have to show that $\simeq_{\mathcal{C}}$ is a $\mathcal{C}$-bisimulation. Let $\mathbb{I}$ be the state space of $\mathfrak{D}(\Sigma_A)$ and ι the bijection between $\mathcal{C}$ and AP.

Assume $\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}, \rho)$ for two states $\langle \mathcal{I}, \sigma, \rho \rangle \in Q_{\mathcal{P}}$ and $(\mathfrak{t}, \mathsf{L}, \rho) \in Q_{\mathcal{P},\mathfrak{C}}$. There exists $\mathcal{I}_0 \in \mathbb{I}$ such that $\langle \mathcal{I}, \sigma, \rho \rangle$ is reachable from an initial state $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \in I_{\mathcal{P},\mathfrak{C}}$ such that

$$\text{d-type}^{\mathfrak{C}}_{\Sigma_A}(\mathcal{I}_0) = \mathfrak{t} \text{ and } \mathcal{I} = \mathcal{I}_0^{\mathsf{E}}.$$

It follows that

$$\lambda_{\mathcal{P},\mathfrak{C}}((\mathfrak{t}, \mathsf{E}, \rho)) = \{\iota(\psi) \mid \psi \in \text{s-type}_{\mathcal{C}}(\mathcal{I})\}.$$

Let

$$\mathfrak{s} = \{\psi \in \mathcal{C} \mid (\psi, \mathsf{E}) \in \mathfrak{t}\}.$$

$\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}, \rho)$ implies $\text{s-type}_{\mathcal{C}}(\mathcal{I}) = \mathfrak{s}$. We have to show that for every outgoing transition of $\langle \mathcal{I}, \sigma, \rho \rangle$ there is a matching transition from $(\mathfrak{t}, \mathsf{E}, \rho)$ leading to a state that is in $\simeq_{\mathcal{C}}$-relation with the successor of $\langle \mathcal{I}, \sigma, \rho \rangle$ and vice versa. As in the proof of Lemma 4.16 we distinguish three cases regarding the action history σ:

- the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ do not occur in σ;
- σ is of the form $\sigma_{\mathsf{p}} \cdot \sigma_{\mathsf{s}}$, where the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ do not occur in σ_{p} and the suffix is of the form $\sigma_{\mathsf{s}} \in \{\epsilon\}^*$ with $|\sigma_{\mathsf{s}}| \geq 1$;
- σ is of the form $\sigma_{\mathsf{p}} \cdot \sigma_{\mathsf{s}}$, where the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ do not occur in σ_{p} and the suffix is of the form $\sigma_{\mathsf{s}} \in \{\mathfrak{f}\}^*$ with $|\sigma_{\mathsf{s}}| \geq 1$.

According to Lemma 3.2 this covers all possible cases.

(I) Assume that the actions $\boldsymbol{\epsilon}$ and $\mathfrak{f}$ do not occur in σ. Consider a transition

$$\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle.$$

As shown in the proof of Lemma 4.16 it holds that $\boldsymbol{\alpha} \in A \setminus \{\mathfrak{f}\}$

iff there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ such that $\mathfrak{a}$ is executable in $\mathcal{I}$ and $\mathcal{I} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma_A)} \mathcal{I}'$ and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$

iff there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ such that $\mathfrak{a}$ is executable in $\mathfrak{s}$ and $\mathcal{I}' = \mathcal{I}^{\Sigma_A(\mathfrak{s},\boldsymbol{\alpha})}$ and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$.

And, we have $\boldsymbol{\alpha} = \mathfrak{f}$

iff there is no guarded action $\mathfrak{a} \in \mathsf{head}(\rho)$ that is executable in $\mathcal{I}$, $\rho' = \rho$ and $\mathcal{I} \Rightarrow^{\mathfrak{f}}_{\mathfrak{D}(\Sigma_A)} \mathcal{I}'$

iff there is no guarded action $\mathfrak{a} \in \mathsf{head}(\rho)$ that is executable in $\mathfrak{s}$, $\rho' = \rho$ and $\mathcal{I}' = \mathcal{I}^{\Sigma_A(\mathfrak{s},\mathfrak{f})}$.

Since $\mathfrak{E}$ is closed under accumulations w.r.t. Σ_A and $\mathsf{E} \subseteq \mathfrak{E}$, it holds that

$$\mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}) \Subset \mathfrak{E}.$$

Thus, there exists $\mathsf{E}' \subseteq \mathfrak{E}$ such that $\mathsf{E}' \equiv \mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$. Consequently, the existence of the transition $\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle$ implies

$$(\mathfrak{t}, \mathsf{E}, \rho) \hookrightarrow_{\mathcal{P},\mathfrak{E}} (\mathfrak{t}, \mathsf{E}', \rho') \text{ with } \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}', \rho').$$

Assume there is a transition $(\mathfrak{t}, \mathsf{E}, \rho) \hookrightarrow_{\mathcal{P},\mathfrak{E}} (\mathfrak{t}, \mathsf{E}', \rho')$ with $\mathsf{E}' \equiv \mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$ for some $\boldsymbol{\alpha} \in \boldsymbol{A}$. It follows that there is a transition

$$\langle \mathcal{I}, \sigma, \rho \rangle \hookrightarrow_{\mathcal{P}} \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle \text{ with } \langle \mathcal{I}', \sigma \cdot \boldsymbol{\alpha}, \rho' \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}', \rho').$$

(II) Assume $\sigma = \sigma_{\mathsf{p}} \cdot \sigma_{\mathsf{s}}$ where the suffix satisfies $\sigma_{\mathsf{s}} \in \{\epsilon\}^*$ and $|\sigma_{\mathsf{s}}| \geq 1$. It follows that $\rho = \langle\rangle$, $\mathsf{head}(\langle\rangle) = \{\epsilon\}$, $\mathcal{I} \models (\mathsf{prog} \sqsubseteq \mathit{Final})$ and $((\mathsf{prog} \sqsubseteq \mathit{Final}), \mathsf{E}) \in \mathfrak{t}$. The definition of $\hookrightarrow_{\mathcal{P}}$ and of $\hookrightarrow_{\mathcal{P},\mathfrak{E}}$ imply that $\langle \mathcal{I}, \sigma \cdot \epsilon, \langle\rangle \rangle$ and $(\mathfrak{t}, \mathsf{E}', \langle\rangle)$ with $\mathsf{E}' \equiv \mathsf{E} \bowtie \langle \mathit{Final}, \{\mathsf{prog}\}\rangle^+$ are the only successors of $\langle \mathcal{I}, \sigma, \rho \rangle$ and $(\mathfrak{t}, \mathsf{E}, \rho)$, respectively. Note that $((\mathsf{prog} \sqsubseteq \mathit{Final}), \mathsf{E}) \in \mathfrak{t}$ implies $\langle \mathit{Final}, \{\mathsf{prog}\}\rangle^+ \in \mathsf{E}$ which implies $\mathsf{E} \equiv \mathsf{E} \bowtie \langle \mathit{Final}, \{\mathsf{prog}\}\rangle^+$. It follows that

$$\langle \mathcal{I}, \sigma \cdot \epsilon, \langle\rangle \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}', \langle\rangle).$$

(III) Assume there is a suffix σ_{s} of σ with $\sigma_{\mathsf{s}} \in \{\mathfrak{f}\}^*$ and $|\sigma_{\mathsf{s}}| \geq 1$. As shown in proof of Lemma 4.16 it follows that $\langle \mathcal{I}, \sigma, \rho \rangle$ and $(\mathfrak{t}, \mathsf{E}, \rho)$ are failure states where $\mathfrak{f}$ is the only executable action. The execution of $\mathfrak{f}$ does not cause any changes. $\langle \mathcal{I}, \sigma \cdot \mathfrak{f}, \rho \rangle$ is the only successor state of $\langle \mathcal{I}, \sigma, \rho \rangle$. All successor states of $(\mathfrak{t}, \mathsf{E}, \rho)$ are of the form $(\mathfrak{t}, \mathsf{E}', \rho)$ with $\mathsf{E} \equiv \mathsf{E}'$ and there exists at least one successor. We have that $\mathsf{E}' \equiv \mathsf{E}$ and $\langle \mathcal{I}, \sigma, \rho \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}, \rho)$ implies

$$\langle \mathcal{I}, \sigma \cdot \mathfrak{f}, \rho \rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathsf{E}', \rho).$$

We have $\langle \mathcal{I}_0, \langle\rangle, \delta \rangle \simeq_{\mathcal{C}} (\mathsf{d\text{-}type}^{\mathfrak{E}}_{\Sigma_A}(\mathcal{I}_0), \emptyset, \delta)$ for all initial interpretations $\mathcal{I}_0 \in \mathbb{I}_{\mathsf{ini}}$ in $\mathfrak{D}(\Sigma_A)$. $\square$

5.3 Decidable Fragments of DL-ConGolog

In general the propositional bisimilar abstraction of a DL-ConGolog defined in the previous section has infinitely many states because there can be infinitely many dynamic types. In this section we consider two syntactical restrictions on the underlying DL-admissible representation that allow us to regain decidability via a construction of a *finite* bisimilar abstraction. The first one (defined in Section 5.3.1) only allows acyclic dependencies between fluents in the effect representation. The second one (Section 5.3.2) restricts the DL syntax for defining the add-sets and delete-sets to a sub-DL where essentially no quantifiers are involved.

Both classes of DL-admissible representations are incomparable regarding expressiveness but include the class of local effect representations. In both cases we are able to show that a finite set of all relevant effect descriptions can be computed. This leads to an effective way of computing a finite bisimilar abstraction, which in turn makes DL-CTL* verification decidable. In this section we always use $\mathcal{DL}$ as the base logic.

5.3.1 An Acyclicity Condition

In Example 5.1 we have defined the ground action `move-fwd` describing the ability of a mobile robot to move to the room that is adjacent to the room the robot is currently located in. To define the current room after doing `move-fwd` we refer to the current room before doing the action. Thus, in the effect representation there is a cyclic dependency involving the concept name *CurrentRoom*.

Given a DL-admissible representation for a finite set of ground action we define a directed graph that captures such dependencies between the relevant fluents.

Definition 5.19. Let $\boldsymbol{A}$ be a finite set of ground action terms and $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$. The *dependency graph of* $\Sigma_{\boldsymbol{A}}$, denoted by $\mathcal{G}(\Sigma_{\boldsymbol{A}}) = (\mathcal{F}, E)$, is a finite directed graph where

- the set of nodes is the set of relevant fluents $\mathcal{F}$ and
- $E \subseteq \mathcal{F} \times \mathcal{F}$ is the set of edges such that $(F, F') \in E$ iff F' occurs in $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$ or in $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$ for some $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ and some $\boldsymbol{\alpha} \in \boldsymbol{A}$.

We say that $\Sigma_{\boldsymbol{A}}$ is *acyclic* iff the dependency graph $\mathcal{G}(\Sigma_{\boldsymbol{A}})$ is acyclic.

For an acyclic graph $\mathcal{G}(\Sigma_{\boldsymbol{A}}) = (\mathcal{F}, E)$ and $F \in \mathcal{F}$ the length of the longest path in $\mathcal{G}(\Sigma_{\boldsymbol{A}})$ starting in F is called the *depth of F in $\Sigma_{\boldsymbol{A}}$* and is denoted by $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(F)$. The corresponding *depth of an acyclic $\mathcal{DL}$-admissible representation* $\Sigma_{\boldsymbol{A}}$ is given by

$$\mathsf{depth}(\Sigma_{\boldsymbol{A}}) := \max(\{\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(F) \mid F \in \mathcal{F}\}).$$

▲

If the actions in $\boldsymbol{A}$ only have local effects, then $\mathcal{G}(\Sigma_{\boldsymbol{A}})$ is obviously acyclic. There are no dependencies at all because in the local effect case disjunctions of nominal concepts and disjunctions of nominal roles are sufficient to define the effects.

Example 5.20. First, we consider the domain with devices and batteries from Example 2.48. There, preconditions and effects are defined using a DL-action theory. The corresponding DL-admissible representation is defined as given in the proof of Lemma 2.33. The (acyclic) dependency graph consists of the following edges due to the non-local effect of `discharged`(bat) on the concept name *On*:

$$(On, EDev), (On, ConTo), (On, PowerS).$$

On has depth one and all other fluents have depth zero.

The graph of the action theory in Example 5.1 contains a self-loop on the concept name *CurrentRoom*. ▲

Now, we consider $\mathcal{DL}$-ConGolog programs over ground actions with an underlying *acyclic* $\mathcal{DL}$-admissible representation. In the following we show that the acyclicity allows us to obtain a finite set of effect descriptions that is closed under accumulations. First, some more auxiliary notions are defined.

Definition 5.21. Let $\boldsymbol{A}$ be a finite set of ground action terms and $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ an acyclic $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$. For a concept or role X over $\mathcal{F}$ we define the *depth of X w.r.t. $\Sigma_{\boldsymbol{A}}$* by

$$\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(X) := \max(\{\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(F) \mid F \in \mathcal{F} \text{ and } F \text{ occurs in } X\}).$$

For a set of effects E over $\mathcal{F}$ and a number n the set $\mathsf{E}_{\leq n}$ is the restriction of E to effects on fluents with a depth smaller or equal n. For $F \in \mathcal{F}$ the *sets of all relevant (positive and negative) effect descriptors* are given by

$$\begin{aligned}\mathsf{Pos}(F) &:= \{X \mid \langle F,X\rangle^+ \in \Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha}) \text{ for some } \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}} \text{ and some } \boldsymbol{\alpha} \in \boldsymbol{A}\}\\ \mathsf{Neg}(F) &:= \{X \mid \langle F,X\rangle^- \in \Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha}) \text{ for some } \mathfrak{s} \in \mathfrak{S}_{\mathcal{C}} \text{ and some } \boldsymbol{\alpha} \in \boldsymbol{A}\}.\end{aligned}$$

▲

The next lemma follows directly from the definitions.

Lemma 5.22. *Let $\boldsymbol{A}$ be a finite set of ground action terms and $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ an acyclic $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$.*

1. *For a fluent $F \in \mathcal{F}$ and $X \in \mathsf{Pos}(F) \cup \mathsf{Neg}(F)$ it holds that $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(F) > \mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(X)$ if $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(F) > 0$ and $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(X) = 0$ if $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(F) = 0$.*
2. *Let X be a concept or role over $\mathcal{F}$ with $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(X) = n$ and E a set of effects over $\mathcal{F}$. It holds that $\mathfrak{R}[X, \mathsf{E}] = \mathfrak{R}[X, \mathsf{E}_{\leq n}]$ and $\mathsf{depth}_{\Sigma_{\boldsymbol{A}}}(\mathfrak{R}[X, \mathsf{E}]) = n$.*

Due to the acyclicity condition it is ensured that for the definition of the effects on a fluent F only fluents with a depth strictly smaller than the depth of F are used. Therefore, regression does not increase the depth of a concept or role.

Next, we define the (finite) *set of all relevant effect descriptions*, where "relevant" refers to closure under accumulations. The set is constructed stepwise. First, all relevant effects on fluents of depth zero are defined because these fluents do not depend on other fluents. Given this set, we can proceed with the fluents of depth one because they only depend on fluents of depth zero. This is continued until we reach the maximum depth determined by the acyclic dependency graph of the underlying effect representation.

Definition 5.23. Let $\boldsymbol{A}$ be a finite set of ground action terms and

$$\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$$

an acyclic $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ and $\mathsf{depth}(\Sigma_{\boldsymbol{A}}) = n$ for some $n \geq 0$. We inductively define a sequence of length n of sets of effect descriptions over $\mathcal{F}$ denoted by

$$\mathfrak{E}^0, \mathfrak{E}^1, \ldots, \mathfrak{E}^n.$$

such that the set $\mathfrak{E}^i$ for some $0 \leq i \leq n$ represents the set of all relevant effects on fluents with a depth $\leq i$. For $i = 0$ we define

$$\mathfrak{E}^0 := \mathsf{Add}^0 \cup \mathsf{Del}^0,$$

where Add^0 is the set of all add-effects on fluents of depth zero:

$$\mathsf{Add}^0 := \left\{ \left\langle F, X \sqcap \bigsqcap_{Y \in \mathcal{D}} \neg Y \right\rangle^+ \;\middle|\; F \in \mathcal{F}, \mathsf{depth}_{\Sigma_A}(F) = 0, X \in \mathsf{Pos}(F), \mathcal{D} \subseteq \mathsf{Neg}(F) \right\},$$

and Del^0 the corresponding set of all delete-effects:

$$\mathsf{Del}^0 = \left\{ \langle F, X \rangle^- \;\middle|\; F \in \mathcal{F}, \mathsf{depth}_{\Sigma_A}(F) = 0, X \in \mathsf{Neg}(F) \right\}.$$

Next, let $0 < i \leq n$ and assume $\mathfrak{E}^{i-1}$ is already defined. The set $\mathfrak{E}^i$ is defined as follows:

$$\mathfrak{E}^i := \mathfrak{E}^{i-1} \cup \mathsf{Add}^i \cup \mathsf{Del}^i,$$

where

$$\mathsf{Add}^i := \left\{ \left\langle F, \mathfrak{R}[X, \mathsf{E}] \sqcap \bigsqcap_{(Y,\mathsf{E}') \in \mathcal{R}} \neg \mathfrak{R}[Y, \mathsf{E}'] \right\rangle^+ \;\middle|\; \begin{array}{l} \mathsf{depth}_{\Sigma_A}(F) = i, X \in \mathsf{Pos}(F), \mathsf{E} \subseteq \mathfrak{E}^{i-1}, \\ \mathcal{R} \subseteq \left(\mathsf{Neg}(F) \times \mathfrak{E}^{i-1}\right) \end{array} \right\}$$

and

$$\mathsf{Del}^i := \{\langle F, \mathfrak{R}[X, \mathsf{E}] \rangle^- \mid \mathsf{depth}_{\Sigma_A}(F) = i, X \in \mathsf{Neg}(F), \mathsf{E} \subseteq \mathfrak{E}^{i-1}\}.$$

The *set of all relevant effects* is given by $\mathfrak{E} := \mathfrak{E}^n$. In the definition of Add^0 and Add^i, $i > 0$ the conjuncts

$$\bigsqcap_{Y \in \mathcal{D}} \neg Y \text{ and } \bigsqcap_{(Y,\mathsf{E}') \in \mathcal{R}} \neg \mathfrak{R}[Y, \mathsf{E}'],$$

respectively, are omitted in case $\mathcal{D} = \emptyset$ and $\mathcal{R} = \emptyset$, respectively. ▲

The definition of the set $\mathfrak{E}$ is constructive. Given an acyclic $\mathcal{DL}$-admissible representation $\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_\mathcal{C})$, it can be effectively computed. $\mathfrak{E}$ is closed under accumulations.

Lemma 5.24. *Let A be a finite set of ground action terms and $\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_\mathcal{C})$ an acyclic $\mathcal{DL}$-admissible representation of A and let $\mathfrak{E}$ be the set of all relevant effects as defined above.*

It holds that $\mathfrak{E}$ is closed under accumulations w.r.t. Σ_A.

Proof. Let $\mathsf{depth}(\Sigma_A) = n$ for some $n \geq 0$. Let $\mathfrak{s} \in \mathfrak{S}_\mathcal{C}$, $\boldsymbol{\alpha} \in A$ and $\mathsf{E} \subseteq \mathfrak{E}$. We consider an effect $\mathsf{e} \in (\mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}))$ and show that there is an equivalent effect in $\mathfrak{E}$. According to Definition 5.8 we have

$$(\mathsf{E} \bowtie \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})) := \mathsf{Regr}(\Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}), \mathsf{E}) \cup \mathsf{Diff}^+(\mathsf{E}, \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})) \cup \{\langle F, X \rangle^- \in \mathsf{E}\}.$$

The case $\mathsf{e} \in \{\langle F, X\rangle^- \in \mathsf{E}\}$ is trivial. We go through the remaining cases.

Case 1: e is of the form $\langle F, X\rangle^\pm$ with

$$\mathsf{depth}_{\Sigma_A}(F) = 0 \text{ and } \langle F, X\rangle^\pm \in \mathsf{Regr}(\Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}), \mathsf{E}).$$

There exists $\langle F, Y\rangle^\pm \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$ such that $\langle F, X\rangle^\pm = \langle F, \mathfrak{R}[\mathsf{E}, Y]\rangle^\pm$. It follows that $\mathsf{depth}_{\Sigma_A}(Y) = 0$ with Lemma 5.22. Consequently, we have $\mathfrak{R}[\mathsf{E}, Y] = Y = X$. Since $\langle F, Y\rangle^\pm \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$ it follows that $Y \in \mathsf{Pos}(F) \cup \mathsf{Neg}(F)$ and $\langle F, Y\rangle^\pm \in \mathfrak{E}^0$. Hence, $\mathsf{e} \in \mathfrak{E}^0$.

Case 2: e is a positive effect of the form $\langle F, X\rangle^+$ with

$$\mathsf{depth}_{\Sigma_A}(F) = 0 \text{ and } \langle F, X\rangle^+ \in \mathsf{Diff}^+(\mathsf{E}, \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}))$$

There exists $\left\langle F, X'\right\rangle^+ \in \mathsf{E}$ such that

$$X = X' \sqcap \bigsqcap_{\langle F, Y\rangle^- \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})} \neg \mathfrak{R}[Y, \mathsf{E}].$$

$\mathsf{depth}_{\Sigma_A}(F) = 0$ and Lemma 5.22 yield

$$X = X' \sqcap \bigsqcap_{\langle F, Y\rangle^- \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})} \neg Y. \tag{5.17}$$

Since by assumption $\left\langle F, X'\right\rangle^+ \in \mathsf{E} \subseteq \mathfrak{E}$ and $\mathsf{depth}_{\Sigma_A}(F) = 0$, it follows that

$$\left\langle F, X'\right\rangle^+ \in \mathfrak{E}^0.$$

Therefore, by definition of $\mathfrak{E}^0$ there exists $\widehat{X} \in \mathsf{Pos}(F)$ and a set $\mathcal{D} \subseteq \mathsf{Neg}(F)$ such that

$$X' = \widehat{X} \sqcap \bigsqcap_{\widehat{Y} \in \mathcal{D}} \neg \widehat{Y}. \tag{5.18}$$

Let $\mathcal{D}' = \mathcal{D} \cup \{Y \mid \langle F, Y\rangle^- \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})\}$. Obviously, $\mathcal{D}' \subseteq \mathsf{Neg}(F)$. With (5.17) and (5.18) it follows that

$$X \equiv \widehat{X} \sqcap \bigsqcap_{\widehat{Y}' \in \mathcal{D}'} \neg \widehat{Y}' \text{ and } \left\langle F, \widehat{X} \sqcap \bigsqcap_{\widehat{Y}' \in \mathcal{D}'} \neg \widehat{Y}' \right\rangle^+ \in \mathfrak{E}^0.$$

Case 3: e is of the form $\langle F, X\rangle^\pm$ with

$$\mathsf{depth}_{\Sigma_A}(F) = f, 0 < f \leq n \text{ and } \langle F, X\rangle^\pm \in \mathsf{Regr}(\Sigma_A(\mathfrak{s}, \boldsymbol{\alpha}), \mathsf{E}).$$

There exists $\langle F, Y\rangle^\pm \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$ such that $X = \mathfrak{R}[Y, \mathsf{E}]$. It follows with Lemma 5.22 that $\mathsf{depth}_{\Sigma_A}(Y) \leq f - 1$. We obtain

$$X = \mathfrak{R}[Y, \mathsf{E}] = \mathfrak{R}[Y, \mathsf{E}_{\leq f-1}].$$

The assumption $\mathsf{E} \subseteq \mathfrak{E}$ implies $\mathsf{E}_{\leq f-1} \subseteq \mathfrak{E}^{f-1}$. Since $Y \in \mathsf{Pos}(F) \cup \mathsf{Neg}(F)$ we get

$$\left\langle F, \mathfrak{R}[Y, \mathsf{E}_{\leq f-1}]\right\rangle^{\pm} \in \mathfrak{E}^{f}.$$

Case 4: e is of the form $\langle F, X\rangle^{+}$ with

$$\mathsf{depth}_{\Sigma_A}(F) = f, 0 < f \leq n \text{ and } \langle F, X\rangle^{+} \in \mathsf{Diff}^{+}(\mathsf{E}, \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})).$$

There exists $\left\langle F, X'\right\rangle^{+} \in \mathsf{E}$ such that

$$X = X' \sqcap \bigsqcap_{\langle F, Y\rangle^{-} \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})} \neg \mathfrak{R}[Y, \mathsf{E}].$$

With Lemma 5.22 and $\mathsf{depth}_{\Sigma_A}(Y) \leq f-1$ for all $\langle F, Y\rangle^{-} \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})$ we obtain

$$X = X' \sqcap \bigsqcap_{\langle F, Y\rangle^{-} \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})} \neg \mathfrak{R}[Y, \mathsf{E}_{\leq f-1}]. \tag{5.19}$$

$\left\langle F, X'\right\rangle^{+} \in \mathsf{Add}^{f} \cap \mathfrak{E}$ implies that there are

$$Z \in \mathsf{Pos}(F), \mathsf{E}_Z \subseteq \mathfrak{E}^{f-1} \text{ and } \mathcal{R} \subseteq (\mathsf{Neg}(F) \times \mathfrak{E}^{f-1})$$

such that

$$X' = \mathfrak{R}[Z, \mathsf{E}_Z] \sqcap \bigsqcap_{(\widehat{Z}, \widehat{E}) \in \mathcal{R}} \neg \mathfrak{R}[\widehat{Z}, \widehat{E}]. \tag{5.20}$$

Let

$$\mathcal{R}' = \mathcal{R} \cup \{(Y, \mathsf{E}_{\leq f-1}) \mid \langle F, Y\rangle^{-} \in \Sigma_A(\mathfrak{s}, \boldsymbol{\alpha})\}.$$

With (5.19) and (5.20) it follows that

$$X \equiv \mathfrak{R}[Z, \mathsf{E}_Z] \sqcap \bigsqcap_{(\widehat{Z}', \widehat{E}') \in \mathcal{R}'} \neg \mathfrak{R}[\widehat{Z}', \widehat{E}'] \text{ and}$$

$$\left\langle F, \mathfrak{R}[Z, \mathsf{E}_Z] \sqcap \bigsqcap_{(\widehat{Z}', \widehat{E}') \in \mathcal{R}'} \neg \mathfrak{R}[\widehat{Z}', \widehat{E}'] \right\rangle^{+} \in \mathfrak{E}^{f}.$$

□

Since $\mathfrak{E}$ is finite and computable the set of all dynamic types w.r.t. Σ_A and $\mathfrak{E}$ is finite and computable as well. All complete subsets of $\mathcal{C} \times 2^{\mathfrak{E}}$ can be enumerated. Checking realizability can be reduced to a decidable $\mathcal{DL}$-consistency problem using the regression operator.

Lemma 5.25. *Let $\boldsymbol{A}$ be a finite set of ground action terms and $\Sigma_A = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^{+}, \mathsf{E}^{-}, \mathsf{Pre}_{\mathcal{C}})$ an acyclic $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ and let $\mathfrak{E}$ be the set of all relevant effects as defined above.*

The set of all dynamic types w.r.t. $\Sigma_{\boldsymbol{A}}$ and $\mathfrak{E}$ is effectively computable.

Proof. $\mathfrak{E}$ is a finite set. Let $\mathfrak{t} \subseteq \mathcal{C} \times 2^{\mathfrak{E}}$ be such that $(\psi, \mathsf{E}) \in \mathfrak{t}$ or $(\neg\psi, \mathsf{E}) \in \mathfrak{t}$ for all $\psi \in \mathcal{C}$ and all $\mathsf{E} \subseteq \mathfrak{E}$. It holds that $\mathfrak{t}$ is realizable iff the following Boolean $\mathcal{DL}$-KB has a model:

$$\bigwedge_{(\varphi,\mathsf{E}) \in \mathfrak{t}} \mathfrak{R}[\varphi, \mathsf{E}].$$

Thus, the problem whether a given subset of $\mathcal{C} \times 2^{\mathfrak{E}}$ is realizable is decidable. Consequently, $\mathsf{D\text{-}Types}(\Sigma_{\boldsymbol{A}}, \mathfrak{E})$ is computable. □

We consider the verification problem where the input consists of the following components:

- a finite set $\boldsymbol{A}$ of ground action terms;
- an acyclic $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ of the form $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$;
- a pick-free program expression δ, where all action terms are from $\boldsymbol{A}$ and for all tests of the form $\psi?$ in δ we have $\psi \in \mathcal{C}$;
- a $\mathcal{DL}$-CTL* state formula Φ over axioms from $\mathcal{C}$.

We want to check whether Φ is valid in $\mathcal{P} = (\mathfrak{D}(\Sigma_{\boldsymbol{A}}), \delta)$. Note that $\mathcal{C}$ is a proper $\mathcal{DL}$-context for $\mathcal{P}$ according to Definition 3.16.

Theorem 5.26. *Verifying $\mathcal{DL}$-CTL* properties of $\mathcal{DL}$-ConGolog programs over ground actions with an acyclic $\mathcal{DL}$-admissible representation is decidable.*

Proof. We can compute a finite set of effects $\mathfrak{E}$ that is closed under accumulations w.r.t. $\Sigma_{\boldsymbol{A}}$. Thus, according to Lemma 5.25 the set of all dynamic types w.r.t. $\Sigma_{\boldsymbol{A}}$ and $\mathfrak{E}$ is computable. Therefore, a finite bisimilar propositional abstraction of the program is effectively computable: The finite set of all abstract states $\mathsf{D\text{-}Types}(\Sigma_{\boldsymbol{A}}, \mathfrak{E}) \times 2^{\mathfrak{E}} \times \mathsf{sub}(\delta)$ can be enumerated. By assumption on $\Sigma_{\boldsymbol{A}}$ the effects of a ground action given a static type is computable. The result of the accumulation operator on two finite sets of effects is computable as well. Furthermore, checking equivalence of $\mathcal{DL}$-concept and $\mathcal{DL}$-roles is decidable and the head and tail functions are decidable as well. It follows that the transition relation on the set of abstract states is decidable. Obviously, the label of the states is also computable. Thus, the verification problem boils down to a decidable model checking problem. □

The result strengthens the decidability result formulated in Theorem 4.18 for programs over local effect actions. Note that actions with only local effects do not cause any dependencies between fluents. The model used in Example 2.48 includes an action with non-local effects but it only has acyclic dependencies. A classical example domain from the literature that has non-local effect actions is the *logistics domain*. It was introduced in [Bac01] and is designed for planning benchmarks. In [Yeh+12] a Situation Calculus axiomatization of the domain is used for experiments with a projection solver. It considers an open-world setting where different kinds of vehicles transport items between different locations. The Situation Calculus theory in [Yeh+12] uses a DL as its base logic and can be viewed as an acyclic $\mathcal{DL}$-admissible representation. We consider a simplified version of this domain in the following example.

Example 5.27. A $\mathcal{DL}$-action theory for a simple transportation domain is described. There are items that are loaded into at most one box (related via the role name *Loaded*). Boxes and items are located in at most one room (via the role name *In*). The TBox consists of the following role inclusions and concept inclusions:

$$\begin{aligned}\mathcal{T} = \{&\mathit{Loaded} \sqsubseteq \mathit{Item} \times \mathit{Box}, \quad \mathit{Item} \sqsubseteq {\leq}1\,\mathit{Loaded}.\mathit{Box}, \quad \mathit{Box} \sqcap \mathit{Item} \sqsubseteq \bot\\ &\mathit{In} \sqsubseteq (\mathit{Box} \sqcup \mathit{Item}) \times \mathit{Room}, \quad \mathit{Box} \sqcup \mathit{Item} \sqsubseteq {\leq}1\,\mathit{In}.\mathit{Room}\}.\end{aligned}$$

An initial situation with two named rooms, two named boxes and one named item is described using the following ABox assertions:

$$\mathcal{A} = \{\mathsf{b}_1 \in (\mathit{Box} \sqcap \exists \mathit{In}.\{\mathsf{r}_1\}), \quad \mathsf{b}_2 \in (\mathit{Box} \sqcap \exists \mathit{In}.\{\mathsf{r}_2\}), \quad \mathsf{i} \in (\mathit{Item} \sqcap \exists \mathit{In}.\{\mathsf{r}_1\} \sqcap \forall \mathit{Loaded}.\bot)\}.$$

We define the ground action $\texttt{move-to}(\mathsf{b}_1,\mathsf{r}_1,\mathsf{r}_2)$ for moving the box b_1 from room r_1 to room r_2 as follows:

$$\begin{aligned}\mathsf{pre}(\texttt{move-to}(\mathsf{b}_1,\mathsf{r}_1,\mathsf{r}_2)) &:= \{(\mathsf{b}_1,\mathsf{r}_1) \in \mathit{In}\};\\ \mathsf{eff}(\texttt{move-to}(\mathsf{b}_1,\mathsf{r}_1,\mathsf{r}_2)) &:= \{\, \langle \mathit{In}, (\{\mathsf{b}_1\} \sqcup \exists \mathit{Loaded}.\{\mathsf{b}_1\}) \times \{\mathsf{r}_1\}\rangle^-,\\ &\qquad \langle \mathit{In}, (\{\mathsf{b}_1\} \sqcup \exists \mathit{Loaded}.\{\mathsf{b}_1\}) \times \{\mathsf{r}_2\}\rangle^+\}.\end{aligned}$$

The concept $\{\mathsf{b}_1\} \sqcup \exists \mathit{Loaded}.\{\mathsf{b}_1\}$ describes the box b_1 itself and the set of loaded items inside b_1. Both, the box and its content move to the new room r_2. The actions $\texttt{move-to}(\mathsf{b}_1,\mathsf{r}_2,\mathsf{r}_1)$, $\texttt{move-to}(\mathsf{b}_2,\mathsf{r}_1,\mathsf{r}_2)$ and $\texttt{move-to}(\mathsf{b}_2,\mathsf{r}_2,\mathsf{r}_1)$ are defined similarly. There is another action $\texttt{load}(\mathsf{i},\mathsf{b}_1)$ for loading the item i into the box b_1:

$$\begin{aligned}\mathsf{pre}(\texttt{load}(\mathsf{i},\mathsf{b}_1)) &:= \{(\top \sqsubseteq {\leq}1\,U.(\exists \mathsf{inv}(\mathit{In}).\{\mathsf{i}\} \sqcup \exists \mathsf{inv}(\mathit{In}).\{\mathsf{b}_1\})\};\\ \mathsf{eff}(\texttt{load}(\mathsf{i},\mathsf{b}_1)) &:= \{\langle \mathit{Loaded}, \{(\mathsf{i},\mathsf{b}_1)\}\rangle^+\}.\end{aligned}$$

The concept $\exists \mathsf{inv}(\mathit{In}).\{\mathsf{i}\}$ describes the set of rooms the item i is located in and $\exists \mathsf{inv}(\mathit{In}).\{\mathsf{b}_1\}$ the rooms of b_1. Note that in models of the TBox both concepts are interpreted as singleton sets. The precondition expresses that the *disjunction* of both concepts is a singleton set using an at most restriction on the universal role. Thus, the precondition is satisfied if both objects are located in the same room. The preconditions and effects of the ground action term $\texttt{load}(\mathsf{i},\mathsf{b}_2)$ are defined analogously. The $\mathcal{DL}$-admissible representation given by $\mathcal{T}$, $\mathcal{A}$, pre, and eff for the set of ground actions described above is acyclic. There is only a single dependency between *In* and *Loaded*. The action theory has depth one. For instance, one could verify that the TBox $\mathcal{T}$ is preserved by any ground action sequence. ▲

The cycle caused by the action `move-fwd` in Example 5.1 is avoided in the example above by requiring that the starting point and the target of the move are explicitly named in the argument of the move action. The consequence is that the moves are restricted to an a priori fixed finite set of named rooms whereas in Example 5.1 moves to an unbounded number of unknown rooms are possible as well.

5.3.2 Flat Effect Representations

In this section we introduce another decidable fragment of $\mathcal{DL}$-ConGolog over actions with possibly non-local effects. It is based on the observation that cyclic dependencies between fluents are only problematic if quantifiers are involved. We define *flat $\mathcal{DL}$*-admissible representations, where the definitions of the add-sets and delete-sets are formulated in the quantifier-free fragment of FO^2. In this restricted case it is possible to show that only a finite number of effect descriptions is relevant. Once we have shown this, the abstraction technique in 5.2 yields a finite bisimilar abstraction.

We consider an example with “flat" cyclic dependencies between fluents.

Example 5.28. We extend the domain in Example 5.27 with an additional action. We consider the action of pouring the content of b_1 into b_2. It is defined as follows:

$$\begin{aligned}\mathsf{pre}(\mathtt{pour}(\mathsf{b}_1,\mathsf{b}_2)) &:= \{\top \sqsubseteq {\leq}1\,U.(\exists\mathsf{inv}(In).\{\mathsf{b}_1\} \sqcup \exists\mathsf{inv}(In).\{\mathsf{b}_2\})\}\\ \mathsf{eff}(\mathtt{pour}(\mathsf{b}_1,\mathsf{b}_2)) &:= \{\,\langle Loaded, \exists Loaded.\{\mathsf{b}_1\} \times \{\mathsf{b}_1\}\rangle^-\\ &\qquad \langle Loaded, \exists Loaded.\{\mathsf{b}_1\} \times \{\mathsf{b}_2\}\rangle^+\}\end{aligned}$$

The precondition says that b_1 and b_2 have to be in the same room. The effect is that b_1 is empty and all items that were contained in b_1 *before* doing the action are contained in b_2 *after* doing the action. In the fluent dependency graph the move action leads to an edge from *In* to *Loaded* and *Loaded* has a self-loop attached to it due to the non-local effect of $\mathtt{pour}(\mathsf{b}_1,\mathsf{b}_2)$. However, all effect definitions in this example can be defined in FO^2 without quantifiers. For instance, the role $\exists Loaded.\{\mathsf{b}_1\} \times \{\mathsf{b}_2\}$ can be equivalently formulated as the quantifier-free formula

$$Loaded(x,\mathsf{b}_1) \wedge y \approx \mathsf{b}_2$$

in FO syntax. (Note that it is not possible to formulate effects of the `move-fwd` in Example 5.1 without quantifiers.) ▲

First, a sublogic called $\mathcal{DL}_{\mathsf{QF}}$ of $\mathcal{DL}$ is defined. It corresponds to the quantifier-free fragment of FO^2 (two-variable fragment of FO).

Definition 5.29. *$\mathcal{DL}_{\mathsf{QF}}$-concepts* C and *$\mathcal{DL}_{\mathsf{QF}}$-roles* R are built according to the following syntax rules

$$\begin{aligned}C &::= \top \mid \{o\} \mid A \mid \exists P.\{o\} \mid \exists\mathsf{inv}(P).\{o\} \mid \exists(\mathsf{id} \sqcap P).\top \mid \forall U.(\neg\{o\} \sqcup C) \mid \neg C \mid C \sqcap C,\\ R &::= \mathsf{id} \mid P \mid \mathsf{inv}(P) \mid \top \times C \mid C \times \top \mid \neg R \mid R \sqcap R.\end{aligned}$$

where $o \in \mathsf{N_O}$, $A \in \mathsf{N_C}$ and $P \in \mathsf{N_R}$. Disjunction of concepts and roles and other Boolean connectors are obtained as usual. The set of all *atomic $\mathcal{DL}_{\mathsf{QF}}$-concepts* is the smallest set satisfying the following conditions:

- every concept of the form $\top, \{o\}, A, \exists P.\{o\}, \exists\mathsf{inv}(P).\{o\}$ and $\exists(\mathsf{id} \sqcap P).\top$ is atomic, and
- if B is of the form $\top, \{o\}, A, \exists P.\{o\}, \exists\mathsf{inv}(P).\{o\}$ or $\exists(\mathsf{id} \sqcap P).\top$ and $o \in \mathsf{N_O}$, then also $\forall U.(\neg\{o\} \sqcup B)$ is atomic.

Atomic $\mathcal{DL}_{\mathsf{QF}}$-roles are of the form id, P, $\mathsf{inv}(P)$ or $\top \times B$ or $B \times \top$, where B stands for an atomic $\mathcal{DL}_{\mathsf{QF}}$-concept. ▲

$\mathcal{DL}_{\mathsf{QF}}$-concepts capture the quantifier-free formulas of FO^2 with exactly one free variable. Using the translation of DL syntax into FO syntax (see Definition 2.6) the atomic $\mathcal{DL}_{\mathsf{QF}}$-concepts of the form $\exists P.\{o\}, \exists \mathsf{inv}(P).\{o\}$ and $\exists(\mathsf{id}\sqcap P).\top$ translate to the following FO^2-atoms:

$$\begin{aligned}
\mathsf{tr}^x(\exists P.\{o\}) \quad\quad &= \exists y.(P(x,y) \wedge y \approx o) \equiv_{\mathrm{FO}} P(x,o),\\
\mathsf{tr}^x(\exists \mathsf{inv}(P).\{o\}) &= \exists y.(P(y,x) \wedge y \approx o) \equiv_{\mathrm{FO}} P(o,x),\\
\mathsf{tr}^x(\exists(\mathsf{id} \sqcap P).\top) &= \exists y.(x \approx y \wedge P(x,y) \wedge x \approx x) \equiv_{\mathrm{FO}} P(x,x).
\end{aligned}$$

Furthermore, $\top$ translates to $x \approx x$, A to $A(x)$ and $\{o\}$ to $x \approx o$. $\mathcal{DL}_{\mathsf{QF}}$-concepts of the form $\forall U.(\neg\{o\} \sqcup C)$ are used to express ground atoms. For an interpretation $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ it holds that

$$\begin{aligned}
(\forall U.(\neg\{o\} \sqcup C))^{\mathcal{I}} &= \Delta_{\mathcal{I}} \text{ iff } \mathcal{I} \models o \sqsubseteq C, \text{ and}\\
(\forall U.(\neg\{o\} \sqcup C))^{\mathcal{I}} &= \emptyset \quad \text{ iff } \mathcal{I} \not\models o \sqsubseteq C.
\end{aligned}$$

For example, the FO^2-formula $A(x) \wedge P(o,o')$, where o and o' are object names, is equivalent to the concept

$$A \sqcap \forall U.(\neg\{o\} \sqcup \exists P.\{o'\}).$$

Note that the role assertion $(o,o') \sqsubseteq P$ is equivalent to the concept assertion $o \sqsubseteq \exists P.\{o'\}$.

Atomic $\mathcal{DL}_{\mathsf{QF}}$-roles id, P, and $\mathsf{inv}(P)$ correspond to FO^2-atoms with exactly two free variables of the form $x \approx y$, $P(x,y)$ and $P(y,x)$, respectively. For example, the FO^2 formula $A(x) \wedge A(y)$ is equivalent to the role $(A \times \top) \sqcap (\top \times A)$ and the formula

$$A(x) \vee P_1(y,x) \vee \neg(P_1(x,o) \wedge P_2(o',y))$$

is equivalent to the $\mathcal{DL}_{\mathsf{QF}}$-role

$$(A \times \top) \sqcup \mathsf{inv}(P_1) \sqcup \neg\Big(\big((\exists P_1.\{o\}) \times \top\big) \sqcap \big(\top \times (\exists \mathsf{inv}(P_2).\{o'\})\big)\Big).$$

We define a *conjunctive normal form* (CNF) for $\mathcal{DL}_{\mathsf{QF}}$-concepts and $\mathcal{DL}_{\mathsf{QF}}$-roles.

Definition 5.30. A $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role L is called a *literal* iff L is atomic or the negation of an atomic $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role, respectively. A $\mathcal{DL}_{\mathsf{QF}}$-*clause* is a disjunction of the form $L_1 \sqcup \cdots \sqcup L_n$, for some $n \geq 0$, where the L_is are literals.

We say that *a $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role is in CNF* iff it is of the form $C_1 \sqcap \cdots \sqcap C_m$, for some $m \geq 0$, where the C_js are $\mathcal{DL}_{\mathsf{QF}}$-clauses. ▲

To transform a $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role into a Boolean combination of atomic concepts or roles, respectively, the following equivalences can be used as rewrite rules (applied from left to right):

$$\begin{aligned}
\forall U.(\neg\{o\} \sqcup (C \sqcap D)) &\equiv \forall U.(\neg\{o\} \sqcup C) \sqcap \forall U.(\neg\{o\} \sqcup D);\\
\forall U.(\neg\{o\} \sqcup (\neg C)) &\equiv \neg\forall U.(\neg\{o\} \sqcup C);\\
\forall U.(\neg\{o\} \sqcup (\forall U.(\neg\{o'\} \sqcup C))) &\equiv \forall U.(\neg\{o'\} \sqcup C).
\end{aligned} \tag{5.21}$$

Similar equivalences hold for concept products:

$$\begin{aligned}\top \times (C \sqcap D) &\equiv (\top \times C) \sqcap (\top \times D);\\ \top \times (\neg C) &\equiv \neg(\top \times C);\\ (C \sqcap D) \times \top &\equiv (C \times \top) \sqcap (D \times \top);\\ (\neg C) \times \top &\equiv \neg(C \times \top).\end{aligned} \tag{5.22}$$

Note that C and D stand for arbitrary complex concepts and o and o' for object names. Every $\mathcal{DL}_{\mathsf{QF}}$-concept and $\mathcal{DL}_{\mathsf{QF}}$-role can be transformed into CNF using the equivalences above and the standard CNF transformation for propositional logic. Let X be a $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role, the corresponding expression in CNF is denoted by $\mathsf{cnf}(X)$.

To define the class of *flat* $\mathcal{DL}$-admissible representations we require that the corresponding computable functions $\mathsf{E}^+[\cdot]$ and $\mathsf{E}^-[\cdot]$ provide an output formulated in $\mathcal{DL}_{\mathsf{QF}}$.

Definition 5.31. Let $\boldsymbol{A}$ be a finite set of ground action terms and $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a $\mathcal{DL}$-admissible representation. We call $\Sigma_{\boldsymbol{A}}$ *flat* iff for all $(\mathfrak{s}, \boldsymbol{\alpha}, F) \in \mathfrak{S}_{\mathcal{C}} \times \boldsymbol{A} \times \mathcal{F}$ the concepts (or roles) $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$ and $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$ are formulated in $\mathcal{DL}_{\mathsf{QF}}$. ▲

For example, the action theories in Example 5.27 and 5.28 are flat $\mathcal{DL}$-admissible representations.

Note that the context $\mathcal{C}$ can be an arbitrary $\mathcal{DL}$-context. Let $F \in \mathsf{N_C} \cup \mathsf{N_R}$ be a concept name or role name, an unconditional effect description of the form

$$\langle F, X \rangle^{\pm}$$

is called $\mathcal{DL}_{\mathsf{QF}}$-*effect* iff X is a $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role, respectively.

Given a flat $\mathcal{DL}$-admissible representation $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ for a finite set of actions $\boldsymbol{A}$, the effects of executing a ground action $\boldsymbol{\alpha} \in \boldsymbol{A}$ in an interpretation of type $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ can be described as a set of $\mathcal{DL}_{\mathsf{QF}}$-effects denoted by $\Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$ (see Definition 5.4). Next, we will show that also the effects of action sequences in a flat representation can be described as sets of $\mathcal{DL}_{\mathsf{QF}}$-effects. The accumulation of effect sets according to 5.8 involves regression. Thus, it suffices to show that for a given $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role X and a set of $\mathcal{DL}_{\mathsf{QF}}$-effects E, the regression result $\mathfrak{R}[X, \mathsf{E}]$ can be transformed into $\mathcal{DL}_{\mathsf{QF}}$. The regression results may contain complex inverse roles. They can be removed by applying the following equivalences, where R and S stand for possibly complex $\mathcal{DL}$-roles, from left to right:

$$\begin{aligned}\mathsf{inv}(\mathsf{inv}(R)) &\equiv R;\\ \mathsf{inv}(\mathsf{id}) &\equiv \mathsf{id};\\ \mathsf{inv}(R \sqcap S) &\equiv \mathsf{inv}(R) \sqcap \mathsf{inv}(S);\\ \mathsf{inv}(\neg R) &\equiv \neg \mathsf{inv}(R);\\ \mathsf{inv}(C \times D) &\equiv D \times C.\end{aligned} \tag{5.23}$$

The regression of atomic $\mathcal{DL}_{\mathsf{QF}}$-concepts of the form $\exists P.\{o\}$ and $\exists \mathsf{inv}(P).\{o\}$ through a set of $\mathcal{DL}_{\mathsf{QF}}$-effects yields $\mathcal{DL}$-concepts of the form $\exists R.\{o\}$ and $\exists(\mathsf{id} \sqcap R).\top$, where R is a possibly complex $\mathcal{DL}_{\mathsf{QF}}$-role. To deal with such concepts we exploit the following equivalences

$$\begin{aligned}\exists(R \sqcap S).\{o\} &\equiv \exists R.\{o\} \sqcap \exists S.\{o\};\\ \exists(\neg R).\{o\} &\equiv \neg\exists R.\{o\};\\ \exists\mathsf{id}.\{o\} &\equiv \{o\};\\ \exists(\top \times C).\{o\} &\equiv \forall U.(\neg\{o\} \sqcup C);\\ \exists(C \times \top).\{o\} &\equiv C.\end{aligned} \tag{5.24}$$

Next, we consider concepts of the form $\exists(\mathsf{id} \sqcap R).\top$, where R is a (possibly complex) $\mathcal{DL}_{\mathsf{QF}}$-role. We have

$$\begin{aligned}\exists(\mathsf{id} \sqcap (R \sqcap S)).\top &\equiv \exists(\mathsf{id} \sqcap R).\top \sqcap \exists(\mathsf{id} \sqcap S).\top;\\ \exists(\mathsf{id} \sqcap \neg R).\top &\equiv \neg\exists(\mathsf{id} \sqcap R).\top;\\ \exists(\mathsf{id} \sqcap \mathsf{inv}(R)).\top &\equiv \exists(\mathsf{id} \sqcap R).\top;\\ \exists(\mathsf{id} \sqcap \mathsf{id}).\top &\equiv \top;\\ \exists(\mathsf{id} \sqcap (\top \times C)).\top &\equiv C;\\ \exists(\mathsf{id} \sqcap (C \times \top)).\top &\equiv C.\end{aligned} \tag{5.25}$$

Lemma 5.32. *Let X be a $\mathcal{DL}_{\mathsf{QF}}$-concept or $\mathcal{DL}_{\mathsf{QF}}$-role and E a set of $\mathcal{DL}_{\mathsf{QF}}$-effects. The regression $\mathfrak{R}[X, \mathsf{E}]$ of X through E can be equivalently transformed into $\mathcal{DL}_{\mathsf{QF}}$.*

Proof. We have that $\mathfrak{R}[X, \mathsf{E}] \equiv \mathfrak{R}[\mathsf{cnf}(X), \mathsf{E}]$. It suffices to prove the claim for the case where X is atomic, because the regression operator can be pushed inside until it only appears in front of atoms. In case X is of the form $\top$, $\{o\}$, A, id or P for a concept name A, object name o and role name P the claim is immediately satisfied by assumption on E and by definition of regression (see Figure 5.2). The regression of atoms of the form $\exists P.\{o\}$, $\exists\mathsf{inv}(P).\{o\}$, $\exists(\mathsf{id} \sqcap P).\top$, $\mathsf{inv}(P)$, $\top \times B$ and $B \times \top$, where B is atomic, can be transformed into corresponding $\mathcal{DL}_{\mathsf{QF}}$ expressions using the equivalences (5.23), (5.24) and (5.25) as rewrite rules from left to right. $\square$

It now follows that sets of unconditional $\mathcal{DL}_{\mathsf{QF}}$-effects can be accumulated.

Lemma 5.33. *Let E_0 and E_1 be sets of unconditional $\mathcal{DL}_{\mathsf{QF}}$-effects. It holds that the effects in $\mathsf{E}_0 \bowtie \mathsf{E}_1$ can be equivalently transformed to $\mathcal{DL}_{\mathsf{QF}}$-effects.*

Proof. According to Definition 5.8 we have

$$\mathsf{E}_0 \bowtie \mathsf{E}_1 := \mathsf{Regr}(\mathsf{E}_1, \mathsf{E}_0) \cup \mathsf{Diff}^+(\mathsf{E}_0, \mathsf{E}_1) \cup \{\langle F, X\rangle^- \in \mathsf{E}_0\}.$$

Lemma 5.32 and the assumption on E_0 and E_1 imply that the effects in $\mathsf{Regr}(\mathsf{E}_1, \mathsf{E}_0)$ and $\mathsf{Diff}^+(\mathsf{E}_0, \mathsf{E}_1)$ can be formulated in $\mathcal{DL}_{\mathsf{QF}}$. The same is true for $\{\langle F, X\rangle^- \in \mathsf{E}_0\}$ by assumption on E_0. $\square$

Let $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ be a flat $\mathcal{DL}$-admissible representation for a finite set of actions $\boldsymbol{A}$. There are finitely many concept names, role names and object names mentioned in the effect definitions $\mathsf{E}^+[\mathfrak{s}, \boldsymbol{\alpha}, F]$, $\mathsf{E}^-[\mathfrak{s}, \boldsymbol{\alpha}, F]$, where $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$. Thus, there are finitely many $\mathcal{DL}_{\mathsf{QF}}$-concept literals and finitely many $\mathcal{DL}_{\mathsf{QF}}$-role literals that can be built using these names. With $\mathsf{L}_\mathsf{C}(\Sigma_{\boldsymbol{A}})$ we denote the *set of all $\mathcal{DL}_{\mathsf{QF}}$-concept literals* over the names relevant in $\Sigma_{\boldsymbol{A}}$ and with $\mathsf{L}_\mathsf{R}(\Sigma_{\boldsymbol{A}})$ the corresponding *set of all $\mathcal{DL}_{\mathsf{QF}}$-role literals*. There are infinitely many syntactically different $\mathcal{DL}_{\mathsf{QF}}$-concepts and $\mathcal{DL}_{\mathsf{QF}}$-roles that can be built using a finite set of literals, but there are only finitely many equivalence classes w.r.t. "$\equiv$". To obtain a representative for each equivalence class we consider the CNF of a $\mathcal{DL}_{\mathsf{QF}}$-concept or role and view it as a set of sets of literals.

Definition 5.34. Let $\boldsymbol{A}$ be a finite set of ground action terms, $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a flat $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ and $\mathsf{L}_\mathsf{C}(\Sigma_{\boldsymbol{A}})$ and $\mathsf{L}_\mathsf{R}(\Sigma_{\boldsymbol{A}})$ the sets of literals as defined above. The *set of all relevant effects*, denoted by $\mathfrak{E}$, is defined as follows:

$$\mathfrak{E} := \{\langle A, C\rangle^+, \langle A, C\rangle^- \mid A \in \mathcal{F} \cap \mathsf{N}_\mathsf{C}, C \subseteq 2^{\mathsf{L}_\mathsf{C}(\Sigma_{\boldsymbol{A}})}\} \cup$$
$$\{\langle P, R\rangle^+, \langle P, R\rangle^- \mid P \in \mathcal{F} \cap \mathsf{N}_\mathsf{R}, R \subseteq 2^{\mathsf{L}_\mathsf{R}(\Sigma_{\boldsymbol{A}})}\}.$$

▲

Note that a set $C \subseteq 2^{\mathsf{L}_\mathsf{C}(\Sigma_{\boldsymbol{A}})}$ stands for the $\mathcal{DL}_{\mathsf{QF}}$-concept

$$C := \bigsqcap_{\mathcal{D} \in C} \Big(\bigsqcup_{L \in \mathcal{D}} L \Big)$$

in CNF, and analogous for roles.

Since we have already shown that the accumulation of $\mathcal{DL}_{\mathsf{QF}}$-effect sets can be expressed within $\mathcal{DL}_{\mathsf{QF}}$, it is now straightforward to prove that the set $\mathfrak{E}$ is closed under accumulations w.r.t. $\Sigma_{\boldsymbol{A}}$.

Lemma 5.35. *Let $\boldsymbol{A}$ be a finite set of ground actions, $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a flat $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$. The set of all relevant effects $\mathfrak{E}$ as defined above is closed under accumulations w.r.t. $\Sigma_{\boldsymbol{A}}$.*

Proof. Let $\mathsf{E} \subseteq \mathfrak{E}$, $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}}$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$. By assumption E and $\Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha})$ are sets of $\mathcal{DL}_{\mathsf{QF}}$-effects over names mentioned in in the effect definitions provided by $\mathsf{E}^+[\cdot]$ and $\mathsf{E}^-[\cdot]$. Let $\langle F, X\rangle^{\pm} \in (\mathsf{E} \bowtie \Sigma_{\boldsymbol{A}}(\mathfrak{s}, \boldsymbol{\alpha}))$. According to Lemma 5.32 and 5.33 X can be transformed into $\mathcal{DL}_{\mathsf{QF}}$ without introducing new names. Let $\mathsf{cnf}(X)$ be the CNF of X viewed as a set of sets of literals. It follows that $\mathsf{cnf}(X) \subseteq 2^{\mathsf{L}_\mathsf{C}(\Sigma_{\boldsymbol{A}})}$, if F is a concept name, and $\mathsf{cnf}(X) \subseteq 2^{\mathsf{L}_\mathsf{R}(\Sigma_{\boldsymbol{A}})}$, if F is a role name. By definition of $\mathfrak{E}$ we have $\langle F, \mathsf{cnf}(X)\rangle^{\pm} \in \mathfrak{E}$ and $X \equiv \mathsf{cnf}(X)$. □

Since $\mathfrak{E}$ is finite we obtain finitely many dynamic types.

Lemma 5.36. *Let $\boldsymbol{A}$ be a finite set of ground action terms and $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$ a flat $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ and let $\mathfrak{E}$ be the set of all relevant effects as defined above. The set of all dynamic types w.r.t. $\Sigma_{\boldsymbol{A}}$ and $\mathfrak{E}$ is effectively computable.*

Proof. The proof is analogous to the proof of Lemma 5.25. □

Analogous to the verification problem for $\mathcal{DL}$-ConGolog based on an acyclic action representations the input consists of the following components:

- a finite set $\boldsymbol{A}$ of ground action terms;

- a flat $\mathcal{DL}$-admissible representation of $\boldsymbol{A}$ of the form $\Sigma_{\boldsymbol{A}} = (\mathcal{K}, \mathcal{F}, \mathcal{C}, \mathsf{E}^+, \mathsf{E}^-, \mathsf{Pre}_{\mathcal{C}})$;
- a pick-free program expression δ, where all action terms are from $\boldsymbol{A}$ and for all tests of the form ψ? in δ we have $\psi \in \mathcal{C}$;
- a $\mathcal{DL}$-CTL* state formula Φ over axioms from $\mathcal{C}$.

We want to check whether Φ is valid in $\mathcal{P} = (\mathfrak{D}(\Sigma_{\boldsymbol{A}}), \delta)$. The construction of a finite $\mathcal{C}$-bisimilar abstraction of $\mathcal{P}$ works in the same way as for the acyclic case. This leads to the following decidability result.

Theorem 5.37. *Verifying $\mathcal{DL}$-CTL* properties of $\mathcal{DL}$-ConGolog programs over ground actions with a flat $\mathcal{DL}$-admissible representation is decidable.*

The result generalizes the decidability result for programs over local effect actions (Theorem 4.18). Note that a $\mathcal{DL}$-admissible local effect representation can be equivalently described as a flat (and acyclic) $\mathcal{DL}$-admissible representation.

5.4 Summary and Related Work

Summary

In this chapter, we have shown undecidability and decidability results for verifying DL-CTL* properties of DL-ConGolog programs over non-local effect actions. Non-local action effects easily lead to undecidability even for rather inexpressive DLs (Corollary 5.3). However, we have identified two expressive (incomparable) fragments of DL-ConGolog programs for which verification is decidable. One decidable class is obtained by disallowing cyclic dependencies between fluents (Theorem 5.26) and the other one by resorting to so-called *flat* (quantifier-free) effect definitions (Theorem 5.37). Both classes are considerably more expressive than the one based only on local effects, and involve actions that possibly affect an unbounded number of unnamed objects. Decidability is obtained by a reduction to propositional model checking using a generalization of the abstraction technique introduced in the previous chapter.

Related Work

Another decidability result for a verification task in a fragment of the Situation Calculus has been obtained in [Ter99]. In this fragment only propositional fluents are available but the domain of all actions can be infinite. Branching-time temporal properties of the infinite tree of all executable situations are considered. Decidability is shown by reducing the verification problem to a decidable emptiness problem of an appropriate tree automata model.

Recently, the class of *Bounded Situation Calculus Action Theories* [DLP16] has been introduced as an approach to achieve decidability of the verification problem for first-order μ-calculus properties. The class is based on a semantical restriction requiring that the extension of each fluent in each situation is bounded by some constant. Nevertheless, an infinite object domain and also an infinite action domain is present whereas in our case only a finite domain of actions is considered. However, due to the boundedness assumption, for instance, DL KBs cannot be integrated, because DL KBs are usually interpreted without such an assumption. A decidability result for verifying ConGolog programs over bounded action theories has been obtained in [De +16b].

In a DL-based setting, (un)decidability of verification of so-called *Description Logic Knowledge and Action Bases* has been studied in [Har+13]. The action formalism is based on a different notion of change compared to ours. The states in this formalism are finite ABoxes that can be extended with new individuals during an execution. This leads to a different source of infiniteness of the underlying transition system.

Chapter 6

Decidable Reasoning about Actions with Knowledge and Sensing

We consider dynamical systems where in addition to the current state of the world also the subjective incomplete view of a single agent on this world is modeled. This allows us to distinguish between the world-changing and the knowledge-changing effects of an action. Clearly, the decision of an agent with incomplete information about its surroundings on what to do next depends on what is known or not known by the agent at the time the decision has to be made.

Example 6.1. As a running example in this chapter we consider an agent that takes care of a possibly faulty electrical device. Assume the (static) knowledge base $\mathcal{K}$ of the agent includes facts about the current status of the device but also information about possible faults provided in the manual of the device. The agent has sensors to scan the display of the device in order to receive error messages reporting about faults. In addition, the agent also has actuators at its disposal to repair certain faults. Let's say $\mathcal{K}$ contains the assertion

$$\mathsf{dev} \sqsubseteq \exists \mathit{HasFault}.\top$$

about the faulty device named dev. We assume that faults have identifiers that are objects in the domain and *HasFault* is a role name relating devices to their faults. The axiom says that dev has *some* fault but not *which* one it is. Since $\mathcal{K}$ only provides incomplete information, it might be the case that the identity of the faults of dev are not given in $\mathcal{K}$ and need to be figured out by the knowledge-based agent using its sensors. ▲

To handle such domains we define an epistemic DL-based action formalism and extend our DL-action theories with a simple notion of sensing actions that describe the ability of an agent to gain new information from the environment. For the sake of simplicity, we use the basic prototypical DL $\mathcal{ALCO}$.

A question that arises in the example is whether the agent is able to identify and repair the faults of dev given the knowledge provided in $\mathcal{K}$ and the sensing actions. It might also happen that dev has *unknown* faults, i.e. faults that are not named in $\mathcal{K}$. A basic reasoning problem that is relevant for answering this question is the *epistemic projection problem*. The projection problem asks whether a given formula (called *projection query*) is true after executing a sequence of actions. In our setting, the projection query is formulated in the epistemic DL $\mathcal{ALCOK}$ [Don+98] that extends $\mathcal{ALCO}$ with a knowledge operator $\mathbf{K}$. For instance, in our example the $\mathbf{K}$ is useful to explicitly talk about known and unknown faults in the projection query. To decide the epistemic projection problem we reduce it to a standard DL reasoning task by combining known methods for answering static epistemic queries [Don+98; MRG11] with the reduction approach developed for the non-epistemic projection problem [Baa+05a].

The chapter is organized as follows. In Section 6.1, we formally define epistemic first-order dynamical systems (epistemic FO-DS for short) as a general meta-level structure for describing the meaning of world-changing and knowledge-changing actions. In Section 6.2, we define the epistemic projection problem for projection queries formulated in the epistemic DL $\mathcal{ALCOK}$ and actions represented in a DL-action theory with sensing actions. In Section 6.3, we show that our semantics of action theories is theoretically well-founded on existing epistemic extensions of the Situation Calculus [SL03; LL04; LL11]. We provide an embedding into the modal variant of the epistemic Situation Calculus $\mathcal{ES}$ [LL04; LL11]. In Section 6.4, we investigate the complexity of the epistemic projection problem. A brief summary of the chapter is provided in Section 6.5.

6.1 Epistemic First-Order Dynamical Systems

An epistemic first-order dynamical system is a meta-level structure for describing the meaning of a set of ground action terms. It is an epistemic extension of an FO-DS. A *state* in this system is an epistemic interpretation.

Definition 6.2. An *epistemic interpretation* is a pair of the form $(\mathcal{I}, \mathcal{W})$, where $\mathcal{W}$ is a non-empty set of first-oder interpretations that satisfy the SNA and $\mathcal{I}$ is an element of $\mathcal{W}$. ▲

We consider the same signature as for ordinary FO-DSs (see Definition 2.1). All interpretations have a fixed common countably infinite domain given by the set of all object names $\mathsf{N_O}$ (see Definition 2.4).

As before, the interpretation $\mathcal{I}$ in $(\mathcal{I}, \mathcal{W})$ completely describes the current state of the world. $\mathcal{W}$ is a set of possible worlds represents the *epistemic state* (or *knowledge state*) of a single agent. With requiring $\mathcal{I} \in \mathcal{W}$ we assume that the real world $\mathcal{I}$ is always considered possible.

Definition 6.3. An *epistemic first-oder dynamical system* (epistemic FO-DS) is a tuple

$$\mathfrak{D}_{\mathsf{K}} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}, \sim_{\mathsf{s}})$$

that extends an ordinary FO-DS $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$ (considered under the SNA) with a *sensing compatibility relation*

$$\sim_{\mathsf{s}} \subseteq \mathbb{I} \times \mathsf{ground(Act)} \times \mathbb{I}.$$

Instead of $(\mathcal{I}, \boldsymbol{\alpha}, \mathcal{J}) \in \sim_{\mathsf{s}}$ we choose to write $\mathcal{I} \sim_{\mathsf{s}}^{\boldsymbol{\alpha}} \mathcal{J}$. The *state space* of $\mathfrak{D}_{\mathsf{K}}$, denoted by $\mathbb{I}_{\mathsf{K}}$, is the following set of epistemic interpretations:

$$\mathbb{I}_{\mathsf{K}} := \{(\mathcal{I}, \mathcal{W}) \mid \mathcal{W} \subseteq \mathbb{I}, \mathcal{I} \in \mathcal{W}\}.$$

We sometimes also write $\mathfrak{D}_{\mathsf{K}} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}, \sim_{\mathsf{s}})$ as a pair of the form

$$\mathfrak{D}_{\mathsf{K}} = (\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}), \sim_{\mathsf{s}}).$$

▲

In the following

$\mathbb{I}$ always denotes the set of all interpretations satisfying the SNA.

and $\mathbb{I}_\mathsf{K}$ the set of all epistemic interpretations. Thus, all epistemic FO-DSs have the same state space. We therefore often omit $\mathbb{I}$ as the first element of the tuple when writing an epistemic FO-DS. Consider an FO sentence ϕ and a state $(\mathcal{I}, \mathcal{W}) \in \mathbb{I}_\mathsf{K}$. We say that ϕ is *known* in $(\mathcal{I}, \mathcal{W})$ iff ϕ is true in all interpretations $\mathcal{J} \in \mathcal{W}$. For example, if $\mathcal{W} = \{\mathcal{I}\}$, then the agent has complete knowledge and knows everything about the current state of the world. In the opposite extreme case we have

$$\mathcal{W} = \{\mathcal{J} = (\Delta_\mathcal{J}, \cdot^\mathcal{J}) \mid \Delta_\mathcal{J} = \mathsf{N_O} \text{ and } o^\mathcal{J} = o \text{ for all } o \in \mathsf{N_O}\}.$$

It means that *all* interpretations (satisfying the SNA) are considered possible. Consequently, the agent does not know anything about $\mathcal{I}$ (note that the identity of all object names is known because they are interpreted in the same way in all possible worlds according to the SNA). Intuitively, the less worlds are contained in $\mathcal{W}$ the more the agent knows about $\mathcal{I}$. The relation $\mathcal{I} \sim_\mathsf{s}^{\boldsymbol{\alpha}} \mathcal{J}$ for a ground action $\boldsymbol{\alpha}$ means that the worlds $\mathcal{I}$ and $\mathcal{J}$ agree on the sensing result provided by $\boldsymbol{\alpha}$. The transition relation induced by $\mathcal{E}$ and $\sim_\mathsf{s}$ is defined as follows.

Definition 6.4. Let $\mathfrak{D}_\mathsf{K} = (\mathbb{I}_\mathsf{ini}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}, \sim_\mathsf{s})$ be an epistemic FO-DS that extends the FO-DS $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_\mathsf{ini}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$. Furthermore, let $\boldsymbol{\alpha} \in \mathsf{ground(Act)}$ be a ground action and $(\mathcal{I}, \mathcal{W}), (\mathcal{I}', \mathcal{W}') \in \mathbb{I}_\mathsf{K}$ two epistemic interpretations. We say that $\boldsymbol{\alpha}$ *transforms* $(\mathcal{I}, \mathcal{W})$ *into* $(\mathcal{I}', \mathcal{W}')$, written as

$$(\mathcal{I}, \mathcal{W}) \Longrightarrow_{\mathfrak{D}_\mathsf{K}}^{\boldsymbol{\alpha}} (\mathcal{I}', \mathcal{W}'),$$

iff the following conditions are satisfied

- $\mathcal{I} \Rightarrow_\mathfrak{D}^{\boldsymbol{\alpha}} \mathcal{I}'$;
- $\mathcal{W}' = \{\mathcal{J}' \mid \mathcal{J} \Rightarrow_\mathfrak{D}^{\boldsymbol{\alpha}} \mathcal{J}' \text{ for some } \mathcal{J} \in \mathcal{W} \text{ with } \mathcal{I} \sim_\mathsf{s}^{\boldsymbol{\alpha}} \mathcal{J}\}$.

Let $\sigma = \boldsymbol{\alpha}_0\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \mathsf{ground(Act^*)}$ for some $n \geq 0$ be a sequence of ground actions and $(\mathcal{I}, \mathcal{W})$ and $(\mathcal{J}, \mathcal{V})$ two epistemic interpretations. We write $(\mathcal{I}, \mathcal{W}) \Longrightarrow_{\mathfrak{D}_\mathsf{K}}^{\sigma} (\mathcal{J}, \mathcal{V})$ iff there exists a sequence of epistemic interpretations $(\mathcal{I}_0, \mathcal{W}_0), \ldots, (\mathcal{I}_{n+1}, \mathcal{W}_{n+1})$ such that $(\mathcal{I}_i, \mathcal{W}_i) \Longrightarrow_{\mathfrak{D}_\mathsf{K}}^{\boldsymbol{\alpha}_i}$ $(\mathcal{I}_{i+1}, \mathcal{W}_{i+1})$ for all $i = 0, \ldots, n+1$ and $(\mathcal{I}, \mathcal{W}) = (\mathcal{I}_0, \mathcal{W}_0)$ and $(\mathcal{J}, \mathcal{V}) = (\mathcal{I}_{n+1}, \mathcal{W}_{n+1})$. ▲

Note that "$\Rightarrow_\mathfrak{D}^{\boldsymbol{\alpha}}$" for a ground action $\boldsymbol{\alpha}$ is the ordinary transition relation of an FO-DS determined by $\mathcal{E}$ (see Definition 2.8). The effect function $\mathcal{E}$ captures the physical effects of an action. The real world $\mathcal{I}$ in an epistemic interpretation $(\mathcal{I}, \mathcal{W})$ is updated according to $\mathcal{E}$. Those interpretations $\mathcal{J} \in \mathcal{W}$ that are sensing compatible with $\mathcal{I}$ are updated with $\mathcal{E}$ as well and yield the new epistemic state whereas those interpretations that do not agree with $\mathcal{I}$ on the sensing result of $\boldsymbol{\alpha}$ are discarded. In our semantics the real world itself and all possible worlds that are sensing compatible with the real world are updated using the same pair $\mathcal{E}$ of effect functions. Thus, the agent is fully aware of all effects of an action and perceives the occurrences of all actions.

Example 6.5. Consider an FO sentence ϕ over some finite set of fluents $\mathcal{F}$ and some epistemic FO-DS $\mathfrak{D}_\mathsf{K} = (\mathbb{I}_\mathsf{ini}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}, \sim_\mathsf{s})$. In $\mathfrak{D}_\mathsf{K}$ we define a ground action $\boldsymbol{\alpha}_\phi \in \mathsf{Act}$ as a purely sensing action that stands for the ability of the agent to check the truth of ϕ. There are no world changing effects:

$$\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}_\phi, F) = \mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}_\phi, F) = \emptyset \text{ for all } \mathcal{I} \in \mathbb{I} \text{ and all } F \in \mathcal{F},$$

with $\mathfrak{D} = (\mathbb{I}, \mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre})$. The sensing compatibility relation for $\boldsymbol{\alpha}_\phi$ is defined by

$$\mathcal{I} \sim_{\mathsf{s}}^{\boldsymbol{\alpha}_\phi} \mathcal{J} \text{ iff } (\mathcal{I} \models \phi \Leftrightarrow \mathcal{J} \models \phi)$$

for all $\mathcal{I}, \mathcal{J} \in \mathbb{I}$. Consider two epistemic states $(\mathcal{I}, \mathcal{W}), (\mathcal{J}, \mathcal{W}) \in \mathbb{I}_{\mathbf{K}}$ with $\mathcal{I} \models \phi$ and $\mathcal{J} \models \neg\phi$. In case $\boldsymbol{\alpha}_\phi$ is executed in $(\mathcal{I}, \mathcal{W})$ the agent observes that ϕ is true and discards all interpretations where ϕ is false:

$$(\mathcal{I}, \mathcal{W}) \Longrightarrow_{\mathfrak{D}_{\mathbf{K}}}^{\boldsymbol{\alpha}_\phi} (\mathcal{I}, \mathcal{W}') \text{ with } \mathcal{W}' = \{\mathcal{Y} \in \mathcal{W} \mid \mathcal{I} \sim_{\mathsf{s}}^{\boldsymbol{\alpha}_\phi} \mathcal{Y}\}.$$

Since ϕ is true in $\mathcal{I}$, the resulting epistemic state $\mathcal{W}'$ consists of those interpretations from $\mathcal{W}$ where ϕ is true as well. In case of $(\mathcal{J}, \mathcal{W})$ it is the other way round. We obtain the epistemic state with all interpretations from $\mathcal{W}$ where ϕ is false. Thus, by executing $\boldsymbol{\alpha}_\phi$ the agent gets to know whether ϕ is true and nothing else is changed. ▲

6.2 An Agent Language with Sensing

In this section we extend DL-action theories with a simple notion of sensing for describing the basic abilities of an agent to gain new information from the environment. Equipped with an action theory the agent should be able to check whether a chosen action sequence leads to the expected result before actually executing it. To formulate subjective projection queries that refer to what the agent knows or does not know we choose a DL extended with a knowledge modality. In the next subsection we recall basic notions of the epistemic DL $\mathcal{ALCOK}$ [Don+98]. Afterwards, action theories and the *epistemic projection problem* as a basic reasoning task are defined.

6.2.1 An Epistemic DL

Epistemic extensions of DLs have been well studied (see for instance [Don+98; Cal+07b; uR11; Meh14]). In most of these works the main motivation of enriching a DL with an epistemic operator was to obtain a more expressive query language that allows for knowledge base introspection. Donini et al. [Don+98] introduced an epistemic DL called $\mathcal{ALCK}$. It is an extension of $\mathcal{ALC}$ with a knowledge operator $\mathbf{K}$ for concepts and roles that is interpreted using a possible world semantics under the SNA. The resulting logic was mainly considered as an epistemic language for querying standard (objective) $\mathcal{ALC}$ knowledge bases. Mehdi and Rudolph [uR11; Meh14] studied the problem of answering epistemic subsumption and instance queries in presence of the more expressive DLs $\mathcal{SRIQ}$ and $\mathcal{SROIQ}$.

Following [Don+98] we define $\mathcal{ALCOK}$, i.e. $\mathcal{ALCK}$ with nominals. The knowledge modality will be later useful for formulating subjective projection queries.

Definition 6.6. *$\mathcal{ALCOK}$-concepts C* and *$\mathcal{ALCOK}$-roles R* are built according to the following syntax rules:

$$C ::= \top \mid \bot \mid A \mid \{o\} \mid \neg C \mid C \sqcup C \mid C \sqcap C \mid \exists R.C \mid \forall R.C \mid \mathbf{K}C;$$

$$R ::= P \mid \mathbf{K}R,$$

where A ranges over $\mathsf{N_C}$, o over $\mathsf{N_O}$ and P over $\mathsf{N_R}$. Note that every $\mathcal{ALCO}$-concept/role (without a $\mathbf{K}$) is also an $\mathcal{ALCOK}$-concept/role. $\mathcal{ALCK}$-concepts are $\mathcal{ALCOK}$-concepts without nominals. ▲

Intuitively, the epistemic concept $\mathbf{K}C$ describes the set of all those objects that belong to C in all possible worlds. And likewise, the role $\mathbf{K}R$ captures all pairs of objects that are R-related in all possible worlds. We use the concept constructor $\mathbf{Kw}C$ (*known whether*) as an abbreviation of the concept $\mathbf{K}C \sqcup \mathbf{K}\neg C$. Intuitively, an object belongs to $\mathbf{Kw}C$ if all possible worlds agree on the membership of this object to C.

The knowledge operator is also allowed in front of axioms.

Definition 6.7. A *Boolean $\mathcal{ALCOK}$-KB ψ* is built according to the following syntax rule

$$\psi ::= t \mathrel{\varepsilon} C \mid (t,t') \mathrel{\varepsilon} R \mid C \sqsubseteq D \mid t \approx t' \mid \neg\psi \mid \psi \wedge \psi \mid \mathbf{K}\psi,$$

where $t,t' \in \mathsf{N_O} \cup \mathsf{N_V}$ are object terms, C and D stand for $\mathcal{ALCOK}$-concepts and R for an $\mathcal{ALCOK}$-role. The expressions $t \mathrel{\varepsilon} C$, $(t,t') \mathrel{\varepsilon} R$ and $t \approx t'$ are called $\mathcal{ALCOK}$-ABox assertions and $C \sqsubseteq D$ an $\mathcal{ALCOK}$-concept inclusion. A Boolean $\mathcal{ALCOK}$-KB is called *Boolean $\mathcal{ALCOK}$-ABox* if it is a Boolean combination of ABox assertions.

A Boolean $\mathcal{ALCOK}$-KB is called *ground* if no variable names are mentioned. Let $v : \mathsf{N_V} \cup \mathsf{N_O} \to \mathsf{N_O}$ be a variable mapping and ψ a Boolean $\mathcal{ALCOK}$-KB. The *grounding* of ψ with v, denoted by ψ^v, is a ground Boolean $\mathcal{ALCOK}$-KB obtained from ψ by replacing all occurrences of all variable names x in ψ with $v(x)$. ▲

As for concepts we use

$$\mathbf{Kw}\psi \text{ ("knowing whether } \psi \text{ is true")}$$

as an abbreviation of $\mathbf{K}\psi \vee \mathbf{K}\neg\psi$.

An $\mathcal{ALCOK}$-concept, $\mathcal{ALCOK}$-role or Boolean $\mathcal{ALCOK}$-KB is called *subjective* if all occurring concept names and role names occur within the scope of a $\mathbf{K}$. It is called *objective* if no $\mathbf{K}$ occurs at all. Thus, the objective concepts, roles and Boolean KBs are the non-epistemic ones. For instance,

- the concepts $\{o\}$, $\top$ and $\bot$ are both: objective and subjective;
- a concept of the form $A \sqcap \mathbf{K}B$, where A and B are concept names, is neither subjective nor objective.

The semantics is defined in terms of epistemic interpretations.

Definition 6.8. Let $(\mathcal{I},\mathcal{W})$ be an epistemic interpretation. The induced *interpretation function*, denoted by $\cdot^{\mathcal{I},\mathcal{W}}$, maps $\mathcal{ALCOK}$-concepts to subsets of $\mathsf{N_O}$ and $\mathcal{ALCOK}$-roles to subsets of

constructor	image of the mapping $\cdot^{\mathcal{I},\mathcal{W}}$
$P \in \mathsf{N_R}$	$P^{\mathcal{I}}$
$\mathbf{K}R$	$\bigcap_{\mathcal{J}\in\mathcal{W}} R^{\mathcal{J},\mathcal{W}}$
$A \in \mathsf{N_C}$	$A^{\mathcal{I}}$
$\top$	$\mathsf{N_O}$
$\bot$	$\emptyset$
$\{o\}$	$\{o^{\mathcal{I}}\}$
$\neg C$	$\Delta \setminus C^{\mathcal{I},\mathcal{W}}$
$C \sqcap D$	$C^{\mathcal{I},\mathcal{W}} \cap D^{\mathcal{I},\mathcal{W}}$
$C \sqcup D$	$C^{\mathcal{I},\mathcal{W}} \cup D^{\mathcal{I},\mathcal{W}}$
$\exists R.C$	$\{o \mid \exists o' \in \mathsf{N_O}.(o,o') \in R^{\mathcal{I},\mathcal{W}} \wedge o' \in C^{\mathcal{I},\mathcal{W}}\}$
$\forall R.C$	$\{o \mid \forall o' \in \mathsf{N_O}.(o,o') \in R^{\mathcal{I},\mathcal{W}} \rightarrow o' \in C^{\mathcal{I},\mathcal{W}}\}$
$\mathbf{K}C$	$\bigcap_{\mathcal{J}\in\mathcal{W}}(C^{\mathcal{J},\mathcal{W}})$

Table 6.1: Syntax and semantics of $\mathcal{ALCOK}$-roles and $\mathcal{ALCOK}$-concepts

$\mathsf{N_O} \times \mathsf{N_O}$ and is defined inductively as given in Table 6.1. The satisfaction relation between epistemic interpretations and ground Boolean $\mathcal{ALCOK}$-KBs, denoted by "$||\!\!=$", is inductively defined as follows:

$$\begin{array}{ll}
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= o \sqsubseteq C & \text{iff } o^{\mathcal{I}} \in C^{\mathcal{I},\mathcal{W}} \\
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= (o_1,o_2) \sqsubseteq R & \text{iff } (o_1^{\mathcal{I}},o_2^{\mathcal{I}}) \in R^{\mathcal{I},\mathcal{W}} \\
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= C \sqsubseteq D & \text{iff } C^{\mathcal{I},\mathcal{W}} \subseteq D^{\mathcal{I},\mathcal{W}} \\
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= o_1 \approx o_2 & \text{iff } o_1^{\mathcal{I}} = o_2^{\mathcal{I}} \\
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= \neg\psi & \text{iff } (\mathcal{I},\mathcal{W}) \not\mid\mid\!\!= \psi \\
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= \psi_1 \wedge \psi_2 & \text{iff } (\mathcal{I},\mathcal{W}) \mid\mid\!\!= \psi_1 \text{ and } (\mathcal{I},\mathcal{W}) \models \psi_2 \\
(\mathcal{I},\mathcal{W}) \mid\mid\!\!= \mathbf{K}\psi & \text{iff } (\mathcal{J},\mathcal{W}) \mid\mid\!\!= \psi \text{ for all } \mathcal{J} \in \mathcal{W}.
\end{array}$$

▲

For interpreting an objective concept, role or Boolean KB in an epistemic interpretation $(\mathcal{I},\mathcal{W})$ the epistemic state $\mathcal{W}$ is irrelevant. For an objective concept or role X it holds that $X^{\mathcal{I},\mathcal{W}} = X^{\mathcal{I}}$, and for an objective Boolean KB we have $(\mathcal{I},\mathcal{W}) \mid\mid\!\!= \psi$ iff $\mathcal{I} \models \psi$ for all epistemic interpretations $(\mathcal{I},\mathcal{W})$. If we deal with a subjective concept or role X, we sometimes write $X^{\mathcal{W}}$ instead of $X^{\mathcal{I},\mathcal{W}}$.

In presence of nested knowledge constructors the following equalities are true for an arbitrary epistemic interpretation $(\mathcal{I},\mathcal{W})$, $\mathcal{ALCOK}$-role R and $\mathcal{ALCOK}$-concept C:

$$(\mathbf{K}R)^{\mathcal{W}} = (\mathbf{KK}R)^{\mathcal{W}}; \quad (\mathbf{K}C)^{\mathcal{W}} = (\mathbf{KK}C)^{\mathcal{W}}; \quad (\mathbf{K}\neg\mathbf{K}C)^{\mathcal{I},\mathcal{W}} = (\neg\mathbf{K}C)^{\mathcal{I},\mathcal{W}}.$$

Analogously, for an arbitrary Boolean $\mathcal{ALCOK}$-KB ψ we have

$$(\mathcal{I},\mathcal{W}) \mathrel{|\!\!\models} \mathbf{K}\psi \quad \text{iff } (\mathcal{I},\mathcal{W}) \models \mathbf{K}\mathbf{K}\psi;$$
$$(\mathcal{I},\mathcal{W}) \mathrel{|\!\!\models} \mathbf{K}\neg\mathbf{K}\psi \text{ iff } (\mathcal{I},\mathcal{W}) \models \neg\mathbf{K}\psi.$$

In the following we assume that an $\mathcal{ALCOK}$-role is either an atomic role name P or an epistemic role of the form $\mathbf{K}P$ with $P \in \mathsf{N_R}$.

Next, we define the notion of epistemic entailments via an entailment relation between $\mathcal{ALCO}$-KBs and Boolean $\mathcal{ALCOK}$-KBs.

Definition 6.9. Let $\mathcal{K}$ be an (objective) $\mathcal{ALCO}$-KB consisting of an ABox and a TBox as defined in Definition 2.21. A set of interpretations $\mathcal{M}$ is called an *epistemic model* iff the following conditions are satisfied:

- for all $\mathcal{I} \in \mathcal{M}$ it holds that $\mathcal{I}$ satisfies the SNA and $\mathcal{I} \models \mathcal{K}$;
- for all sets of SNA-interpretations $\mathcal{M}'$ with $\mathcal{M} \subsetneq \mathcal{M}'$ there exists $\mathcal{J} \in \mathcal{M}'$ with $\mathcal{J} \not\models \mathcal{K}$.

Let ψ be a Boolean $\mathcal{ALCOK}$-KB. We say that ψ is *epistemically entailed* by $\mathcal{K}$, written as $\mathcal{K} \mathrel{|\!\!\models} \psi$, iff for all epistemic models $\mathcal{M}$ of $\mathcal{K}$ and all $\mathcal{I} \in \mathcal{M}$ it holds that $(\mathcal{I},\mathcal{M}) \mathrel{|\!\!\models} \psi$. ▲

If $\mathcal{K}$ is consistent under the SNA, then it has a *unique epistemic model* denoted by $\mathcal{M}(\mathcal{K})$. It holds that

$$\mathcal{M}(\mathcal{K}) = \{\mathcal{I} \mid \Delta_{\mathcal{I}} = \mathsf{N_O}, o^{\mathcal{I}} = o \text{ for all } o \in \mathsf{N_O}, \mathcal{I} \models \mathcal{K}\}.$$

Note that for example the concept inclusion $\top \sqsubseteq \{o\}$ is unsatisfiable under the SNA but is satisfied in a first-order interpretation $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\Delta_{\mathcal{I}}$ only consists of a single element and all object names are mapped to this element. This incompatibility problem with the standard semantics does not occur if nominals are disallowed in the TBox. The following Lemma is a consequence of Theorem 13 in [MRG11].

Lemma 6.10. *Let $\mathcal{K} = (\mathcal{T},\mathcal{A})$ be an $\mathcal{ALC}$-KB, ψ a Boolean $\mathcal{ALCO}$-KB, and* Obj *the finite set of all object names occurring in $\mathcal{K}$ and ψ. Furthermore, let*

$$\mathcal{K}' = (\mathcal{T}, \mathcal{A} \cup \{o \not\approx o' \mid o, o' \in \mathsf{Obj}, o \neq o'\}).$$

It holds that $\mathcal{K} \mathrel{|\!\!\models} \psi$ iff $\mathcal{K}' \models \psi$.

In $\mathcal{K}'$ we enforce unique names for the object names in Obj. Every finite model of $\mathcal{K}'$ can be lifted to an infinite model of $\mathcal{K}'$ satisfying the SNA such that both satisfy the same Boolean $\mathcal{ALCO}$-KBs. The lemma is also true in case $\mathcal{A}$ is an $\mathcal{ALCO}$-ABox and $\mathcal{T}$ an $\mathcal{ALC}$-TBox.

Example 6.11. We consider a domain with electrical devices as before (concept name *EDev*) that might have faults (concept name *Fault* and role name *HasFault*). Some faults are critical (concept name *CriticalFault*). The TBox $\mathcal{T}$ consists of the following CIs:

$$\mathcal{T} = \{\exists \mathit{HasFault}.\top \sqsubseteq \mathit{EDev}, \quad \mathit{EDev} \sqsubseteq \forall \mathit{HasFault}.\mathit{Fault}, \quad \mathit{CriticalFault} \sqsubseteq \mathit{Fault}\}.$$

The first two CIs define the domain and range of the role *HasFault*. The last one says that every critical fault is a fault. Some ground facts about the current situation are given in the following ABox $\mathcal{A}$:

$$\mathcal{A} = \{\mathsf{dev} \in \mathit{EDev}, \quad (\mathsf{dev}, \mathsf{err01}) \in \neg\mathit{HasFault}, \quad (\mathsf{dev}, \mathsf{err02}) \in \neg\mathit{HasFault}\}.$$

There is an electrical device named dev. Assume the object names err01 and err02 are identifiers of possible faults mentioned in the manual of dev. And, we know that currently dev does not have these two faults. Assume $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is everything we know about the domain. Thus, in the epistemic model of $\mathcal{K}$ the two faults err01 and err02 are the only *known* faults. But due to the open world assumption there are faults not mentioned in $\mathcal{K}$. For example, the following epistemic concept assertions about dev are epistemically entailed by $\mathcal{K}$:

$\mathsf{dev} \in \neg\mathbf{Kw}(\exists \mathit{HasFault}.\top)$	"*It is not known whether* dev *has a fault.*";
$\mathsf{dev} \in \forall(\mathbf{K}\mathit{HasFault}).\bot$	"dev *has no known faults.*";
$\mathsf{dev} \in \mathbf{K}(\forall \mathit{HasFault}.\neg\mathbf{K}\mathit{Fault})$	"*It is known that all faults of* dev *are unknown ones*".

▲

6.2.2 Consistency of Boolean KBs under the SNA

In this section, we prove that the problem of deciding consistency of Boolean $\mathcal{ALCO}$-KBs under the SNA is ExpTime-complete. ExpTime-completeness *without the SNA* is a consequence of a corresponding result for the more expressive DL $\mathcal{SHOQ}^{(\sqcap)}$ in [Lip14] (Corollary 3.34, page 54). For instance, $\top \sqsubseteq \{o_1\} \sqcup \{o_2\}$ has *no* SNA-model but is consistent without the SNA. Both semantics are not compatible. However, it is easy to reduce consistency under the SNA to standard consistency.

First, we show that a model with unnamed elements can be lifted to a model with an infinite domain.

Definition 6.12. Let $\mathsf{Obj} \subset \mathsf{N_O}$ be a finite set of object names and $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ an interpretation. The *lifting of $\mathcal{I}$ w.r.t.* Obj is an interpretation $\mathcal{I}^\ell = (\Delta_{\mathcal{I}^\ell}, \cdot^{\mathcal{I}^\ell})$ that is defined as follows:

$$\begin{aligned}
\Delta_{\mathcal{I}^\ell} &:= \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\} \cup (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}) \times \mathbb{N};\\
A^{\mathcal{I}^\ell} &:= \{o^{\mathcal{I}} \mid o^{\mathcal{I}} \in A^{\mathcal{I}}, o \in \mathsf{Obj}\} \cup\\
&\qquad \{\langle d, i\rangle \mid d \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}) \cap A^{\mathcal{I}}, i \in \mathbb{N}\} \text{ for all } A \in \mathsf{N_C};\\
P^{\mathcal{I}^\ell} &:= \{(o_1^{\mathcal{I}}, o_2^{\mathcal{I}}) \mid o_1^{\mathcal{I}}, o_2^{\mathcal{I}} \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}, (o_1^{\mathcal{I}}, o_2^{\mathcal{I}}) \in P^{\mathcal{I}}\} \cup\\
&\qquad \{(o_1, \langle d, i\rangle) \mid o_1^{\mathcal{I}} \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}, d \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}), i \in \mathbb{N}, (o_1, d) \in P^{\mathcal{I}}\} \cup\\
&\qquad \{(\langle d, i\rangle, o_2) \mid o_2^{\mathcal{I}} \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}, d \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}), i \in \mathbb{N}, (d, o_2) \in P^{\mathcal{I}}\} \cup\\
&\qquad \{(\langle d, i\rangle, \langle e, j\rangle) \mid d, e \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}) \wedge i, j \in \mathbb{N} \wedge (d, e) \in P^{\mathcal{I}}\} \text{ for all } P \in \mathsf{N_R};\\
b^{\mathcal{I}^\ell} &:= \begin{cases} b^{\mathcal{I}} & b^{\mathcal{I}} \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}\\ \langle b^{\mathcal{I}}, 0\rangle & b^{\mathcal{I}} \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}) \end{cases} \quad \text{for all } b \in \mathsf{N_O}.
\end{aligned}$$

▲

We show that lifting an interpretation does not harm its model property.

Lemma 6.13. *Let* Obj, $\mathcal{I}$ *and* $\mathcal{I}^\ell$ *be as in Definition 6.12.*

1. *Let C be an $\mathcal{ALCO}$-concept such that all object names mentioned in C are from* Obj.
   a) *For all $d \in \{o^\mathcal{I} \mid o \in \mathsf{Obj}\}$ it holds that $d \in C^\mathcal{I}$ iff $d \in C^{\mathcal{I}^\ell}$.*
   b) *For all $\langle d,i\rangle \in (\Delta_\mathcal{I} \setminus \{o^\mathcal{I} \mid o \in \mathsf{Obj}\}) \times \mathbb{N}$ it holds that $d \in C^\mathcal{I}$ iff $\langle d,i\rangle \in C^{\mathcal{I}^\ell}$.*
2. *Let ψ be a Boolean $\mathcal{ALCO}$-KB such that all object names mentioned in ψ are from* Obj. *It holds that $\mathcal{I} \models \psi$ iff $\mathcal{I}^\ell \models \psi$.*

Proof.

1. The proof is by induction on the structure of C. Let

$$b^\mathcal{I} \in \{o^\mathcal{I} \mid o \in \mathsf{Obj}\} \text{ for some } b \in \mathsf{Obj}, \text{ and}$$
$$\langle d,i\rangle \in (\Delta_\mathcal{I} \setminus \{o^\mathcal{I} \mid o \in \mathsf{Obj}\}) \times \mathbb{N}.$$

$C = \top$: a) We have $b^\mathcal{I} \in (\top)^\mathcal{I}$ iff $b^\mathcal{I} \in \Delta_\mathcal{I}$ iff $b^\mathcal{I} \in \Delta_{\mathcal{I}^\ell}$ iff $b^\mathcal{I} \in (\top)^{\mathcal{I}^\ell}$.

b) $d \in \top^\mathcal{I}$ iff $\langle d,i\rangle \in \top^{\mathcal{I}^\ell}$ because $d \in (\Delta_\mathcal{I} \setminus \{o^\mathcal{I} \mid o \in \mathsf{Obj}\})$ holds by assumption.

$C = A$: for some $A \in \mathsf{N_C}$.

a) It holds that $b^\mathcal{I} \in A^\mathcal{I}$ iff $b^\mathcal{I} \in A^{\mathcal{I}^\ell}$ with $b^\mathcal{I} \in \{o^\mathcal{I} \mid o \in \mathsf{Obj}\}$ by definition of $A^{\mathcal{I}^\ell}$.

b) It holds that $d \in A^\mathcal{I}$ iff $\langle d,i\rangle \in A^{\mathcal{I}^\ell}$ by definition of $A^{\mathcal{I}^\ell}$.

$C = \{a\}$: for some $a \in \mathsf{Obj}$.

a) It holds that $b^\mathcal{I} \in (\{a\})^\mathcal{I}$ iff $b^\mathcal{I} = a^\mathcal{I}$ iff $b^\mathcal{I} = a^\mathcal{I}$ iff $b^\mathcal{I} = a^{\mathcal{I}^\ell}$ (with $a^\mathcal{I} = a^{\mathcal{I}^\ell}$ because $a \in \mathsf{Obj}$ and $a^\mathcal{I} \in \{o^\mathcal{I} \mid o \in \mathsf{Obj}\}$) iff $b^\mathcal{I} \in (\{a\})^{\mathcal{I}^\ell}$.

b) With $d \in (\Delta_\mathcal{I} \setminus \{o^\mathcal{I} \mid o \in \mathsf{Obj}\})$ and $a \in \mathsf{Obj}$ it follows that $d \notin \{a\}^\mathcal{I}$ and $\langle d,i\rangle \notin \{a\}^{\mathcal{I}^\ell}$.

$C = D_1 \sqcap D_2$: Assume both claims are true for D_1 and D_2.

a) It holds that $b^\mathcal{I}(D_1 \sqcap D_2)^\mathcal{I}$ iff $b^\mathcal{I} \in D_1^\mathcal{I}$ and $b^\mathcal{I} \in D_2^\mathcal{I}$ iff $b^\mathcal{I} \in D_1^{\mathcal{I}^\ell}$ and $b^\mathcal{I} \in D_2^{\mathcal{I}^\ell}$ iff $b^\mathcal{I} \in (D_1 \sqcap D_2)^{\mathcal{I}^\ell}$.

b) It holds that $d(D_1 \sqcap D_2)^\mathcal{I}$ iff $d \in D_1^\mathcal{I}$ and $d \in D_2^\mathcal{I}$ iff $\langle d,i\rangle \in D_1^{\mathcal{I}^\ell}$ and $\langle d,i\rangle \in D_2^{\mathcal{I}^\ell}$ iff $\langle d,i\rangle \in (D_1 \sqcap D_2)^{\mathcal{I}^\ell}$.

$C = \neg D$: Assume both claims are true for D.

a) It holds that $b^\mathcal{I} \in (\neg D)$ iff $b^\mathcal{I} \notin D^\mathcal{I}$ iff $b^\mathcal{I} \notin D^{\mathcal{I}^\ell}$ iff $b^\mathcal{I} \in (\neg D)^{\mathcal{I}^\ell}$.

b) It holds that $d \in (\neg D)$ iff $d \notin D^\mathcal{I}$ iff $\langle d,i\rangle \notin D^{\mathcal{I}^\ell}$ iff $\langle d,i\rangle \in (\neg D)^{\mathcal{I}^\ell}$.

$C = \exists P.D$: Assume both claims are true for D.

a) It holds that $b^{\mathcal{I}} \in (\exists P.D)^{\mathcal{I}}$

iff there exists $e \in \Delta_{\mathcal{I}}$ such that $(b^{\mathcal{I}}, e) \in P^{\mathcal{I}}$ and $e \in D^{\mathcal{I}}$

iff there exists $e \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}$ such that $(b^{\mathcal{I}}, e) \in P^{\mathcal{I}}$ and $e \in D^{\mathcal{I}}$, or there exists $e \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\})$ such that $(b^{\mathcal{I}}, e) \in P^{\mathcal{I}}$ and $e \in D^{\mathcal{I}}$

iff there exists $e \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}$ such that $(b^{\mathcal{I}}, e) \in P^{\mathcal{I}^\ell}$ and $e \in D^{\mathcal{I}^\ell}$, or there exists $e \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\})$ and $i \in \mathbb{N}$ such that $(b^{\mathcal{I}}, \langle e, i\rangle) \in P^{\mathcal{I}^\ell}$ and $\langle e, i\rangle \in D^{\mathcal{I}^\ell}$

iff $b^{\mathcal{I}} \in (\exists P.D)^{\mathcal{I}^\ell}$.

b) This case is analogous to a).

2. Without loss of generality we can restrict our attention to a Boolean combination of concept inclusions. We prove the claim for a single concept inclusion. The property for Boolean combinations follows immediately.

   Let $C \sqsubseteq D$ be an $\mathcal{ALCO}$-CI that mentions only object names from Obj. We show that

   $$\mathcal{I} \models C \sqsubseteq D \text{ iff } \mathcal{I}^\ell \models C \sqsubseteq D.$$

   "⇒": Assume $\mathcal{I} \models C \sqsubseteq D$. Let $x \in C^{\mathcal{I}^\ell}$. We either have $x \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}$ or $x \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}) \times \mathbb{N}$.

   First, assume $x \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}$. We have that 1 implies $x \in C^{\mathcal{I}}$ and with $\mathcal{I} \models C \sqsubseteq D$ we get $x \in D^{\mathcal{I}}$. Again 1 yields $x \in D^{\mathcal{I}^\ell}$.

   Second, assume $x = \langle d, i\rangle$ for some $d \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\})$ and $i \in \mathbb{N}$. $\langle d, i\rangle \in C^{\mathcal{I}^\ell}$ implies $d \in C^{\mathcal{I}}$ with 1. Since $\mathcal{I} \models C \sqsubseteq D$ we get $d \in D^{\mathcal{I}}$, and $\langle d, i\rangle \in D^{\mathcal{I}^\ell}$ with 1. It follows that $\mathcal{I}^\ell \models C \sqsubseteq D$.

   "⇐": Assume $\mathcal{I}^\ell \models C \sqsubseteq D$. Let $d \in C^{\mathcal{I}}$.

   First, assume $d \in \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\}$. With 1 and the assumption $\mathcal{I}^\ell \models C \sqsubseteq D$ it follows that $d \in D^{\mathcal{I}}$.

   Second, assume $d \in (\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\})$. We have $d \in C^{\mathcal{I}}$ implies $\langle d, i\rangle \in C^{\mathcal{I}^\ell}$ for all $i \in \mathbb{N}$ which implies $\langle d, i\rangle \in D^{\mathcal{I}^\ell}$ with $\mathcal{I}^\ell \models C \sqsubseteq D$. Using 1 we obtain $d \in D^{\mathcal{I}}$. Consequently, $\mathcal{I} \models C \sqsubseteq D$.

□

Any interpretation $\mathcal{I}$ that has at least one unnamed element (i.e. $(\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}\})$ is non-empty) can be lifted to an interpretation with a countably infinite domain. If in addition all object names in Obj have unique names, then the we can define an SNA-interpretation that is isomorphic to the lifting and models the same Boolean $\mathcal{ALCO}$-KBs over object names from Obj. For the proof we need the notion of a renamed interpretation.

Definition 6.14 (renamed interpretation). Let $\mathcal{Y} = (\Delta_{\mathcal{Y}}, \cdot^{\mathcal{Y}})$ be an interpretation with an arbitrary countably infinite domain $\Delta_{\mathcal{Y}}$. Let $\iota : \Delta_{\mathcal{Y}} \rightarrow \mathsf{N_O}$ be a bijection. The *renamed* *interpretation of $\mathcal{Y}$ with ι*, denoted by $\iota(\mathcal{Y})$, is defined as follows:

$$\begin{aligned}
\Delta_{\iota(\mathcal{Y})} &:= \mathsf{N_O};\\
o^{\iota(\mathcal{Y})} &:= o \text{ for all } o \in \mathsf{N_O};\\
A^{\iota(\mathcal{Y})} &:= \{\iota(d) \mid d \in A^{\mathcal{Y}}\} \text{ for all } A \in \mathsf{N_C};\\
P^{\iota(\mathcal{Y})} &:= \{(\iota(d), \iota(e)) \mid (d,e) \in P^{\mathcal{Y}}\} \text{ for all } P \in \mathsf{N_R}.
\end{aligned}$$

▲

Lemma 6.15. *Let $\mathsf{Obj} \subset \mathsf{N_O}$ be a finite set of object names and $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ an interpretation such that $\Delta_{\mathcal{I}}$ is countably infinite and $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ holds for all $a, b \in \mathsf{Obj}$ with $a \neq b$. Furthermore, let $\iota : \Delta_{\mathcal{I}} \to \mathsf{N_O}$ be a bijection with $\iota(o^{\mathcal{I}}) = o$ for all $o \in \mathsf{Obj}$.*

For any Boolean $\mathcal{ALCO}$-KB ψ that mentions only object names from Obj it holds that

$$\mathcal{I} \models \psi \text{ iff } \iota(\mathcal{I}) \models \psi.$$

Proof. In case $\mathcal{I}$ satisifies $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ for all $a, b \in \mathsf{Obj}$ with $a \neq b$ a bijection $\iota : \Delta_{\mathcal{I}} \to \mathsf{N_O}$ with $\iota(o^{\mathcal{I}}) = o$ for all $o \in \mathsf{Obj}$ exists.

For a role name $P \in \mathsf{N_R}$ we have

$$(d,e) \in P^{\mathcal{I}} \text{ iff } (\iota(d), \iota(e)) \in P^{\iota(\mathcal{I})}$$

by definition of $\iota(\mathcal{I})$. Let C be an $\mathcal{ALCO}$-concept that mentions only object names from Obj. By induction on the structure of C it can be shown that $d \in C^{\mathcal{I}}$ iff $\iota(d) \in C^{\iota(\mathcal{I})}$ which implies $\mathcal{I} \models \psi$ iff $\iota(\mathcal{I}) \models \psi$ for Boolean $\mathcal{ALCO}$-KBs ψ with only object names from Obj. □

The existence of an unnamed element can be enforced with a Boolean $\mathcal{ALCO}$-KB and also the unique name assumption for a finite set of object names can be expressed using inequality assertions. Therefore, consistency under the SNA boils down to consistency w.r.t. the standard semantics.

Lemma 6.16. *Let ψ be a Boolean $\mathcal{ALCO}$-KB and $\mathsf{Obj}(\psi)$ the set of all object names mentioned in ψ. Furthermore, let*

$$\mathsf{unnamed}(\psi) := \neg\Big(\top \sqsubseteq \bigsqcup_{o \in \mathsf{Obj}(\psi)} \{o\}\Big) \quad \textit{and} \quad \mathsf{UNA}(\psi) := \bigwedge_{\substack{a,b \in \mathsf{Obj}(\psi);\\ a \neq b}} \neg\big(\{a\} \sqsubseteq \{b\}\big).$$

It holds that ψ has a model that satisfies the SNA iff

$$\psi \wedge \mathsf{unnamed}(\psi) \wedge \mathsf{UNA}(\psi)$$

is consistent under the standard semantics (without the SNA and without the UNA).

Proof. "⇒": Let $\mathcal{I}$ be an interpretation satisfying the SNA such that $\mathcal{I} \models \psi$ holds. It is easy to see that $\mathcal{I} \models \mathsf{unnamed}(\psi) \wedge \mathsf{UNA}(\psi)$.

"⇐": Assume there is an arbitrary interpretation $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ such that

$$\mathcal{I} \models \psi \wedge \mathsf{unnamed}(\psi) \wedge \mathsf{UNA}(\psi)$$

holds. Given $\mathcal{I}$ we construct a model of ψ satisfying the SNA. It follows that

$$(\Delta_{\mathcal{I}} \setminus \{o^{\mathcal{I}} \mid o \in \mathsf{Obj}(\psi)\})$$

is non-empty. Consequently, the lifting of $\mathcal{I}$ w.r.t. $\mathsf{Obj}(\psi)$, denoted by $\mathcal{I}^{\ell}$, has a countably infinite domain. Lemma 6.13 implies

$$\mathcal{I}^{\ell} \models \psi \wedge \mathsf{unnamed}(\psi) \wedge \mathsf{UNA}(\psi).$$

Therefore, a bijection $\iota : \Delta_{\mathcal{I}^{\ell}} \to \mathsf{N_O}$ with $\iota(o^{\mathcal{I}^{\ell}}) = o$ for all $o \in \mathsf{Obj}(\psi)$ exists. Let $\mathcal{J} := \iota(\mathcal{I}^{\ell})$ be the renamed interpretation. Lemma 6.15 implies

$$\mathcal{J} \models \psi.$$

Obviously, $\mathcal{J}$ satisfies the SNA. □

As a consequence we get that consistency under the SNA is in ExpTime. In $\mathcal{ALC}$ the SNA is without loss of generality. Thus, the ExpTime lower bound comes from the consistency problem of $\mathcal{ALC}$-KBs.

Corollary 6.17. *Consistency of Boolean $\mathcal{ALCO}$-KBs under the SNA is* ExpTime*-complete.*

6.2.3 Actions with Sensing Results

We define an extension of local effect $\mathcal{ALCO}$-action theories (see Definition 2.29) where actions now also have sensing results. In our model sensing corresponds to the ability of an agent to observe the truth of an objective Boolean $\mathcal{ALCO}$-KB.

Definition 6.18. An *epistemic $\mathcal{ALCO}$-action theory* is a tuple

$$\Sigma_{\mathsf{s}} = (\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}), \mathsf{sense}),$$

that consists of a

- local effect $\mathcal{ALCO}$-action theory $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$;
- a function sense that associates any of the action terms $\alpha(\bar{t}) \in \mathsf{Act}$ to an (objective) Boolean $\mathcal{ALCO}$-KB, denoted by $\mathsf{sense}(\alpha(\bar{t}))$, that mentions only object terms included in the arguments $\bar{t}$.

For any $\alpha(\bar{o}) \in \mathsf{ground}(\mathsf{Act})$ such that $\alpha(\bar{o})$ is the ground instantiation of some $\alpha(\bar{t}) \in \mathsf{Act}$ with ν we define

$$\mathsf{sense}(\alpha(\bar{o})) := \psi^{\nu} \text{ with } \psi = \mathsf{sense}(\alpha(\bar{t})).$$

In the following we often write the action theory as a 5-tuple $\Sigma_{\mathsf{s}} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ instead of $\Sigma_{\mathsf{s}} = (\Sigma, \mathsf{sense})$ with $\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff})$. ▲

In addition to the preconditions and effects, an epistemic $\mathcal{ALCO}$-action theory explicitly provides a Boolean KBs for each action. By an execution of a ground action $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$

the agent is able to observe the truth of the Boolean $\mathcal{ALCO}$-KB $\mathsf{sense}(\boldsymbol{\alpha})$. $\mathsf{sense}(\boldsymbol{\alpha})$ is objective since sensors only provide information about the outside world.

In case an action $\alpha(\bar{t})$ is a purely physical action (without any sensing result) we define $\mathsf{sense}(\alpha(\bar{t})) := \textsc{True}$. Note that effect conditions are restricted to be objective as well since eff is supposed to only encode physical effects.

The *size of an epistemic $\mathcal{ALCO}$-action theory* $\Sigma_{\mathsf{s}} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$, denoted by $|\Sigma_{\mathsf{s}}|$, is given by the number of all symbols needed to write down $\mathcal{K}$, the preconditions, effects, and sensing property for all actions in Act.

The semantics of an epistemic $\mathcal{ALCO}$-action theory is given terms of an epistemic FO-DS.

Definition 6.19. Let $\Sigma_{\mathsf{s}} = (\Sigma = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}), \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory. The *epistemic FO-DS induced by* Σ_{s}, denoted by $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})$, is an epistemic FO-DS

$$\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}}, \sim_{\mathsf{s}}),$$

that consists of the FO-DS $\mathfrak{D}(\Sigma) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}})$ induced by Σ under the SNA and a sensing compatibility relation that is given by

$$\sim_{\mathsf{s}} := \{(\mathcal{I}, \boldsymbol{\alpha}, \mathcal{J}) \in \mathbb{I} \times \mathsf{ground}(\mathsf{Act}) \times \mathbb{I} \mid \mathcal{I} \models \mathsf{sense}(\boldsymbol{\alpha}) \text{ iff } \mathcal{J} \models \mathsf{sense}(\boldsymbol{\alpha})\}.$$

▲

Example 6.20. We describe an action theory in our domain with electrical devices and faults. The initial KB is given by

$$\begin{aligned}
\mathcal{T} &= \{\exists \mathit{HasFault}.\top \sqsubseteq \mathit{EDev}, \quad \mathit{EDev} \sqsubseteq \forall \mathit{HasFault}.\mathit{Fault}, \quad \mathit{CriticalFault} \sqsubseteq \mathit{Fault}\};\\
\mathcal{A} &= \{\mathsf{dev} \in (\mathit{EDev} \sqcap \neg \mathit{On}), \quad \mathsf{err01} \in \mathit{Fault}\}.
\end{aligned}$$

The TBox is the same as in Example 6.11. The ABox describes an electrical device dev that is currently not on and the agent is aware of a single fault named $\mathsf{err01}$. The agent has the following actions at its disposal:

$$\mathsf{Act} = \{\texttt{turn-on}(x), \texttt{repair}(x,y), \texttt{sense-fault}(x,y), \texttt{sense-on}(x)\}.$$

We have

$$\begin{aligned}
\mathsf{pre}(\texttt{turn-on}(x)) &:= \{(x \in \mathit{EDev})\};\\
\mathsf{eff}(\texttt{turn-on}(x)) &:= \big\{(x \in (\neg \exists \mathit{HasFault}.\mathit{CriticalFault})) \rhd \langle \mathit{On}, \{x\}\rangle^{+}\big\};\\
\mathsf{sense}(\texttt{turn-on}(x)) &:= (x \in \top).
\end{aligned}$$

The agent is able to turn on electrical devices. The action has the desired effect only if the device does not have a critical fault. It is possible to repair a fault y of a device x:

$$\begin{aligned}
\mathsf{pre}(\texttt{repair}(x,y)) &:= \{(x \in \mathit{EDev}), (y \in \mathit{Fault})\};\\
\mathsf{eff}(\texttt{repair}(x,y)) &:= \big\{\langle \mathit{HasFault}, \{(x,y)\}\rangle^{-}\big\};\\
\mathsf{sense}(\texttt{repair}(x,y)) &:= (x \in \top).
\end{aligned}$$

Both actions are purely physical actions and do not produce any sensing outcome. The action sense-on(x) is a sensing action representing the agent's ability to perceive whether $x \sqsubseteq \mathit{On}$ is true in the real world:

$$\begin{aligned}\mathsf{pre}(\texttt{sense-on}(x)) &:= \{(x \sqsubseteq \mathit{EDev})\};\\ \mathsf{eff}(\texttt{sense-on}(x)) &:= \emptyset;\\ \mathsf{sense}(\texttt{sense-on}(x)) &:= (x \sqsubseteq \mathit{On}).\end{aligned}$$

Furthermore, the agent is able to check whether a device x has a certain fault y:

$$\begin{aligned}\mathsf{pre}(\texttt{sense-fault}(x,y)) &:= \{(x \sqsubseteq \mathit{EDev}), (y \sqsubseteq \mathit{Fault})\};\\ \mathsf{eff}(\texttt{sense-fault}(x,y)) &:= \emptyset;\\ \mathsf{sense}(\texttt{sense-fault}(x,y)) &:= ((x,y) \sqsubseteq \mathit{HasFault}).\end{aligned}$$

▲

Now, we are ready to define the epistemic projection problem.

Definition 6.21. Let

- $\Sigma_\mathsf{s} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory,
- $\sigma \in \mathsf{ground}(\mathsf{Act})^*$ a ground action sequence, and
- ψ a Boolean $\mathcal{ALCOK}$-KB (called *projection query*).

Furthermore, let $\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_\mathsf{poss}, \sim_\mathsf{s})$ be the epistemic FO-DS induced by Σ_s. We say that *ψ is valid after doing σ in Σ_s* iff for all $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ it holds that $(\mathcal{J}, \mathcal{M}) \mid\!\models \psi$ where $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})} (\mathcal{J}, \mathcal{M})$.

The *epistemic projection problem* takes Σ_s, σ and ψ as input and asks whether ψ is valid after doing σ in Σ_s. ▲

The initial KB $\mathcal{K}$ represents everything the agent knows about the initial situation. The unique epistemic model of $\mathcal{K}$ is the only initial epistemic state. The projection problem asks whether the subjective projection query is true in all possible epistemic states that evolve from the epistemic model of $\mathcal{K}$ by executing a given ground action sequence.

Example 6.22. We consider the action theory from Example 6.20. In $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ it is *known* that the device dev is *not on*. The concept assertion $(\mathsf{dev} \sqsubseteq \mathbf{K}(\neg \mathit{On}))$ is epistemically entailed by $\mathcal{K}$. Now, consider the following action sequences:

$$\sigma_1 := \texttt{turn-on}(\mathsf{dev}) \quad \sigma_2 = \texttt{turn-on}(\mathsf{dev})\ \texttt{sense-on}(\mathsf{dev}).$$

Initially, the agent does not know whether the effect condition

$$(\mathsf{dev} \sqsubseteq (\neg \exists \mathit{HasFault}.\mathit{CriticalFault}))$$

of turn-on(dev) is satisfied. There are models in $\mathcal{M}(\mathcal{K})$ where dev has a critical fault and some where this is not the case. Thus, the agent is not able to foresee the outcome of σ_1. It holds that $(\mathsf{dev} \sqsubseteq \neg\mathbf{Kw}\mathit{On})$ is valid after doing σ_1. The agent looses its knowledge about the status of dev. If the agent now in turn executes `sense-on`(dev), it will also come to know whether dev has a critical fault or not, i.e. both assertions

$$(\mathsf{dev} \sqsubseteq \mathbf{Kw}(\exists \mathit{HasFault}.\mathit{CriticalFault})) \text{ and } (\mathsf{dev} \sqsubseteq \mathbf{Kw}\mathit{On})$$

are valid after doing σ_2. ▲

In our setting the agent only has incomplete information about the initial situation. Although the actions are deterministic, the outcome of an action might be only partially observable due to conditional effects.

6.3 Relation to the Epistemic Situation Calculus

To justify our action semantics we provide an embedding into the epistemic Situation Calculus formulated in the logic $\mathcal{ES}$ [LL04; LL11].

In $\mathcal{ES}$ (Section 6.3.1) it is possible to represent Reiter's *basic action theories* (BATs) [Rei91; Rei01a] of the Situation Calculus. The language also offers modalities for talking about the knowledge of a single agent. We consider BATs formulated in $\mathcal{ES}$ as an axiomatic approach for representing (possibly epistemic) FO-DSs. In Section 6.3.2 we show that FO-definable ground actions can be axiomatized in a BAT and vice versa: ground actions definable in a BAT are FO-definable. And we prove a characterization of the epistemic projection problem with epistemic $\mathcal{ALCO}$-action theories as an entailment problem formulated in $\mathcal{ES}$.

6.3.1 Basic Notion of the Epistemic Situation Calculus

The logic $\mathcal{ES}$ was first introduced by Lakemeyer and Levesque in [LL04; LL11] as a logic for reasoning about knowledge and action. $\mathcal{ES}$ can be viewed as an extension of FO with specific modal operators for talking about actions and knowledge. Actions are treated as a separate sort in the logic and it is possible to quantify over actions. We use the same fixed signature of predicate, variable, object and action names ($\mathsf{N_F}$,$\mathsf{N_V}$,$\mathsf{N_O}$, $\mathsf{N_A}$) as introduced in Definition 2.1 and 2.5 plus some additional symbols.

Definition 6.23 (terms and standard names)**.** Terms of sort object and terms of sort action are distinguished. The sets of symbols $\mathsf{N_V}$ (object variable names), $\mathsf{N_O}$ (object names) and $\mathsf{N_A}$ (action names) are given as in the Definitions 2.1 and 2.5. In addition there is one distinguished *variable name a of sort action*.

- Every element of $\mathsf{N_V} \cup \mathsf{N_O}$ is a *term of sort object* and
- every element of $\mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O}) \cup \{a\}$ is a *term of sort action*.

Ground terms (terms without variable names) of both sorts are also treated as standard names.

- The sets of all ground terms of sort object and of sort action are denoted by $\mathcal{N}_O$ (*object standard names*) and $\mathcal{N}_A$ (*action standard names*), respectively.
- $\mathcal{N} = \mathcal{N}_O \cup \mathcal{N}_A$ denotes the set of all standard names.

Thus, we have $\mathcal{N}_O = \mathsf{N_O}$. To construct atomic formulas the set of predicate names (called *fluent predicates*) $\mathsf{N_F}$ with arguments of sort object is given as in Definition 2.1. In addition there are two distinguished unary predicates *Poss* and *SF* (not contained in $\mathsf{N_F}$) that take an action as argument. ▲

Definition 6.24 ($\mathcal{ES}$-formulas)**.** The set of all *$\mathcal{ES}$-formulas* is defined as the least set satisfying the following conditions:

- If $t_1, ..., t_k$ are terms of sort object, t_a a term of sort action and $F \in \mathsf{N_F}$ a k-ary fluent, then $F(t_1, ..., t_k)$, $Poss(t_a)$ and $SF(t_a)$ are formulas.
- If t_1 and t_2 are terms of the same sort, then $t_1 \approx t_2$ is a formula.
- If ϕ and ϕ' are formulas and v a variable, then $\neg\phi$, $\phi \wedge \phi'$, $\phi \vee \phi'$, $\exists v.\phi$ and $\forall v.\phi$ are formulas.
- If ϕ is a formula and t^a a term of sort action, then $[t^a]\phi$ and $\Box\phi$ are formulas.
- If ϕ is a formula, then also $Know(\phi)$ and $OKnow(\phi)$ are formulas.

We understand the additional Boolean connectives $\rightarrow$ and $\equiv$ as the usual abbreviations. Furthermore, the symbol TRUE with TRUE $:= o \approx o$, for some arbitrary but fixed $o \in \mathsf{N_O}$ stands for a tautology. ▲

The fluent predicates from $\mathsf{N_F}$ describe relations among objects. The formula $Poss(t^a)$ expresses that action t^a is possible. $SF(t^a)$ is true if t^a provides the truth value "true" as its sensing result. There are the usual first-order logical connectives including quantification over objects and actions. In the Situation Calculus the current situation of a world is characterized by the sequence of actions that have occurred so far. To refer to future situations there are two *action modalities*:

$[t^a]\phi$ means that "*ϕ is true after the action t^a has occurred*" and

$\Box\phi$ should be read as "*ϕ is true after any sequence of actions*".

Note that the modality $[t^a]$ encloses an arbitrary (possibly non-ground) action term. A formula of the form $Know(\phi)$ should be read as "*ϕ is known*" and $OKnow(\phi)$ as "*ϕ is all that is known*".

We distinguish different sets of formulas:

- *Static formulas* are formulas without action modalities.
- *Fluent formulas* are static formulas without action terms.
- *Primitive formulas* are of the form $F(\bar{o})$ or $Poss(\boldsymbol{\alpha})$ or $SF(\boldsymbol{\alpha})$ with $F \in \mathsf{N_F}$, $\bar{o} \in (\mathcal{N}_O)^{\mathsf{ar}(F)}$ and $\boldsymbol{\alpha} \in \mathcal{N}_A$. The set of all primitive formulas is denoted by $\mathcal{P}_F$.
- *Objective formulas* do not mention *Know* or *OKnow*.
- In *Subjective formulas* all predicate names occur within the scope of a *Know* or *OKnow*.

For an action sequence $\sigma = \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_m \in \mathcal{N}_A^*$ and a formula ϕ we write $[\sigma]\phi$ as an abbreviation of $[\boldsymbol{\alpha}_1]([\boldsymbol{\alpha}_2](\cdots([\boldsymbol{\alpha}_m]\phi)))$. For a k-tuple of variables $\bar{v} = (v_1, \cdots, v_k)$ we write $\exists\bar{v}.\phi$ as an abbreviation of $\exists v_1.(\exists v_2.(\cdots(\exists v_k.\phi)))$. And for a k-tuple of terms $\bar{t} = (t_1, \cdots, t_k)$ the formula $\bar{v} \approx \bar{t}$ abbreviates $v_1 \approx t_1 \wedge \cdots \wedge v_k \approx t_k$.

The semantics is given in terms of worlds that have a two-dimensional structure. A world assigns truth values to all primitive formulas (the *first dimension*) relative to the sequence of ground actions that have occurred so far (the *second dimension*).

Definition 6.25. A *world* w is a total function of the form

$$w : \mathcal{P}_F \times \mathcal{N}_A^* \to \{0,1\}.$$

The set of all worlds is denoted by W. The symbol $\langle\rangle$ is used to denote the empty sequence of actions. A set of worlds $e \subseteq W$ is called *epistemic state*. ▲

To define the meaning of the epistemic operators *Know* and *OKnow* some additional notions are needed.

Definition 6.26. Let $w, w' \in W$ be two worlds, $\sigma \in \mathcal{N}_A^*$ an action sequence and $\boldsymbol{\alpha} \in \mathcal{N}_A$ a ground action. *Sensing compatibility* of w and w' w.r.t. σ, denoted by $w \simeq_\sigma w'$, is defined by induction on the length of σ:

- It holds that $w \simeq_{\langle\rangle} w'$.
- It holds that $w \simeq_{\sigma\cdot\boldsymbol{\alpha}} w'$ iff $w \simeq_\sigma w'$ and $w[SF(\boldsymbol{\alpha}), \sigma] = w'[SF(\boldsymbol{\alpha}), \sigma]$.

Let $w \in W$ and $\sigma \in \mathcal{N}_A^*$. The *progression of w through σ* is a world w_σ such that

$$w_\sigma[\xi, \sigma'] = w[\xi, \sigma \cdot \sigma'] \text{ for all } \xi \in \mathcal{P}_F \text{ and all } \sigma' \in \mathcal{N}_A^*.$$

Let e be an epistemic state, $w \in W$ and $\sigma \in \mathcal{N}_A^*$. The *progression of e through σ* w.r.t. w, denoted by e_σ^w, is an epistemic state, that is defined as follows:

$$e_\sigma^w := \{w'_\sigma \mid w' \in e, w' \simeq_\sigma w\}.$$

▲

Quantifiers are interpreted by substituting variables with standard names of the corresponding sort. Let $v \in \mathsf{N}_\mathsf{V} \cup \{a\}$ be a variable, ϕ a formula, $\mathcal{N}_v$ the set of all standard names of the same sort as v and $n \in \mathcal{N}_v$ a standard name. ϕ_n^v denotes the formula that is the result of simultaneously replacing all free occurrences of v in ϕ by n. The notation is also used for tuples of variables and tuples of standard names.

Now we are ready to define the truth of sentences (formulas without variables) in an epistemic state and a world given the history of actions, i.e. the sequence of actions that have occurred so far.

Definition 6.27 (satisfaction of sentences). Let $e \subseteq W$ be an epistemic state $w \in e$ a world and $\sigma \in \mathcal{N}_A^*$ an action sequence. The satisfaction relation $\models_{\mathcal{ES}}$ between e, w, σ and sentences is defined inductively by the following conditions:

1. $e, w, \sigma \models_{\mathcal{ES}} \rho$ iff $w[\rho, \sigma] = 1$ for an $\rho \in \mathcal{P}_F$;

2. $e, w, \sigma \models_{\mathcal{ES}} (t_1 \approx t_2)$ iff t_1 and t_2 are identical;
3. $e, w, \sigma \models_{\mathcal{ES}} \neg\varphi$ iff $e, w, \sigma \not\models \varphi$;
4. $e, w, \sigma \models_{\mathcal{ES}} \varphi_1 \wedge \varphi_2$ iff $e, w, \sigma \models_{\mathcal{ES}} \varphi_1$ and $e, w, \sigma \models_{\mathcal{ES}} \varphi_2$;
5. $e, w, \sigma \models_{\mathcal{ES}} \varphi_1 \vee \varphi_2$ iff $e, w, \sigma \models_{\mathcal{ES}} \varphi_1$ or $e, w, \sigma \models_{\mathcal{ES}} \varphi_2$;
6. $e, w, \sigma \models_{\mathcal{ES}} \exists v.\phi$ iff $e, w, \sigma \models_{\mathcal{ES}} \phi^v_n$ for some $n \in \mathcal{N}_v$;
7. $e, w, \sigma \models_{\mathcal{ES}} \forall v.\phi$ iff $e, w, \sigma \models_{\mathcal{ES}} \phi^v_n$ for all $n \in \mathcal{N}_v$;
8. $e, w, \sigma \models_{\mathcal{ES}} [\boldsymbol{\alpha}]\varphi$ iff $e, w, \sigma \cdot \boldsymbol{\alpha} \models_{\mathcal{ES}} \varphi$;
9. $e, w, \sigma \models_{\mathcal{ES}} \Box\varphi$ iff $e, w, \sigma \cdot \sigma' \models_{\mathcal{ES}} \varphi$ for all $\sigma' \in \mathcal{N}_A^*$;
10. $e, w, \sigma \models_{\mathcal{ES}} \mathit{Know}(\phi)$ iff for all $w' \in e^w_\sigma$: $e^w_\sigma, w', \langle\rangle \models \phi$;
11. $e, w, \sigma \models_{\mathcal{ES}} \mathit{OKnow}(\phi)$ iff for all $w' \in W$: $w' \in e^w_\sigma$ iff $e^w_\sigma, w', \langle\rangle \models \phi$.

We often write $e, w \models_{\mathcal{ES}} \varphi$ instead of $e, w, \langle\rangle \models \varphi$. Let Γ be a set of sentences. We write $\Gamma \models_{\mathcal{ES}} \varphi$ iff for all $e \subseteq W$ and $w \in e$ we have $e, w \models_{\mathcal{ES}} \Gamma$ ($e, w \models \psi$ for all $\psi \in \Gamma$) implies $e, w \models_{\mathcal{ES}} \varphi$. In case of objective sentences we omit the epistemic state. ▲

In $\mathcal{ES}$ we can express basic action theories of the epistemic situation calculus as follows.

Definition 6.28. Let $\mathcal{F} \subset \mathsf{N_F}$ be a finite set of fluents A *basic action theory* (BAT)

$$\Sigma = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$$

over $\mathcal{F}$ is a set of sentences mentioning only fluents from $\mathcal{F}$ and satisfying the following conditions.

1. Σ_0, the *initial theory*, is a finite set of objective fluent sentences describing the initial state of the world.
2. Σ_{pre} is a set containing a single *precondition axiom* of the form

   $$\forall a.\Box(\mathit{Poss}(a) \equiv \vartheta),$$

   where ϑ is a disjunction of formulas the form

   $$\exists \bar{x}.(a \approx \alpha(\bar{t}) \wedge \varphi),$$

   such that $\alpha(\bar{t}) \in \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$ and φ is an objective fluent formula.
3. Σ_{post} is a finite set of *successor state axioms* (SSAs), one for each fluent in $\mathcal{F}$, incorporating Reiter's [Rei01a] solution to the frame problem, and encoding the effects the actions have on the different fluents. The SSA for a fluent predicate $F \in \mathcal{F}$ has the form

   $$\forall a.\forall \bar{x}.\Box(\ ([a]F(\bar{x})) \ \equiv \ (\gamma_F^+ \vee F(\bar{x}) \wedge \neg\gamma_F^-)\ ),$$

where the *positive effect condition* γ_F^+ and *negative effect condition* γ_F^- are static formulas with free variables $\bar{x}$ and a. We assume that the positive and negative effect conditions γ_F^+ and γ_F^- are of a certain normal form. Both, γ_F^+ and γ_F^- are (possibly empty) disjunctions of formulas of the form

$$\exists \bar{y}.(a \approx \alpha(\bar{t}) \land \phi),$$

where $\alpha(\bar{t}) \in \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$ such that the variables in $\bar{t}$ are among $\bar{y}$ and $\bar{x}$, and ϕ is an objective fluent formula with free variables among $\bar{y}$ and $\bar{x}$. Note that $\bar{y}$ can be empty in case the existential quantifiers at the front are omitted.

4. Σ_{sense} is a set containing a single sentence of the form

$$\forall a.\Box(Poss(a) \equiv \vartheta),$$

where ϑ is a disjunction of formulas of the form

$$\exists \bar{x}.(a \approx \alpha(\bar{t}) \land \varphi),$$

such that $\alpha(\bar{t}) \in \mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$ and φ is an objective fluent formula.

▲

The specific shape of the SSAs correspond to the normal form as introduced in [Rei01a].

Example 6.29. We axiomatize (some parts of the) domain from Example 6.20 using a BAT. Here, we consider the actions $\texttt{turn-on}(x)$, $\texttt{turn-off}(x)$ and $\texttt{sense-on}(x)$. The initial theory Σ_0 consists of the FO translations of the axioms in $\mathcal{T}$ and $\mathcal{A}$ given in Example 6.20. The precondition axiom in Σ_{pre} requires that actions need to be instantiated with an electrical device (unary fluent *EDev*):

$$\begin{aligned}\forall a.\ \Box\Big(\ Poss(a) \equiv\ \big(&\exists x.(a \approx \texttt{turn-on}(x) \land EDev(x)) \lor \\ &\exists x.(a \approx \texttt{turn-off}(x) \land EDev(x)) \lor \\ &\exists x.(a \approx \texttt{sense-on}(x) \land EDev(x))\ \big)\ \Big).\end{aligned}$$

The SSA for *On* in Σ_{post} is given by

$$\begin{aligned}\forall x.\Box(\ [a]On(x)\ \equiv\ (&\ a \approx \texttt{turn-on}(x) \land \neg\exists y.(HasFault(x,y) \land CriticalFault(y))\ ) \lor \\ &On(x) \land a \not\approx \texttt{turn-off}(x)\ ).\end{aligned}$$

In words: it always holds that after executing an action an object x is on iff x was turned on and had no critical fault or it was on before and was not turned off. The fluent *CriticalFault* is not changing and has the SSA

$$\forall x.\Box(\ [a]CriticalFault(x)\ \equiv\ CriticalFault(x)\ ).$$

The fluent *HasFault* is handled analogously. Σ_{sense} consists of the following axiom:

$$\begin{aligned}\forall a.\,\Box\Big(\; SF(a) \equiv \big(&\exists x.(a \approx \texttt{turn-on}(x)) \lor \\ &\exists x.(a \approx \texttt{turn-off}(x)) \lor \\ &\exists x.(a \approx \texttt{sense-on}(x) \land On(x)) \;\big)\Big).\end{aligned}$$

The actions $\texttt{turn-on}(x)$ and $\texttt{turn-off}(x)$ always return true as their sensing result. The truth value of $SF(\texttt{sense-on}(x))$ is always equal to the truth value of $On(x)$. ▲

Let Σ be a BAT, $\sigma \in \mathcal{N}_A^*$ and ϕ a (possibly subjective) fluent formula. The *epistemic projection problem* can be formalized in $\mathcal{ES}$ as the following entailment problem:

$$\Sigma \land OKnow(\Sigma) \models_{\mathcal{ES}} [\sigma]\phi.$$

Thus, we assume the BAT Σ is *all* that is known about the world and then check whether the truth of ϕ after doing σ is entailed.

6.3.2 Basic Action Theories and Epistemic FO-DSs

The goal in this section is to characterize the epistemic projection problem in our DL-based language as an entailment problem in the epistemic Situation Calculus. As a first step towards this characterization we have to equivalently describe the semantics of a BAT formulated in $\mathcal{ES}$ in terms of an epistemic FO-DS. After doing this we will also establish the relationship between our notion of FO-definability of ground actions in an FO-DS and actions definable in a BAT.

To simplify the presentation we first make a few assumptions. Note that the FO formulas considered in Section 2.1 and $\mathcal{ES}$ fluent formulas (formulas without action terms or modalities) are built using the same sets of symbols including the fixed vocabulary $\mathsf{N_F}, \mathsf{N_O}$ and $\mathsf{N_V}$. Syntactically, we do not distinguish between the two sets of formulas. In the following we therefore view $\mathcal{ES}$ fluent formulas also as FO formulas (interpreted under the SNA) and FO formulas as $\mathcal{ES}$ fluent formulas. This is done to avoid additional notation in form of mappings between FO formulas and $\mathcal{ES}$ fluent formulas.

As a first step we define a world that initially coincides with a given interpretation and satisfies a given BAT.

Definition 6.30. Let $\Sigma = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$ be a BAT over a finite set $\mathcal{F} \subset \mathsf{N_F}$ and $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ an interpretation with $\Delta_{\mathcal{I}} = \mathsf{N_O}$ ($= \mathcal{N}_O$) and $o^{\mathcal{I}} = o$ for all $o \in \mathsf{N_O}$.

$w_{\mathcal{I}}^{\Sigma}$ denotes a world satisfying the following conditions:

- For all $F(\bar{o}) \in \mathcal{P}_F$ with $F \in \mathcal{F}$ it holds that

  $w_{\mathcal{I}}^{\Sigma}[F(\bar{o}), \langle\rangle] = 1$ iff $\bar{o} \in F^{\mathcal{I}}$ and

  for all $\sigma \cdot \boldsymbol{\alpha} \in \mathcal{N}_A^*$: $\;w_{\mathcal{I}}^{\Sigma}[F(\bar{o}), \sigma \cdot \boldsymbol{\alpha}] = 1$ iff $w_{\mathcal{I}}^{\Sigma}, \sigma \models_{\mathcal{ES}} \big(\gamma_F^+ \lor F(\bar{x}) \land \neg\gamma_F^-\big)^{a\;\bar{x}}_{\boldsymbol{\alpha}\;\bar{o}}$,

  where $\gamma_F^+ \lor F(\bar{x}) \land \neg\gamma_F^-$ is the rhs of the SSA of F in Σ_{post}.

- For all $F(\bar{o}) \in \mathcal{P}_F$ with $F \in \mathsf{N_F} \setminus \mathcal{F}$ it holds that

$$\text{for all } \sigma \in \mathcal{N}_A^* : \;\; w_{\mathcal{I}}^{\Sigma}[F(\bar{o}), \sigma] = 1 \text{ iff } \bar{o} \in F^{\mathcal{I}}.$$

- For all $Poss(\boldsymbol{\alpha}) \in \mathcal{P}_F$ with $\boldsymbol{\alpha} \in \mathcal{N}_A$ it holds that

$$\text{for all } \sigma \in \mathcal{N}_A^* : \;\; w_{\mathcal{I}}^{\Sigma}[Poss(\boldsymbol{\alpha}), \sigma] = 1 \text{ iff } w_{\mathcal{I}}^{\Sigma}, \sigma \models_{\mathcal{ES}} \vartheta_{\boldsymbol{\alpha}}^{a},$$

where $\Sigma_{\mathsf{pre}} = \forall a.\Box(Poss(a) \equiv \vartheta)$.

- For all $SF(\boldsymbol{\alpha}) \in \mathcal{P}_F$ with $\boldsymbol{\alpha} \in \mathcal{N}_A$ it holds that

$$\text{for all } \sigma \in \mathcal{N}_A^* : \;\; w_{\mathcal{I}}^{\Sigma}[SF(\boldsymbol{\alpha}), \sigma] = 1 \text{ iff } w_{\mathcal{I}}^{\Sigma}, \sigma \models_{\mathcal{ES}} \vartheta_{\boldsymbol{\alpha}}^{a},$$

where $\Sigma_{\mathsf{sense}} = \forall a.\Box(SF(a) \equiv \vartheta)$.

▲

The following lemma is an easy consequence.

Lemma 6.31. *Let Σ and $\mathcal{I}$ be as in Definition 6.30.*

1. *A world $w_{\mathcal{I}}^{\Sigma}$ satisfying the conditions in Def. 6.30 exists and is uniquely determined.*
2. *It holds that $w_{\mathcal{I}}^{\Sigma}, \langle\rangle \models_{\mathcal{ES}} \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$.*
3. *Let ϕ be an objective fluent formula over $\mathcal{F}$ with free variables $\bar{x}$, and $\bar{o}$ a tuple of object standard names of the same length as $\vec{x}$. It holds that $\mathcal{I} \models \phi_{\bar{o}}^{\bar{x}}$ iff $w_{\mathcal{I}}^{\Sigma}, \langle\rangle \models_{\mathcal{ES}} \phi_{\bar{o}}^{\bar{x}}$.*

We also need a corresponding construction in the other direction. The truth assignments of a world to primitive formulas given a sequence of actions uniquely determines an interpretation.

Definition 6.32. Let w be an $\mathcal{ES}$ world and $\sigma \in \mathcal{N}_A^*$. We define an FO interpretation $\mathcal{I}_w^{\sigma}$ as follows:

- $\Delta_{\mathcal{I}_w^{\sigma}} := \mathcal{N}_O$;
- $F^{\mathcal{I}_w^{\sigma}} := \{\bar{o} \mid w[F(\bar{o}), \sigma] = 1\}$ for all $F \in \mathsf{N_F}$ and
- $o^{\mathcal{I}_w^{\sigma}} := o$ for all $o \in \mathcal{N}_O$.

▲

By construction the following lemma is a direct consequence.

Lemma 6.33. *Let w, σ and $\mathcal{I}_w^{\sigma}$ be as above. Let ϕ be a fluent formula over $\mathcal{F}$ with free variables $\bar{x}$, and $\bar{o}$ a tuple of object standard names of the same length. It holds that $\mathcal{I}_w^{\sigma} \models \phi_{\bar{o}}^{\bar{x}}$ iff $w, \sigma \models_{\mathcal{ES}} \phi_{\bar{o}}^{\bar{x}}$.*

Now we are ready to define the epistemic FO-DS induced by a BAT.

Definition 6.34. Let $\Sigma = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$ be a BAT over a finite set of fluents $\mathcal{F}$. The *FO-DS induced by* Σ, denoted by $\mathfrak{D}_{\mathsf{K}}(\Sigma) = (\mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}}, \sim_{\mathsf{s}})$, is defined as follows:

- $\mathbb{I}_{\mathsf{ini}} = \{\mathcal{I} \mid w^{\Sigma}_{\mathcal{I}}, \langle\rangle \models_{\mathcal{ES}} \Sigma_0\}$.
- Act consists of all action terms from $\mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$ mentioned in $\Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$.
- The effect functions $\mathcal{E} = \langle \mathsf{add}, \mathsf{del}\rangle$ are given as follows: For each $\mathcal{I} \in \mathbb{I}$, $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ and $F \in \mathcal{F}$ we define

$$\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F) := \left\{\bar{o} \mid w^{\Sigma}_{\mathcal{I}}, \langle\rangle \models_{\mathcal{ES}} \left(\gamma^{+}_{F}\right)^{a\ \bar{x}}_{\boldsymbol{\alpha}\ \bar{o}}\ \right\} \text{ and}$$
$$\mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F) := \left\{\bar{o} \mid w^{\Sigma}_{\mathcal{I}}, \langle\rangle \models_{\mathcal{ES}} \left(\gamma^{-}_{F}\right)^{a\ \bar{x}}_{\boldsymbol{\alpha}\ \bar{o}}\ \right\},$$

  where γ^{+}_{F} and γ^{-}_{F} are the positive and negative effect conditions in the SSA of F in Σ_{post}.
- For each $\mathcal{I} \in \mathbb{I}$ and $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ we have:

$$\mathcal{I} \succ_{\mathsf{poss}} \boldsymbol{\alpha} \text{ iff } w^{\Sigma}_{\mathcal{I}}[\mathit{Poss}(\boldsymbol{\alpha}), \langle\rangle] = 1.$$

- The sensing compatibility relation satisfies

$$\sim_{\mathsf{s}} = \left\{(\mathcal{I}, \boldsymbol{\alpha}, \mathcal{J}) \in \mathbb{I} \times \mathsf{ground}(\mathsf{Act}) \times \mathbb{I} \;\middle|\; w^{\Sigma}_{\mathcal{I}}[\mathit{SF}(\boldsymbol{\alpha}), \langle\rangle] = w^{\Sigma}_{\mathcal{J}}[\mathit{SF}(\boldsymbol{\alpha}), \langle\rangle]\right\}.$$

With $\mathfrak{D}(\Sigma)$ we denote the induced FO-DS $(\mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}})$ without the sensing compatibility relation. ▲

The defined $\mathfrak{D}_{\mathsf{K}}(\Sigma)$ is a well-defined epistemic FO-DS. Since Σ_0 only consists of objective fluent sentences, we can view Σ_0 as a first-order KB. All action terms occurring in $\Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$ different from the action variable a are elements of $\mathsf{Term}(\mathsf{N_A}, \mathsf{N_V}, \mathsf{N_O})$. For the set of action terms Act we have $\mathsf{ground}(\mathsf{Act}) \subseteq \mathcal{N}_A$. For each n-ary fluent $F(\bar{x}) \in \mathcal{F}$ and ground action $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ the instantiated positive and negative effect conditions $\left(\gamma^{+}_{F}\right)^{a}_{\boldsymbol{\alpha}}$ and $\left(\gamma^{-}_{F}\right)^{a}_{\boldsymbol{\alpha}}$ obtained from the SSA of F have only free variables among $\bar{x}$. Hence, the sets $\mathsf{add}(\mathcal{I}, \boldsymbol{\alpha}, F)$ and $\mathsf{del}(\mathcal{I}, \boldsymbol{\alpha}, F)$ are well-defined sets of n-tuples from $\mathsf{N_O}$.

Next, we prove the correctness of the definition.

Lemma 6.35. *Let* $\Sigma = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$ *be a BAT over* $\mathcal{F}$ *and*

$$\mathfrak{D}_{\mathsf{K}}(\Sigma) = (\mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}}, \sim_{\mathsf{s}})$$

the induced epistemic FO-DS with $\mathfrak{D}(\Sigma) = (\mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}})$.

1. *For all* $\mathcal{I} \in \mathbb{I}_{\mathsf{ini}}$ *it holds that* $w^{\Sigma}_{\mathcal{I}}, \langle\rangle \models_{\mathcal{ES}} \Sigma$.
2. *For all* $\mathcal{ES}$ *worlds with* $w, \langle\rangle \models_{\mathcal{ES}} \Sigma$ *it holds that* $\mathcal{I}^{\langle\rangle}_{w} \in \mathbb{I}_{\mathsf{ini}}$.
3. *Let* $\mathcal{I} \in \mathbb{I}_{\mathsf{ini}}$, $\sigma \in \mathsf{ground}(\mathsf{Act})^{*}$ *and* $\mathcal{J} \in \mathbb{I}$ *with* $\mathcal{I} \Rightarrow^{\sigma}_{\mathfrak{D}(\Sigma)} \mathcal{J}$.
   a) *For all* $\rho \in \mathcal{P}_F$ *it holds that* $w^{\Sigma}_{\mathcal{I}}[\rho, \sigma] = w^{\Sigma}_{\mathcal{J}}[\rho, \langle\rangle]$.

b) For all $\boldsymbol{\alpha} \in \mathsf{ground(Act)}$ *it holds that* $w^{\Sigma}_{\mathcal{I}}[Poss(\boldsymbol{\alpha}), \sigma]$ *iff* $\mathcal{J} \succ_{\mathsf{poss}} \boldsymbol{\alpha}$.

4. *Let w be a world with* $w, \langle\rangle \models_{\mathcal{ES}} \Sigma$ *and* $\sigma \in \mathsf{ground(Act)}^*$.
   a) It holds that $\mathcal{I}^{\langle\rangle}_{w} \Rightarrow^{\sigma}_{\mathfrak{D}(\Sigma)} \mathcal{I}^{\sigma}_{w}$.
   b) For all $\boldsymbol{\alpha} \in \mathsf{ground(Act)}$ *it holds that* $w[Poss(\boldsymbol{\alpha}), \sigma] = 1$ *iff* $\mathcal{I}^{\sigma}_{w} \succ_{\mathsf{poss}} \boldsymbol{\alpha}$.

5. *Let w and ω be two worlds with*

$$w, \langle\rangle \models_{\mathcal{ES}} \Sigma \text{ and } \omega, \langle\rangle \models_{\mathcal{ES}} \Sigma,$$

   $\sigma = \boldsymbol{\alpha}_0 \cdots \boldsymbol{\alpha}_n \in \mathsf{ground(Act)}^*$ *an action sequence for some* $n \geq 0$ *and* $\mathcal{I}_0, \ldots, \mathcal{I}_{n+1}$ *and* $\mathcal{J}_0, \ldots, \mathcal{J}_{n+1}$ *sequences of interpretations with* $\mathcal{I}_0 = \mathcal{I}^{\langle\rangle}_{w}$, $\mathcal{J}_0 = \mathcal{I}^{\langle\rangle}_{\omega}$ *and* $\mathcal{I}_i \Rightarrow^{\boldsymbol{\alpha}_i}_{\mathfrak{D}(\Sigma)} \mathcal{I}_{i+1}$ *and* $\mathcal{J}_i \Rightarrow^{\boldsymbol{\alpha}_i}_{\mathfrak{D}(\Sigma)} \mathcal{J}_{i+1}$ *for all* $i = 0, \ldots, n$.

   It holds that $w \simeq_{\sigma} \omega$ *iff* ($\mathcal{I}_i \sim^{\boldsymbol{\alpha}_i}_{\mathsf{s}} \mathcal{J}_i$ *for all* $i = 0, \ldots, n$).

Proof. The first two claims follow directly from the definition of $\mathfrak{D}_{\mathsf{K}}(\Sigma)$. Claim 3, 4 and 5 can be proven by induction on the length of σ. We omit further details. □

As a consequence of this lemma it follows that the semantics of a BAT can be properly characterized using an epistemic FO-DS. Furthermore, it follows that ground actions defined in a BAT $\Sigma = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}}$ (without sensing) are FO-definable in $\mathfrak{D}(\Sigma)$.

Lemma 6.36. *Let* $\Sigma = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}}$ *be a BAT (without sensing) over* $\mathcal{F}$,

$$\mathfrak{D}(\Sigma) = (\mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}})$$

the induced FO-DS and $\boldsymbol{A} \subseteq \mathsf{ground(Act)}$ *a finite set of ground actions. It holds that the actions* $\boldsymbol{A}$ *are FO-definable in* $\mathfrak{D}(\Sigma)$.

Proof. Let $\boldsymbol{\alpha} \in \boldsymbol{A}$ and $F \in \mathcal{F}$. Let γ^{+}_{F} and γ^{-}_{F} be the positive and negative effect condition, respectively, in the SSA for F in Σ_{post}. In the effect conditions we substitute the action variable a by the ground term $\boldsymbol{\alpha}$. It can be shown that there are FO formulas that are equivalent to $\left(\gamma^{+}_{F}\right)^{a}_{\boldsymbol{\alpha}}$ and $\left(\gamma^{-}_{F}\right)^{a}_{\boldsymbol{\alpha}}$ and provide context-free definitions of the add-set and delete-set, respectively.
The context for the preconditions are obtained by instantiating the action variable in the precondition axiom in Σ_{pre}. □

Thus, a BAT can be viewed as an FO-admissible representation for a set of ground actions.
Lemma 6.35 also allows us to establish the relationship of the transition relation on epistemic interpretations "$\Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathsf{K}}}$" and the progression of epistemic states in $\mathcal{ES}$ through an action sequence σ.

Lemma 6.37. *Let* Σ *be a BAT over* $\mathcal{F} \subset \mathsf{N_F}$, $\mathfrak{D}_{\mathsf{K}}(\Sigma) = (\mathbb{I}_{\mathsf{ini}}, \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}}, \sim_{\mathsf{s}})$ *the induced epistemic FO-DS, and* $e \subseteq W$ *an epistemic state with* $w \in e$ *such that*

$$e, w \models_{\mathcal{ES}} \Sigma \wedge OKnow(\Sigma).$$

Furthermore, let $\sigma \in \mathsf{ground}(\mathsf{Act})^*$ be an action sequence, $\mathcal{I} \in \mathbb{I}_{\mathsf{ini}}$ an interpretation with $\mathcal{I} = \mathcal{I}_w^{\langle\rangle}$ and $(\mathcal{J}, \mathcal{M})$ the epistemic interpretation with $(\mathcal{I}, \mathcal{W}) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{J}, \mathcal{M})$, where $\mathcal{W} = \mathbb{I}_{\mathsf{ini}}$. It holds that

$$\mathcal{J} = \mathcal{I}_w^{\sigma} \text{ and } \mathcal{M} = \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e_\sigma^w\}.$$

Proof. Note that a BAT Σ is objective. Let $e \subseteq W$ and $w \in e$ with $e, w \models_{\mathcal{ES}} \Sigma \wedge \mathit{OKnow}(\Sigma)$. It is implied that $e = \{w \in W \mid w, \langle\rangle \models_{\mathcal{ES}} \Sigma\}$. Let σ be a ground action sequence and $(\mathcal{J}, \mathcal{M})$ the epistemic interpretation satisfying $(\mathcal{I}_w^{\langle\rangle}, \mathbb{I}_{\mathsf{ini}}) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{J}, \mathcal{M})$. It follows that $\mathcal{J} = \mathcal{I}_w^{\sigma}$ from Lemma 6.35.4. We prove by induction on the length of σ that $\mathcal{M} = \{\mathcal{I}_\omega^{\sigma} \mid \omega \in e_\sigma^w\}$.

First, assume $\sigma = \langle\rangle$. It follows that $e_\sigma^w = e$. We have to show that $\mathbb{I}_{\mathsf{ini}} = \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e\}$. Let $\mathcal{Y} \in \mathbb{I}_{\mathsf{ini}}$ and let $\omega' = w_{\mathcal{Y}}^{\Sigma}$. It follows that $\mathcal{Y} = \mathcal{I}_{\omega'}^{\langle\rangle}$. Lemma 6.35.1 implies $\omega', \langle\rangle \models_{\mathcal{ES}} \Sigma$. Therefore, $\omega' \in e$ and $\mathcal{Y} \in \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e\}$. Consequently, $\mathbb{I}_{\mathsf{ini}} \subseteq \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e\}$. The other direction $\{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e\} \subseteq \mathbb{I}_{\mathsf{ini}}$ follows from $e = \{w \in W \mid w, \langle\rangle \models_{\mathcal{ES}} \Sigma\}$ and Lemma 6.35.2.

Next, assume $\sigma = \theta \cdot \boldsymbol{\alpha}$. Let $(\mathcal{Y}, \mathcal{V})$ be such that

$$(\mathcal{I}_w^{\langle\rangle}, \mathbb{I}_{\mathsf{ini}}) \Longrightarrow^{\theta}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{Y}, \mathcal{V}) \Longrightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{J}, \mathcal{M}).$$

By induction we have $\mathcal{V} = \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e_\theta^w\}$. We have $\mathcal{J}'' \in \mathcal{M}$

iff there exists $\mathcal{J}' \in \mathcal{V}$ with $\mathcal{Y} \sim_{\mathsf{s}}^{\boldsymbol{\alpha}} \mathcal{J}'$ and $\mathcal{J}' \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma)} \mathcal{J}''$

iff there exists $\upsilon \in e_\theta^w$ with $\mathcal{I}_w^{\theta} \sim_{\mathsf{s}}^{\boldsymbol{\alpha}} \mathcal{I}_\upsilon^{\langle\rangle}$ and $\mathcal{I}_\upsilon^{\langle\rangle} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma)} \mathcal{J}''$ (with $\mathcal{V} = \{\mathcal{I}_\omega^{\theta} \mid \omega \in e_\theta^w\}$)

iff there exists $\mu \in e$ with $w \simeq_\theta \mu$ and $\mathcal{I}_w^{\theta} \sim_{\mathsf{s}}^{\boldsymbol{\alpha}} \mathcal{I}_\mu^{\theta}$ and $\mathcal{I}_\mu^{\theta} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma)} \mathcal{J}''$ (by definition of e_θ^w)

iff there exists $\mu \in e$ with $w \simeq_{\theta\cdot\boldsymbol{\alpha}} \mu$ and $\mathcal{I}_\mu^{\theta} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma)} \mathcal{J}''$ and $\mathcal{J}'' = \mathcal{I}_\mu^{\theta\cdot\boldsymbol{\alpha}}$ (by Lemma 6.35)

iff there exists $\mu \in e$ and $\mu_{\theta\cdot\boldsymbol{\alpha}} \in e_{\theta\cdot\boldsymbol{\alpha}}^w$ and $\mathcal{I}_\mu^{\theta} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}(\Sigma)} \mathcal{J}''$ and $\mathcal{J}'' = \mathcal{I}_{\mu_{\theta\cdot\boldsymbol{\alpha}}}^{\langle\rangle}$ (with $\mathcal{I}_\mu^{\theta\cdot\boldsymbol{\alpha}} = \mathcal{I}_{\mu_{\theta\cdot\boldsymbol{\alpha}}}^{\langle\rangle}$)

iff $\mathcal{J}'' \in \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e_{\theta\cdot\boldsymbol{\alpha}}^w\}$.

□

To characterize the epistemic projection problem in an epistemic $\mathcal{ALCO}$-action theory Σ_{s} in $\mathcal{ES}$ we need to translate Σ_{s} to a BAT. We use the translation of DL syntax into FO syntax (see Definition 2.25). For the sake of simplicity we only consider action theories for a finite set of ground actions. This is sufficient because the projection problem only deals with ground actions.

Definition 6.38. Let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory, where $\boldsymbol{A}$ is a finite set of ground action terms. A corresponding BAT $\widehat{\Sigma} = \Sigma_0 \cup \Sigma_{\mathsf{pre}} \cup \Sigma_{\mathsf{post}} \cup \Sigma_{\mathsf{sense}}$ is defined as follows:

$$\Sigma_0 := \{\mathsf{tr}(\varphi) \mid \varphi \text{ is an axiom in } \mathcal{K}\}.$$

The precondition axiom in Σ_{pre} is given by

$$\forall a.\, \Box\, Poss(a) \equiv \Big(\bigvee_{\boldsymbol{\alpha} \in \boldsymbol{A}} a = \boldsymbol{\alpha} \wedge \Big(\bigwedge_{\psi \in \mathsf{pre}(\boldsymbol{\alpha})} \mathsf{tr}(\psi) \Big) \Big).$$

For each relevant concept name and role name F in Σ_{s} there is an SSA of the form

$$\forall a. \forall \bar{x}. \Box(\ ([a]F(\bar{x})) \ \equiv \ (\gamma_F^+ \vee F(\bar{x}) \wedge \neg \gamma_F^-) \ )$$

in Σ_{post} with

$$\gamma_F^+ := \bigvee_{\substack{\boldsymbol{\alpha} \in \boldsymbol{A}, \\ \psi \rhd \langle F, \{\bar{o}\} \rangle^+ \in\, \mathsf{eff}(\boldsymbol{\alpha})}} a = \boldsymbol{\alpha} \wedge \bar{x} = \bar{o} \wedge \mathsf{tr}(\psi) \text{ and}$$

$$\gamma_F^- := \bigvee_{\substack{\boldsymbol{\alpha} \in \boldsymbol{A}, \\ \psi \rhd \langle F, \{\bar{o}\} \rangle^- \in\, \mathsf{eff}(\boldsymbol{\alpha})}} a = \boldsymbol{\alpha} \wedge \bar{x} = \bar{o} \wedge \mathsf{tr}(\psi).$$

The axiom for the sensing results is given by

$$\forall a.\, \Box\, SF(a) \equiv \Big(\bigvee_{\boldsymbol{\alpha} \in \boldsymbol{A}} a = \boldsymbol{\alpha} \wedge \mathsf{tr}(\mathsf{sense}(\boldsymbol{\alpha})) \Big).$$

▲

Using the FO-DS semantics of BATs it is straightforward to prove that the action theory and the translated BAT are equivalent.

Lemma 6.39. *Let Σ_{s} be an epistemic $\mathcal{ALCO}$-action theory and $\widehat{\Sigma}$ the corresponding BAT as defined above. It holds that $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = \mathfrak{D}_{\mathbf{K}}(\widehat{\Sigma})$.*

In order to translate the projection query formulated in $\mathcal{ALCOK}$ to an $\mathcal{ES}$ fluent formula the translation function from Definition 2.25 is extended to $\mathcal{ALCOK}$ as follows: let $\mathbf{K}C$ be an $\mathcal{ALCOK}$-concept we define

$$\mathsf{tr}^x(\mathbf{K}C) := Know(\mathsf{tr}^x(C)) \text{ and } \mathsf{tr}^y(\mathbf{K}C) := Know(\mathsf{tr}^y(C)).$$

The definitions for the other concept constructors are as in Definition 2.25. Similarly, for an epistemic role $\mathbf{K}R$ we define $\mathsf{tr}^{x,y}(\mathbf{K}R) := Know(\mathsf{tr}^{x,y}(R))$. For an epistemic Boolean $\mathcal{ALCOK}$-KB $\mathbf{K}\psi$ we define $\mathsf{tr}(\mathbf{K}\psi) := Know(\mathsf{tr}(\psi))$.

Lemma 6.40. *Let ψ be a Boolean $\mathcal{ALCOK}$-KB, $(\mathcal{I}, \mathcal{W})$ an epistemic interpretation, $e \subseteq W$ a set of $\mathcal{ES}$ worlds and $w \in e$ such that $\mathcal{I} = \mathcal{I}_w^{\langle\rangle}$ and $\mathcal{W} = \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e\}$. It holds that*

$$(\mathcal{I}, \mathcal{W}) \mid\!\models \psi \text{ iff } e, w \models_{\mathcal{ES}} \mathsf{tr}(\psi).$$

Proof. By induction it is straightforward to prove that the FO translation $\mathsf{tr}(\psi)$ of a Boolean $\mathcal{ALCOK}$-KB ψ is correct. We omit the proof. □

Now we are ready to prove the $\mathcal{ES}$-characterization of the epistemic projection problem.

Theorem 6.41. *Let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory, where $\boldsymbol{A}$ is a finite set of ground action terms, $\widehat{\Sigma}$ the equivalent BAT (according to Def. 6.38), $\sigma \in \boldsymbol{A}^*$ a ground action sequence and ψ a Boolean $\mathcal{ALCOK}$-KB. It holds that ψ is valid after doing σ in Σ_{s} iff*

$$\widehat{\Sigma} \wedge \mathit{OKnow}(\widehat{\Sigma}) \models_{\mathcal{ES}} [\sigma]\mathsf{tr}(\psi).$$

Proof. Let $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \succ_{\mathsf{poss}}, \sim_{\mathsf{s}})$. We have $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = \mathfrak{D}_{\mathbf{K}}(\widehat{\Sigma})$ by Lemma 6.39.

"$\Rightarrow$": Assume that ψ is valid after doing σ in Σ_{s}. Let $e \subseteq W$ and $w \in e$ such that

$$e, w \models_{\mathcal{ES}} \widehat{\Sigma} \wedge \mathit{OKnow}(\widehat{\Sigma}).$$

We have to show that $e, w, \sigma \models_{\mathcal{ES}} \mathsf{tr}(\psi)$. From Lemma 6.35 and $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = \mathfrak{D}_{\mathbf{K}}(\widehat{\Sigma})$ it follows that $\mathcal{I}_w^{\langle\rangle} \in \mathcal{M}(\mathcal{K})$, where $\mathcal{M}(\mathcal{K})$ is the initial state space of $\mathfrak{D}_{\mathbf{K}}(\widehat{\Sigma})$. Let $(\mathcal{J}, \mathcal{M})$ be the epistemic interpretation with $(\mathcal{I}_w^{\langle\rangle}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{J}, \mathcal{M})$. Lemma 6.37 implies that

$$\mathcal{J} = \mathcal{I}_w^{\sigma} \text{ and } \mathcal{M} = \{\mathcal{I}_\omega^{\langle\rangle} \mid \omega \in e_\sigma^w\}$$

It follows that $\mathcal{J} = \mathcal{I}_{w_\sigma}^{\langle\rangle}$. Lemma 6.40 implies

$$e_\sigma^w, w_\sigma, \langle\rangle \models_{\mathcal{ES}} \mathsf{tr}(\psi)$$

since we have $(\mathcal{J}, \mathcal{M}) \mid\!\models \psi$ by assumption. Now,

$$e_\sigma^w, w_\sigma, \langle\rangle \models_{\mathcal{ES}} \mathsf{tr}(\psi) \text{ implies } e, w, \sigma \models_{\mathcal{ES}} \mathsf{tr}(\psi).$$

"$\Leftarrow$": Assume $\widehat{\Sigma} \wedge \mathit{OKnow}(\widehat{\Sigma}) \models_{\mathcal{ES}} [\sigma]\mathsf{tr}(\psi)$. Let $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ and $(\mathcal{J}, \mathcal{M})$ the epistemic interpretation that satisfies $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{J}, \mathcal{M})$. We have to show that $(\mathcal{J}, \mathcal{M}) \mid\!\models \psi$. Let

$$w = w_{\mathcal{I}}^{\widehat{\Sigma}} \text{ and } e = \{w_{\mathcal{Y}}^{\widehat{\Sigma}} \mid \mathcal{Y} \in \mathcal{M}(\mathcal{K})\}$$

Lemma 6.35.1 and 6.35.2 imply $e = \{w \in W \mid w, \langle\rangle \models_{\mathcal{ES}} \widehat{\Sigma}\}$. Consequently,

$$e, w \models_{\mathcal{ES}} \widehat{\Sigma} \wedge \mathit{OKnow}(\widehat{\Sigma}).$$

By assumption we have

$$e, w \models_{\mathcal{ES}} [\sigma]\mathsf{tr}(\psi).$$

This implies

$$e_\sigma^w, w_\sigma, \langle\rangle \models_{\mathcal{ES}} \mathsf{tr}(\psi).$$

With Lemma 6.37 and 6.40 it follows that $(\mathcal{J}, \mathcal{M}) \mid\!\models \psi$. □

6.4 Deciding the Epistemic Projection Problem

In Theorem 6.41 we have characterized the epistemic projection problem in our DL-based language as a standard entailment problem in the epistemic Situation Calculus. The Representation Theorem for $\mathcal{ES}$ [LL04] provides us with a method for reducing projection to standard (non-modal) first-order reasoning by eliminating the action and knowledge modalities in the projection query. To deal with the action modality regression is used to obtain a sentence that refers only to the initial situation. Given the initial KB subformulas of the form $Know(\phi)$ are then replaced by objective formulas ϕ' that capture the known instances of ϕ w.r.t. the initial KB. To obtain a decision procedure for the projection problem we show that a similar reduction can be done within $\mathcal{ALCO}$. We combine the reduction approach used in [Baa+05a] for the non-epistemic projection problem and a method for rewriting subjective concepts and roles to objective ones in the projection query resembling the Representation Theorem [Lev84; LL01] in presence of only-knowing. We show that the epistemic projection problem is ExpTime-complete.

First, we consider some basic properties of knowledge states that evolve from the initial epistemic model by executing a sequence of ground actions.

In the following $\mathsf{Obj}(\Sigma_\mathsf{s})$ denotes the finite set of all object names that are mentioned in the epistemic $\mathcal{ALCO}$-action theory Σ_s under consideration.

The elements of $\mathsf{Obj}(\Sigma_\mathsf{s})$ are called *named elements* and the ones in $\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ are called *unnamed elements*. The next lemma basically says that in a knowledge state reached by executing a sequence of ground actions the unnamed elements are indistinguishable.

Lemma 6.42. *Let $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory and let $\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}, \sim_\mathsf{s})$ be the induced epistemic FO-DS.*

Furthermore, let $\sigma \in \boldsymbol{A}^*$ and $(\mathcal{J}, \mathcal{M})$ the epistemic interpretation with

$$(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})} (\mathcal{J}, \mathcal{M})$$

for some $\mathcal{I} \in \mathcal{M}(\mathcal{K})$.

1. *Let $\mathbf{K}D$ be an $\mathcal{ALCOK}$-concept where D is objective. It holds that*

$$\big(\mathbf{K}D\big)^{\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset \textit{ implies } (\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})) \subseteq \big(\mathbf{K}D\big)^{\mathcal{M}}.$$

2. *Let $\mathbf{K}P$ be an epistemic role with $P \in \mathsf{N_R}$. It holds that if $(d,e) \in (\mathbf{K}P)^{\mathcal{M}}$ for some $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ and $e \in \mathsf{N_O}$, then $(c,e) \in (\mathbf{K}P)^{\mathcal{M}}$ for all $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$; and if $(d,e) \in (\mathbf{K}P)^{\mathcal{M}}$ for some $e \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ and $d \in \mathsf{N_O}$, then $(d,c) \in (\mathbf{K}P)^{\mathcal{M}}$ for all $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$.*

We first present an outline of the proof. Donini et. al [Don+98] showed that the interpretations contained in the epistemic model $\mathcal{M}(\mathcal{K})$ of an $\mathcal{ALC}$-KB $\mathcal{K}$ are closed under renaming of unnamed elements. Unnamed elements are indistinguishable. We use this observation for the proof of Lemma 6.42 as follows. Assume to the contrary that there exist two unnamed elements $d, e \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that $d \in \big(\mathbf{K}D\big)^{\mathcal{M}}$ and $e \notin \big(\mathbf{K}D\big)^{\mathcal{M}}$. We show that there exists an interpretation $\mathcal{Y} \in \mathcal{M}$ such that $d \in D^{\mathcal{Y}}$ but $e \notin D^{\mathcal{Y}}$. To obtain the contradiction we construct an interpretation $\mathcal{Y}'$ also contained in $\mathcal{M}$ with $d \notin D^{\mathcal{Y}}$ by just "swapping" the two unnamed elements d and e in $\mathcal{Y}$. For the proof some auxiliary lemmas are needed.

We show that updating a renamed interpretation yields an isomorphic interpretation if names of named elements are fixed.

Lemma 6.43. *Let $\mathcal{I}$ be an interpretation, $\iota : \mathsf{N_O} \to \mathsf{N_O}$ a bijection with $\iota(o) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$, L a set of local effects, C an $\mathcal{ALCO}$-concept, and ψ a Boolean $\mathcal{ALCO}$-KB, where all object names mentioned in L, C and ψ are from $\mathsf{Obj}(\Sigma_\mathsf{s})$. Furthermore, let $\mathcal{J} := \iota(\mathcal{I})$.*

1. *$o \in A^{\mathcal{I}^{\mathsf{L}}}$ iff $\iota(o) \in A^{\mathcal{J}^{\mathsf{L}}}$ for all $o \in \mathsf{N_O}$ and $A \in \mathsf{N_C}$;*
2. *$(o, o') \in P^{\mathcal{I}^{\mathsf{L}}}$ iff $(\iota(o), \iota(o')) \in P^{\mathcal{J}^{\mathsf{L}}}$ for all $o, o' \in \mathsf{N_O}$ and $P \in \mathsf{N_R}$;*
3. *$o \in C^{\mathcal{I}^{\mathsf{L}}}$ iff $\iota(o) \in C^{\mathcal{J}^{\mathsf{L}}}$ for all $o \in \mathsf{N_O}$;*
4. *$\mathcal{I}^{\mathsf{L}} \models \psi$ iff $\mathcal{J}^{\mathsf{L}} \models \psi$.*

Proof.

1. Let $A \in \mathsf{N_C}$ and $d \in \mathsf{N_O}$. We show $d \in A^{\mathcal{I}^{\mathsf{L}}}$ iff $\iota(d) \in A^{\mathcal{J}^{\mathsf{L}}}$. By the definition of renamed interpretations we have $d \in A^{\mathcal{I}}$ iff $\iota(d) \in A^{\mathcal{J}}$.

   Furthermore it holds that $d \in \{o \mid \langle A, \{o\}\rangle^{-} \in \mathsf{L}\}$

   iff there exists $\langle A, \{d\}\rangle^{-} \in \mathsf{L}$,

   iff $\iota(d) = d$ and $\langle A, \{d\}\rangle^{-} \in \mathsf{L}$, by assumption on ι

   iff $\iota(d) \in \{o \mid \langle A, \{o\}\rangle^{-} \in \mathsf{L}\}$.

   We also have $d \in \{o \mid \langle A, \{o\}\rangle^{+} \in \mathsf{L}\}$ iff $\iota(d) \in \{o \mid \langle A, \{o\}\rangle^{+} \in \mathsf{L}\}$. By the definition of interpretation updates it now follows that $d \in A^{\mathcal{I}^{\mathsf{L}}}$ iff $\iota(d) \in A^{\mathcal{J}^{\mathsf{L}}}$.

2. The proof is analogous to the proof of claim 1.

3. The claim is proven by induction on the structure of C.

   $C = A$: for some $A \in \mathsf{N_C}$. See proof of claim 1.

   $C = \{o\}$: for some $o \in \mathsf{Obj}(\Sigma_{\mathsf{s}})$. We get $d \in \{o\}^{\mathcal{I}^{\mathsf{L}}}$ iff $d \in \{o^{\mathcal{I}^{\mathsf{L}}}\}$ iff $d = o$ iff $\iota(d) = o$ iff $\iota(d) \in \{o^{\mathcal{J}^{\mathsf{L}}}\}$ iff $\iota(d) \in \{o\}^{\mathcal{J}^{\mathsf{L}}}$.

   We omit the proof of the remaining cases.

4. The claim directly follows from the other three claims.

□

In particular, we have that renaming unnamed elements does not change the dynamic type of interpretations. It remains to be shown that a relevant context can be chosen such that two interpretations with the same dynamic type w.r.t. that context also given the same sensing result.

Let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory, where $\boldsymbol{A}$ is a finite set of ground action terms. The *set of all relevant local effects* in Σ_{s}, denoted by $\mathsf{Lit}(\Sigma_{\mathsf{s}})$, are given by

$$\mathsf{Lit}(\Sigma_{\mathsf{s}}) := \left\{\langle F, X\rangle^{\pm} \,\middle|\, \psi \rhd \langle F, X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}) \text{ for some } \boldsymbol{\alpha} \in \boldsymbol{A}\right\}. \tag{6.1}$$

The relevant $\mathcal{ALCO}$-context for Σ_{s} is given by

$$\begin{aligned} \mathcal{C} := \{\psi, \neg\psi \mid\ & \psi \rhd \langle F, \{\bar{o}\}\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}) \text{ for some } \boldsymbol{\alpha} \in \boldsymbol{A}, \text{ or} \\ & \psi = \mathsf{sense}(\boldsymbol{\alpha}) \text{ for some } \boldsymbol{\alpha} \in \boldsymbol{A}, \text{ or} \\ & \psi \text{ is an axiom in } \mathcal{K}\}. \end{aligned} \tag{6.2}$$

The preconditions are not relevant for the projection problem and are omitted. Dynamic types w.r.t. $\mathsf{Lit}(\Sigma_\mathsf{s})$ and $\mathcal{C}$ are defined as in Definition 4.7.

Lemma 6.44. *Let $\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_\mathsf{s})$ be the epistemic FO-DS induced by $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$, $\boldsymbol{\alpha}_0\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ a sequence of ground actions,*

$$(\mathcal{I}_0, \mathcal{W}_0) \Longrightarrow^{\boldsymbol{\alpha}_0}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_1, \mathcal{W}_1) \Longrightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} \cdots \Longrightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_{n+1}, \mathcal{W}_{n+1})$$

a sequence of epistemic interpretations with $\mathcal{W}_0 = \mathcal{M}(\mathcal{K})$ and $\mathcal{Y}_0, \mathcal{Y}_1, \ldots, \mathcal{Y}_{n+1}$ a sequence of interpretations such that

- *$\mathcal{Y}_j \in \mathcal{W}_j$ for all $j = 0, \ldots, n+1$ and*
- *$\mathcal{Y}_i \Rightarrow^{\boldsymbol{\alpha}_i}_{\mathfrak{D}} \mathcal{Y}_{i+1}$ for all $i = 0, \ldots, n$.*

Furthermore, let $\mathcal{J}_0 \in \mathcal{M}(\mathcal{K})$ be an interpretation with $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J}_0) = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{Y}_0)$.

It holds that for each $j \in \{0, \ldots, n+1\}$ there exists a set of local effects $\mathsf{L}_j \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ such that

$$\mathcal{J}_0{}^{\mathsf{L}_j} \in \mathcal{W}_j \text{ and } \mathcal{Y}_j = \mathcal{Y}_0{}^{\mathsf{L}_j}.$$

Proof. The proof is by induction on the length of the action sequence.
base case:
We assume $n = 0$. For $j = 0$ the claim trivially holds for $\mathsf{L}_0 = \emptyset$. Let $j = 1$. By definition it holds that $\mathcal{Y}_1 = \mathcal{Y}_0{}^{\mathsf{L}_0}$ with $\mathsf{L}_0 = \{\langle F, X\rangle^{\pm} \mid \psi \rhd \langle F, X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}_0), \mathcal{Y}_0 \models \psi\}$. $\mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{Y}_0) = \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J}_0)$ implies $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{Y}_0) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J}_0)$. Hence, $\mathcal{Y}_0 \sim^{\alpha_0}_\mathsf{s} \mathcal{J}_0$ and $\mathcal{J}_0 \Rightarrow^{\boldsymbol{\alpha}_0}_{\mathfrak{D}} \mathcal{J}_0{}^{\mathsf{L}_0}$. Consequently, $\mathcal{J}_0{}^{\mathsf{L}_0} \in \mathcal{W}_1$.
induction step:
We assume that there exists a set of local effects $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ such that $\mathcal{Y}_n = \mathcal{Y}_0{}^{\mathsf{L}}$ and $\mathcal{J}_0{}^{\mathsf{L}} \in \mathcal{W}_n$. Due to the assumption we have $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{Y}_n) = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{J}_0{}^{\mathsf{L}})$. It follows that $\mathcal{Y}_n \sim^{\boldsymbol{\alpha}_n}_\mathsf{s} \mathcal{J}_0{}^{\mathsf{L}}$. Let $\mathsf{L}' := \{\langle F, X\rangle^{\pm} \mid \psi \rhd \langle F, X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}_n), \mathcal{Y}_n \models \psi\}$. We have $\mathcal{Y}_{n+1} = \mathcal{Y}_n{}^{\mathsf{L}'}$ and

$$\mathcal{Y}_0{}^{\mathsf{L} \setminus \neg \mathsf{L}' \cup \mathsf{L}'} \in \mathcal{W}_{n+1} \text{ and } \mathcal{J}_0{}^{\mathsf{L} \setminus \neg \mathsf{L}' \cup \mathsf{L}'} \in \mathcal{W}_{n+1}.$$

□

Finally we are ready to prove Lemma 6.42.

Proof of Lemma 6.42. Let $\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}, \sim_\mathsf{s})$ be the epistemic FO-DS, $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ and

$$(\mathcal{I}_0, \mathcal{W}_0) \Longrightarrow^{\boldsymbol{\alpha}_0}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_1, \mathcal{W}_1) \Longrightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} \cdots \Longrightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_{n+1}, \mathcal{W}_{n+1})$$

the sequence of epistemic interpretations obtained by executing $\sigma = \boldsymbol{\alpha}_0 \cdots \boldsymbol{\alpha}_n$ in $(\mathcal{I}_0, \mathcal{W}_0)$ with $\mathcal{W}_0 = \mathcal{M}(\mathcal{K})$ and $\mathcal{W}_{n+1} = \mathcal{M}$.

1. Assume $(\mathbf{K}D)^{\mathcal{M}} \cap (\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})) \neq \emptyset$ for an $\mathcal{ALCO}$-concept D. We want to show that

$$(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})) \subseteq (\mathbf{K}D)^{\mathcal{M}}.$$

Assume to the contrary that there is an anonymous elements $e \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_s)$ such that $e \notin (\mathbf{K}D)^{\mathcal{M}}$. Consequently, there exists a sequence of interpretations $\mathcal{Y}_0, \mathcal{Y}_1, \ldots, \mathcal{Y}_{n+1}$ such that $\mathcal{Y}_i \in \mathcal{W}_i$ for all $i = 0, \ldots, n+1$, $\mathcal{Y}_j \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_j} \mathcal{Y}_{j+1}$ for all $j = 0, \ldots, n$ and $e \notin D^{\mathcal{Y}_{n+1}}$. We choose an anonymous element $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_s)$ with $d \in (\mathbf{K}D)^{\mathcal{M}}$. By assumption such an element exists. Furthermore, we choose a bijection $\iota : \mathsf{N_O} \to \mathsf{N_O}$ such that $\iota(o) = o$ for all $o \in \mathsf{Obj}(\Sigma_s)$ and $\iota(e) = d$ and $\iota(d) = e$. Let $\mathcal{J}_0 := \iota(\mathcal{Y}_0)$. Using Lemma 6.43.4 it follows that $\text{d-type}_C^{\text{loc}}(\mathcal{J}_0) = \text{d-type}_C^{\text{loc}}(\mathcal{Y}_0)$ as required for Lemma 6.44. Therefore, Lemma 6.44 implies that there exists a set of literals $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_s)$ such that ${\mathcal{J}_0}^{\mathsf{L}} \in \mathcal{M}$ and $\mathcal{Y}_{n+1} = {\mathcal{Y}_0}^{\mathsf{L}}$. With Lemma 6.43.3, $e \notin D^{\mathcal{Y}_{n+1}}$ implies $\iota(e) \notin D^{{\mathcal{J}_0}^{\mathsf{L}}}$. Since $\iota(e) = d$ and ${\mathcal{J}_0}^{\mathsf{L}} \in \mathcal{M}$ it follows that $d \notin (\mathbf{K}D)^{\mathcal{M}}$ which is a contradiction.

2. Let $(d, e) \in \mathsf{N_O} \times \mathsf{N_O}$ and $(d, e) \in (\mathbf{K}P)^{\mathcal{M}}$ for some $P \in \mathsf{N_R}$. Assume $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_s)$. We want to show that

$$(c, e) \in (\mathbf{K}P)^{\mathcal{M}} \text{ for all } c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_s)$$

is implied. Assume to the contrary that there is an element $a \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_s)$ such that $(a, e) \notin (\mathbf{K}P)^{\mathcal{M}}$. As in the proof of 1 we can choose a sequence of interpretations $\mathcal{Y}_0, \mathcal{Y}_1, \ldots, \mathcal{Y}_{n+1}$ with $\mathcal{Y}_i \in \mathcal{W}_i$ for all $i = 0, \ldots, n+1$, $\mathcal{Y}_j \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_j} \mathcal{Y}_{j+1}$ for all $j = 0, \ldots, n$ and $(a, e) \notin P^{\mathcal{Y}_{n+1}}$. Renaming $\mathcal{Y}_0$ such that d and a are swapped yields an interpretation that is indistinguishable from $\mathcal{Y}_0$ leads to a contradiction with the assumption $(d, e) \in (\mathbf{K}P)^{\mathcal{M}}$.

The second part of the claim where e is unnamed can be shown with symmetrical arguments.

□

Even though actions have only local effects, the knowledge about unnamed object may also change as the following example shows.

Example 6.45. We define an epistemic $\mathcal{ALCO}$-action theory that consists of the following components: an initial knowledge base given by

$$\mathcal{K} := (\mathcal{T} := \emptyset, \mathcal{A} := \{b \in (\forall P.\neg A)\}),$$

and a single ground action

$$\boldsymbol{A} := \{\boldsymbol{\alpha}\} \text{ with } \mathsf{eff}(\boldsymbol{\alpha}) := \{\langle A, \{a\}\rangle^-\} \text{ and } \mathsf{pre}(\boldsymbol{\alpha}) := \emptyset, \mathsf{sense}(\boldsymbol{\alpha}) := (a \in \top).$$

Thus, initially it is only known that all objects that are related to the object name b via the role name P do not belong to A. The only ground action $\boldsymbol{\alpha}$ removes the object a from A and changes nothing else. We are interested in the *known* instances of the concept:

$$C = \forall P.(\neg A \sqcup \neg\{a\})$$

Initially, b is the only known instance of C. However, after doing $\boldsymbol{\alpha}$ all objects are known instances of C. It holds that

$$\mathcal{K} \not\models \top \sqsubseteq C \text{ and } \big(\mathbf{K}C\big)^{\mathcal{M}(\mathcal{K})} = \{b\}.$$

$$\begin{aligned}
[\![X,\kappa]\!] &:= X \text{ with } X \in \mathsf{N_C} \cup \{\top,\bot\} \text{ or } X = \{o\} \text{ for some } o \in \mathsf{Obj}(\Sigma_\mathsf{s})\\
[\![\neg D,\kappa]\!] &:= \neg[\![D,\kappa]\!]\\
[\![D_1 \sqcap D_2,\kappa]\!] &:= [\![D_1,\kappa]\!] \sqcap [\![D_2,\kappa]\!]\\
[\![D_1 \sqcup D_2,\kappa]\!] &:= [\![D_1,\kappa]\!] \sqcup [\![D_2,\kappa]\!]\\
[\![\exists P.E,\kappa]\!] &:= \exists P.[\![E,\kappa]\!]\\
[\![\forall P.E,\kappa]\!] &:= \forall P.[\![E,\kappa]\!]\\
[\![\mathbf{K}D,\kappa]\!] &:= \bigsqcup \kappa(\mathbf{K}[\![D,\kappa]\!])\\
[\![\exists(\mathbf{K}P).D,\kappa]\!] &:= \bigsqcup_{X \in \mathsf{Nom}(\Sigma_\mathsf{s})} \Big(X \sqcap \exists P.\Big(\Big(\bigsqcup \kappa(X,P)\Big) \sqcap [\![D,\kappa]\!]\Big)\Big)\\
[\![\forall(\mathbf{K}P).D,\kappa]\!] &:= \neg[\![\exists \mathbf{K}P.\neg D,\kappa]\!].
\end{aligned}$$

Figure 6.1: Operator for rewriting $\mathbf{K}$ using an instance function

After executing $\boldsymbol{\alpha}$ in $(\mathcal{I},\mathcal{M}(\mathcal{K})) \Longrightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})} (\mathcal{J},\mathcal{M})$ for some $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ we get $\big(\mathbf{K}C\big)^{\mathcal{M}} = \mathsf{N_O}$. Thus, it is possible that the set of all known instances of an $\mathcal{ALCO}$-concept C can be expanded with unnamed elements by executing a local effect action. ▲

In the following we use Lemma 6.42 to equivalently rewrite epistemic concepts into objective concepts using nominals. We have shown that if the concept D is objective, then the epistemic concept $\mathbf{K}D$ is interpreted in an evolving knowledge state as a set of objects that either consists

- only of named objects, or
- it consists of *all* unnamed object plus possibly some of the named objects.

Thus, we observe that the set $(\mathbf{K}D)^{\mathcal{M}} \subseteq \mathsf{N_O}$, where $\mathcal{M}$ is a knowledge state reached via executing a sequence of ground actions, is definable (in any interpretation) as a concept that only mentions nominals: a finite set of named objects is definable with a disjunction of nominal concepts, and the concept

$$\neg\left(\bigsqcup_{o \in \mathsf{Obj}(\Sigma_\mathsf{s})} \{o\}\right)$$

is a definition of the set of all unnamed objects.

To obtain such a definition we introduce the auxiliary notion of an *instance function*. The intuition is that the instance function captures known instances of concepts and known role relationships of object names represented as a set of nominal concepts.

Definition 6.46. Let $\mathsf{Obj}(\Sigma_\mathsf{s})$ the finite set of object names mentioned in the input. An *instance function* maps concepts of the form $\mathbf{K}D$ to a subset of

$$\mathsf{Nom}(\Sigma_\mathsf{s}) := \{\{o\} \mid o \in \mathsf{Obj}(\Sigma_\mathsf{s})\} \cup \{\neg N\} \quad \text{with } N := \bigsqcup_{o \in \mathsf{Obj}(\Sigma_\mathsf{s})} \{o\} \tag{6.3}$$

and an element of $\mathsf{Nom}(\Sigma_\mathsf{s})$ and a role name to a subset of $\mathsf{Nom}(\Sigma_\mathsf{s})$. Let $\mathcal{W}$ be a knowledge state. The *instance function of $\mathcal{W}$*, denoted by $\kappa_\mathcal{W}$, is defined as follows.

$$\begin{aligned}
\kappa_\mathcal{W}(\mathbf{K}D) &:= \{\{o\} \mid o \in (\mathbf{K}D)^\mathcal{W}, o \in \mathsf{Obj}(\Sigma_\mathsf{s})\} \cup \\
& \{\neg N \mid \exists c.c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s}), c \in (\mathbf{K}D)^\mathcal{W}\}; \\
\kappa_\mathcal{W}(\{o\}, P) &:= \left\{\{b\} \,\middle|\, (o,b) \in (\mathbf{K}P)^\mathcal{W}, b \in \mathsf{Obj}(\Sigma_\mathsf{s})\right\} \cup \\
& \left\{\neg N \,\middle|\, \exists c.c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s}), (o,c) \in (\mathbf{K}P)^\mathcal{W}\right\} \\
\kappa_\mathcal{W}(\neg N, P) &:= \left\{\{b\} \,\middle|\, b \in \mathsf{Obj}(\Sigma_\mathsf{s}), \exists c.c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s}), (c,b) \in (\mathbf{K}P)^\mathcal{W}\right\} \cup \\
& \left\{\neg N \,\middle|\, \exists c, d.c, d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s}), (c,d) \in (\mathbf{K}P)^\mathcal{W}\right\}
\end{aligned}$$

for all concepts of the form $\mathbf{K}D$, $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$ and $P \in \mathsf{N_R}$. ▲

We define an operator $[\![\cdot,\cdot]\!]$ that rewrites possibly epistemic concepts into purely objective ones given an instance function.

Definition 6.47. Given an $\mathcal{ALCOK}$-concept D over object names from $\mathsf{Obj}(\Sigma_\mathsf{s})$ and an instance function κ, an objective concept $[\![D,\kappa]\!]$ is defined by induction on the structure of D as shown in Figure 6.1 where $\mathsf{Nom}(\Sigma_\mathsf{s})$ is defined as in (6.3). ▲

Lemma 6.48. *Let $\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}, \sim_\mathsf{s})$ be the epistemic FO-DS induced by $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$, $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ an interpretation, $\sigma \in \boldsymbol{A}^*$ a sequence of ground actions, C an $\mathcal{ALCOK}$-concept over object names from $\mathsf{Obj}(\Sigma_\mathsf{s})$ and $\mathcal{M}$ the epistemic state with $(\mathcal{I}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow^\sigma_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_\sigma, \mathcal{M})$. It holds that*

$$C^{\mathcal{J},\mathcal{M}} = [\![C, \kappa_\mathcal{M}]\!]^\mathcal{J}$$

for any $\mathcal{J} \in \mathcal{M}$.

Proof. It follows from the definition of $[\![\cdot,\cdot]\!]$ that $[\![C,\kappa_\mathcal{M}]\!]$ is objective. Therefore, $[\![C,\kappa_\mathcal{M}]\!]^\mathcal{J}$ is well-defined. Note that the operator $[\![\cdot,\cdot]\!]$ implies the instance function only to concepts of the form $\mathbf{K}D$ where D is objective. We show the claim by induction on the structure of C. The claim trivially holds if C is of the form A with $A \in \mathsf{N_C}$, $\{o\}$ with $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$ or $\top$ or $\bot$.

$C = \neg D$: Assume by induction $D^{\mathcal{J},\mathcal{M}} = [\![D,\kappa_\mathcal{M}]\!]^\mathcal{J}$:

$$(\neg D)^{\mathcal{J},\mathcal{M}} = \mathsf{N_O} \setminus D^{\mathcal{J},\mathcal{M}} = \mathsf{N_O} \setminus [\![D,\kappa_\mathcal{M}]\!]^\mathcal{J} = (\neg[\![D,\kappa_\mathcal{M}]\!])^\mathcal{J} = [\![\neg D,\kappa_\mathcal{M}]\!]^\mathcal{J}.$$

$C = \mathbf{K}D$:

"⇒":

Let $d \in (\mathbf{K}D)^{\mathcal{J},\mathcal{M}}$ for some $\mathcal{J} \in \mathcal{M}$. We show $d \in [\![\mathbf{K}D,\kappa_\mathcal{M}]\!]^\mathcal{J}$. It holds that

$$d \in \bigcap_{\mathcal{Y} \in \mathcal{M}} D^{\mathcal{Y},\mathcal{M}}.$$

Since by induction we have $D^{\mathcal{Y},\mathcal{M}} = [\![D,\kappa_\mathcal{M}]\!]^\mathcal{Y}$, it is implied that

$$d \in \bigcap_{\mathcal{Y} \in \mathcal{M}} [\![D,\kappa_\mathcal{M}]\!]^\mathcal{Y}.$$

First, assume $d \in \mathsf{Obj}(\Sigma_\mathsf{s})$ is named. By definition of the instance function it follows that $\{d\} \in \kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!])$. Therefore, $\{d\}$ is a disjunct in

$$[\![\mathbf{K}D, \kappa_\mathcal{M}]\!] = \bigsqcup \kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!]).$$

and it follows that $d \in [\![\mathbf{K}D, \kappa_\mathcal{M}]\!]^\mathcal{J}$.

Next, assume $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$. It is implied that $\neg N \in \kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!])$. Therefore, $\neg N$ is a disjunct in $[\![\mathbf{K}D, \kappa_\mathcal{M}]\!]$ with $d \in (\neg N)^\mathcal{J}$. Consequently, $d \in [\![\mathbf{K}D, \kappa_\mathcal{M}]\!]^\mathcal{J}$.

"$\Leftarrow$":

Let $d \in [\![\mathbf{K}D, \kappa_\mathcal{M}]\!]^\mathcal{J}$ for an interpretation $\mathcal{J} \in \mathcal{M}$ and $d \in \Delta$. We have

$$[\![\mathbf{K}D, \kappa_{\mathcal{W}_n}]\!] = \bigsqcup \kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!]) \text{ with}$$
$$\kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!]) \subseteq \{\{o\} \mid o \in \mathsf{Obj}(\Sigma_\mathsf{s})\} \cup \{\neg N\}.$$

First, assume $d \in \mathsf{Obj}(\Sigma_\mathsf{s})$. Since $d \notin (\neg N)^\mathcal{I}$, it follows that $\{d\} \in \kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!])$ and therefore

$$d \in \bigcap_{\mathcal{Y} \in \mathcal{M}} [\![D, \kappa_\mathcal{M}]\!]^\mathcal{Y} = \bigcap_{\mathcal{Y} \in \mathcal{M}} D^{\mathcal{Y},\mathcal{M}}$$

by induction. Consequently, $d \in (\mathbf{K}D)^{\mathcal{J},\mathcal{M}}$.

Now, assume $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$. It follows that $\neg N \in \kappa_\mathcal{M}(\mathbf{K}[\![D, \kappa_\mathcal{M}]\!])$. By definition of $\kappa_\mathcal{M}$ it follows that there exists a $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that

$$c \in \bigcap_{\mathcal{Y} \in \mathcal{M}} [\![D, \kappa_\mathcal{M}]\!]^\mathcal{Y} = \bigcap_{\mathcal{Y} \in \mathcal{M}} D^{\mathcal{Y},\mathcal{M}}$$

by induction. With $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ and $c \in (\mathbf{K}D)^\mathcal{M}$ using Lemma 6.42 it follows that $\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s}) \subseteq (\mathbf{K}D)^\mathcal{M}$. Hence, $d \in (\mathbf{K}D)^\mathcal{M}$.

$C = \exists(\mathbf{K}P).D$:

"$\Rightarrow$:"

Let $d \in (\exists(\mathbf{K}P).D)^{\mathcal{J},\mathcal{M}}$ for some interpretation $\mathcal{J} \in \mathcal{M}$ and $d \in \mathsf{N_O}$. We prove that

$$d \in [\![(\exists(\mathbf{K}P).D), \kappa_\mathcal{M}]\!]^\mathcal{J}.$$

There exists $e \in D^{\mathcal{J},\mathcal{M}}$ with $(d,e) \in (\mathbf{K}P)^\mathcal{M}$. There is exactly one $X_d \in \mathsf{Nom}(\Sigma_\mathsf{s})$ with $d \in (X_d)^\mathcal{J}$ and exactly one $X_e \in \mathsf{Nom}(\Sigma_\mathsf{s})$ with $e \in (X_e)^\mathcal{J}$. Note that all concepts in $\mathsf{Nom}(\Sigma_\mathsf{s})$ are interpreted as the same set in each interpretation satisfying the SNA. By definition of $\kappa_\mathcal{M}$ we have that $(d,e) \in (\mathbf{K}P)^\mathcal{M}$ implies $X_e \in \kappa_\mathcal{M}(X_d, P)$. It follows that the concept

$$X_d \sqcap \exists P.\Big(\Big(\bigsqcup \kappa(X_d, P)\Big) \sqcap [\![D, \kappa_\mathcal{M}]\!]\Big)$$

is a disjunct in $[\![(\exists(\mathbf{K}P).D), \kappa_\mathcal{M}]\!]$. We have that $\mathcal{J} \in \mathcal{M}$ and $(d,e) \in (\mathbf{K}P)^\mathcal{M}$ implies $(d,e) \in P^\mathcal{J}$. By induction $e \in D^{\mathcal{J},\mathcal{M}}$ implies $e \in [\![D, \kappa_\mathcal{M}]\!]^\mathcal{J}$. Since $e \in (X_e)^\mathcal{J}$ and

$X_e \in \kappa_{\mathcal{M}}(X_d, P)$ we get

$$e \in \Big(\bigsqcup \kappa(X_d, P)\Big)^{\mathcal{J}}.$$

Consequently,

$$d \in \Big(X_d \sqcap \exists P.\Big(\Big(\bigsqcup \kappa(X_d, P)\Big) \sqcap [\![D, \kappa_{\mathcal{M}}]\!]\Big)\Big)^{\mathcal{J}}$$

which implies $d \in [\![(\exists(\mathbf{K}P).D), \kappa_{\mathcal{M}}]\!]^{\mathcal{J}}$.

"$\Leftarrow$:"

Let $d \in [\![(\exists(\mathbf{K}P).D), \kappa_{\mathcal{M}}]\!]^{\mathcal{J}}$ for some $d \in \mathsf{N_O}$ and $\mathcal{J} \in \mathcal{M}$. We have to show that $d \in (\exists(\mathbf{K}P).D)^{\mathcal{J},\mathcal{M}}$. By definition of $[\![(\exists(\mathbf{K}P).D), \kappa_{\mathcal{M}}]\!]$ it follows that there exists $X_d \in \mathsf{Nom}(\Sigma_\mathsf{s})$ with $d \in (X_d)^{\mathcal{J}}$ and an element $e \in \mathsf{N_O}$ such that $(d, e) \in P^{\mathcal{J}}$, $e \in (X_e)^{\mathcal{J}}$ for some $X_e \in \kappa_{\mathcal{M}}(X_d, P)$ and $e \in [\![D, \kappa_{\mathcal{M}}]\!]$. Using the induction hypothesis we get $e \in D^{\mathcal{J},\mathcal{M}}$. Lemma 6.42 and $X_e \in \kappa_{\mathcal{M}}(X_d, P)$ with $d \in (X_d)^{\mathcal{J}}$ and $e \in (X_e)^{\mathcal{J}}$ implies $(d, e) \in (\mathbf{K}P)^{\mathcal{M}}$. Consequently, $d \in d \in (\exists(\mathbf{K}P).D)^{\mathcal{J},\mathcal{M}}$.

We omit the remaining cases. They can be proven using the induction hypothesis and the semantics of concepts. □

One consequence of the lemma above is that the characterization formulated in Lemma 6.42 also holds for arbitrary *subjective* $\mathcal{ALCOK}$-concepts.

Lemma 6.49. *Let* $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ *be an epistemic* $\mathcal{ALCO}$*-action theory and let* $\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}, \sim_\mathsf{s})$ *be the induced epistemic FO-DS. Furthermore, let C be a subjective* $\mathcal{ALCOK}$*-concept over object names from* $\mathsf{Obj}(\Sigma_\mathsf{s})$*,* $\sigma \in \boldsymbol{A}^*$*,* $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ *and* $(\mathcal{J}, \mathcal{M})$ *the epistemic interpretation with* $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})} (\mathcal{J}, \mathcal{M})$*. It holds that*

1. $C^{\mathcal{Y},\mathcal{M}} = C^{\mathcal{Y}',\mathcal{M}}$ *for all* $\mathcal{Y}, \mathcal{Y}' \in \mathcal{M}$*;*

2. $C^{\mathcal{Y},\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset$ *implies* $(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})) \subseteq C^{\mathcal{Y},\mathcal{M}}$ *for all* $\mathcal{Y} \in \mathcal{M}$*.*

Proof. The proof is by induction on the structure of the subjective concept C. The first claim generally holds for arbitrary knowledge states and arbitrary subjective concepts. The second claim is specific to those knowledge states as described in the claim and to subjective concepts that mention only object names from $\mathsf{Obj}(\Sigma_\mathsf{s})$. We only present proof details for the second claim.

The cases where C is of the form $\{o\}$ for some $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$, $\top$ or $\bot$ are obvious. Let C be of the form $\mathbf{K}D$, where D is an $\mathcal{ALCOK}$-concept over object names from $\mathsf{Obj}(\Sigma_\mathsf{s})$. Lemma 6.48 implies $(\mathbf{K}D)^{\mathcal{Y},\mathcal{M}} = (\mathbf{K}([\![D, \kappa_{\mathcal{M}}]\!]))^{\mathcal{Y},\mathcal{M}}$ for all $\mathcal{Y} \in \mathcal{M}$. Since the concept $[\![D, \kappa_{\mathcal{M}}]\!]$ is objective, Lemma 6.42 yields the claim. For the induction step we distinguish the following cases.

$C = D_1 \sqcap D_2$: Assume $(D_1 \sqcap D_2)^{\mathcal{Y},\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset$ for some $\mathcal{Y} \in \mathcal{M}$ and let $d \in \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big)$. We have to show that $d \in (D_1 \sqcap D_2)^{\mathcal{Y},\mathcal{M}}$. $(D_1 \sqcap D_2)^{\mathcal{Y},\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset$ implies

$$(D_i)^{\mathcal{Y},\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset \text{ for all } i \in 1, 2.$$

The induction hypothesis yields

$$\big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \subseteq (D_i)^{\mathcal{Y},\mathcal{M}} \text{ for all } i \in \{1,2\}.$$

Hence, $d \in (D_1 \sqcap D_2)^{\mathcal{Y},\mathcal{M}}$.

$C = \neg D$: Assume $(\neg D)^{\mathcal{Y},\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset$ for some $\mathcal{Y} \in \mathcal{M}$ and let $d \in \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big)$. We have to show that $d \in (\neg D)^{\mathcal{Y},\mathcal{M}}$. To the contrary assume $d \notin (\neg D)^{\mathcal{Y},\mathcal{M}}$. Consequently, $d \in D^{\mathcal{Y},\mathcal{M}}$. Since $d \in \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big)$, the induction hypothesis yields $\big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \subseteq D^{\mathcal{Y},\mathcal{M}}$. This is a contradiction to $(\neg D)^{\mathcal{Y},\mathcal{M}} \cap \big(\mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})\big) \neq \emptyset$. Therefore, $d \in (\neg D)^{\mathcal{Y},\mathcal{M}}$ follows.

$C = \exists \mathbf{K}P.D$: This case follows directly from part two of Lemma 6.42 about epistemic roles and the induction hypothesis.

□

Thus, if C is subjective, then the rewriting result $[\![C, \kappa_\mathcal{M}]\!]$ can be equivalently written as a disjunction of concepts from $\mathsf{Nom}(\Sigma_\mathsf{s})$ (see (6.3)).

To handle the knowledge constructor in front of axioms and Boolean combinations thereof we extend the domain operator $[\![\cdot,\cdot]\!]$ also to epistemic Boolean KBs. For doing this, the domain of an instance function is extended to KBs as well. An instance function κ maps an epistemic Boolean KB of the form $\mathbf{K}\psi$ either to TRUE ($o \sqsubseteq \top$) or FALSE ($o \sqsubseteq \bot$), where $o \in \mathsf{N_O}$ is arbitrary but fixed.

Let $\kappa_\mathcal{W}$ be the instance function of an epistemic state $\mathcal{W}$ and $\mathbf{K}\psi$ an epistemic Boolean $\mathcal{ALCOK}$-KB. We define

$$\kappa_\mathcal{W}(\mathbf{K}\psi) := \begin{cases} o \sqsubseteq \top & \text{if } (\mathcal{I},\mathcal{W}) \mid\!\models \psi \text{ for all } \mathcal{I} \in \mathcal{W}; \\ o \sqsubseteq \bot & \text{otherwise.} \end{cases}$$

Let κ be an instance function and ψ an epistemic Boolean $\mathcal{ALCOK}$-KB. An objective Boolean $\mathcal{ALCO}$-KB, denoted by $[\![\psi,\kappa]\!]$, is defined inductively such that in addition to the definitions in Figure 6.1 we now also have:

$$\begin{aligned}
[\![o \sqsubseteq C, \kappa]\!] &:= o \sqsubseteq [\![C,\kappa]\!];\\
[\![(o,o') \sqsubseteq P, \kappa]\!] &:= (o,o') \sqsubseteq P;\\
[\![C \sqsubseteq D, \kappa]\!] &:= [\![C,\kappa]\!] \sqsubseteq [\![D,\kappa]\!];\\
[\![o_1 \approx o_2, \kappa]\!] &:= o_1 \approx o_2;\\
[\![\neg\psi, \kappa]\!] &:= \neg[\![\psi,\kappa]\!];\\
[\![\psi_1 \wedge \psi_2, \kappa]\!] &:= [\![\psi_1,\kappa]\!] \wedge [\![\psi_2,\kappa]\!];\\
[\![\mathbf{K}\psi, \kappa]\!] &:= \kappa(\mathbf{K}([\![\psi,\kappa]\!])).
\end{aligned}$$

The following lemma is analogous to Lemma 6.48.

Lemma 6.50. *Let ψ be a Boolean $\mathcal{ALCOK}$-KB, $\mathcal{W}$ an epistemic state and $\kappa_\mathcal{W}$ the corresponding instance function. It holds that*

$$(\mathcal{I},\mathcal{W}) \mid\!\models \psi \textit{ iff } \mathcal{I} \models [\![\psi,\kappa_\mathcal{W}]\!]$$

for any interpretation $\mathcal{I} \in \mathcal{W}$.

Since the instance function is applied only to concepts of the form $\mathbf{K}D$ where D is objective and to Boolean KBs $\mathbf{K}\psi$ where ψ is objective, we only need to determine the known instances of objective concepts and the entailment of objective KBs after executing the given sequence of ground actions σ. For solving the projection problem we need to consider the instance function of any possible epistemic state that evolves from the epistemic model of the initial KB by executing σ. There can be more than one such epistemic state because we need to take all possible sensing results of σ into account. Sensing in our formalization means observing the truth of a Boolean $\mathcal{ALCO}$-KB. We represent the sensing result of an action sequence as a sequence of Boolean $\mathcal{ALCO}$-KBs.

Definition 6.51. Let $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory and $\sigma = \boldsymbol{\alpha}_1\boldsymbol{\alpha}_2 \ldots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ for some $n \geq 0$. A *sensing result of σ w.r.t.* Σ_s is a sequence of Boolean $\mathcal{ALCO}$-KBS $\mathsf{s}_\sigma = (\psi_1, \ldots, \psi_n)$ such that for all $i \in \{1, \ldots, n\}$ we have

$$\psi_i \in \{\mathsf{sense}(\boldsymbol{\alpha}_i), \neg\mathsf{sense}(\boldsymbol{\alpha}_i)\}.$$

Let

$$\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_\mathsf{s})$$

be the epistemic FO-DS induced by Σ_s and let $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ be an interpretation and $\mathcal{I}_0, \ldots, \mathcal{I}_n$ the sequence of interpretations with

$$\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_1} \mathcal{I}_1 \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_2} \cdots \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_n} \mathcal{I}_n.$$

The *sensing result of σ in $\mathcal{I}_0$*, denoted by $\mathsf{s}_\sigma(\mathcal{I}_0)$, is a sensing result of σ w.r.t. Σ_s of the form $(\psi_1, \ldots, \psi_n)$ such that $\mathcal{I}_j \models \psi_{j+1}$ for all $j \in \{0, \ldots, n-1\}$.

We call a sensing result s'_σ of σ w.r.t. Σ_s *consistent* iff there exists an interpretation $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ with $\mathsf{s}'_\sigma = \mathsf{s}_\sigma(\mathcal{I})$.

▲

The epistemic state evolving from $\mathcal{M}(\mathcal{K})$ as the result of doing σ only depends on the sensing result provided by the interpretation representing the actual state of the world. Formally, for any two interpretations $\mathcal{I}, \mathcal{Y} \in \mathcal{M}(\mathcal{K})$ and there epistemic interpretations $(\mathcal{I}', \mathcal{W})$ and $(\mathcal{Y}', \mathcal{M})$ with

$$(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})}^{\sigma} (\mathcal{I}', \mathcal{W}) \text{ and } (\mathcal{Y}, \mathcal{M}(\mathcal{K})) \Longrightarrow_{\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s})}^{\sigma} (\mathcal{Y}', \mathcal{M})$$

it holds that $\mathsf{s}_\sigma(\mathcal{I}) = \mathsf{s}_\sigma(\mathcal{Y})$ implies $\mathcal{W} = \mathcal{M}$ (but *not* necessarily $\mathcal{I}' = \mathcal{Y}'$).

In the following we define an objective *reduction KB* following Baader et al. [Baa+05a]. On the one hand it will allow us to check consistency of sensing results and on the other hand it will serve as a representation of the resulting epistemic state that can be used to compute the images of the corresponding instance function via standard $\mathcal{ALCO}$ entailment checks.

Given

- an epistemic $\mathcal{ALCO}$-action theory $\Sigma_\mathsf{s} = (\mathcal{K} = (\mathcal{T}, \mathcal{A}), \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$, and
- a ground action sequence $\sigma = \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ for some $n \geq 0$,

we define a Boolean $\mathcal{ALCO}$-KB denoted by

$$\mathcal{K}_{\mathsf{red}}^{\sigma}$$

that encodes the following set of interpretations

$$\{\mathcal{I}_\sigma \mid \mathcal{I} \in \mathcal{M}(\mathcal{K}), \mathcal{I} \Rightarrow_{\mathfrak{D}}^{\sigma} \mathcal{I}_\sigma\},$$

where $\mathfrak{D} = (\mathbb{I}, \mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \succ_{\mathsf{poss}})$ is the FO-DS induced by $(\mathcal{K} = (\mathcal{T}, \mathcal{A}), \boldsymbol{A}, \mathsf{pre}, \mathsf{eff})$.

The construction is very similar to the one used for reducing realizability of dynamic types to $\mathcal{ALCO}$-KB consistency.

With $\mathsf{sub}(\Sigma_\mathsf{s})$ we denote the set of all sub-concepts occurring in the input Σ_s. For the construction of $\mathcal{K}_{\mathsf{red}}^{\sigma}$ we use the following additional symbols:

- for each concept name or role name $F \in \mathcal{F}$ there are new (not contained in $\mathcal{F}$) concept names or role names, respectively, of the form $F^{(0)}, \dots, F^{(n)}$;
- for each concept $C \in \mathsf{sub}(\Sigma_\mathsf{s})$ there are new (not from $\mathcal{F}$) concept names $T_C^{(0)}, \dots, T_C^{(n)}$;
- an object name $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$.

The TBox denoted by

$$\mathcal{T}_{\mathsf{sub}(\Sigma_\mathsf{s})}^{\sigma}$$

consists of definitions for all names of the form $T_C^{(i)}$ with $C \in \mathsf{sub}(\Sigma_\mathsf{s})$ and $i \in \{0, \dots, n\}$ as defined in Figure 6.2 and it includes the definition of the concept name N_c that stands for the set of all named elements (including c):

$$N_c \equiv \bigsqcup_{o \in \mathsf{Obj}(\Sigma_\mathsf{s}) \cup \{c\}} \{o\}.$$

Next, the conditional action effects are incorporated. The effect conditions are expressed using the newly defined concept names of the form $T_C^{(i)}$: Let ψ be a Boolean $\mathcal{ALCO}$-KB with $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\Sigma_\mathsf{s})$ and with object names from $\mathsf{Obj}(\Sigma_\mathsf{s}) \cup \{c\}$ and let $i \in \{0, \dots, n\}$. The *i-th copy of* ψ is a Boolean $\mathcal{ALCO}$-KB, denoted by $\psi^{(i)}$, that is obtained from ψ by replacing each CI $C \sqsubseteq D$ in ψ by $T_C^{(i)} \sqsubseteq T_D^{(i)}$, each concept assertion $o \sqsubseteq C$ in ψ by $o \sqsubseteq T_C^{(i)}$ and each role assertion $(o, o') \sqsubseteq P$ in ψ by $(o, o') \sqsubseteq P^{(i)}$.

For each ground action $\boldsymbol{\alpha}_i$ in σ with $i \in \{1, \dots, n\}$ the following Boolean KB captures the action effects:

$$\begin{aligned}
\mathcal{A}_{\mathsf{eff}}^{(i)} := & \bigwedge_{\psi \rhd \langle A, \{o\}\rangle^+ \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \psi^{(i-1)} \to \big(o \sqsubseteq A^{(i)}\big) \wedge \\
& \bigwedge_{\psi \rhd \langle A, \{o\}\rangle^- \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \psi^{(i-1)} \to \neg\big(o \sqsubseteq A^{(i)}\big) \wedge \\
& \bigwedge_{\psi \rhd \langle P, \{(o,o')\}\rangle^+ \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \psi^{(i-1)} \to \big((o, o') \sqsubseteq P^{(i)}\big) \wedge \\
& \bigwedge_{\psi \rhd \langle P, \{(o,o')\}\rangle^- \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \psi^{(i-1)} \to \neg\big((o, o') \sqsubseteq P^{(i)}\big).
\end{aligned}$$

$$\begin{aligned}
T_A^{(i)} &\equiv (N_c \sqcap A^{(i)}) \sqcup (\neg N_c \sqcap A^{(0)}), \text{ with } A \in \mathcal{F} \cap \mathsf{N_C};\\
T_B^{(i)} &\equiv B \text{ with } B \text{ of the form } \{o\}, \top \text{ or } \bot;\\
T_{\neg C}^{(i)} &\equiv \neg T_C^{(i)};\\
T_{C \sqcap D}^{(i)} &\equiv T_C^{(i)} \sqcap T_D^{(i)};\\
T_{C \sqcup D}^{(i)} &\equiv T_C^{(i)} \sqcup T_D^{(i)};\\
T_{\exists P.C}^{(i)} &\equiv \Big(N_c \sqcap \big(\ (\exists P^{(0)}.(\neg N_c \sqcap T_C^{(i)})\ ) \sqcup (\ \exists P^{(i)}.(N_c \sqcap T_C^{(i)})\ )\ \big)\Big) \sqcup\\
&\quad (\neg N_c \sqcap \exists P^{(0)}.T_C^{(i)});\\
T_{\forall P.C}^{(i)} &\equiv \Big(N_c \rightarrow \big(\ (\forall P^{(0)}.(\neg N_c \rightarrow T_C^{(i)})\ ) \sqcap (\ \forall P^{(i)}.(N_c \rightarrow T_C^{(i)})\ )\ \big)\Big) \sqcap\\
&\quad (\neg N_c \rightarrow \forall P^{(0)}.T_C^{(i)}).
\end{aligned}$$

Figure 6.2: Concept definition for $T_C^{(i)}$

The Boolean KB $\mathcal{A}_{\mathrm{min}}^{(i)}$ for each $\boldsymbol{\alpha}_i$ in σ axiomatizes the frame assumption for named object names:

$$\mathcal{A}_{\mathrm{min}}^{(i)} := \left(\bigwedge_{o \in \mathsf{Obj}(\Sigma_\mathsf{s}) \cup \{c\}} \left(\bigwedge_{A \in \mathcal{F} \cap \mathsf{N_C}} \gamma_{A,o}^i\right)\right) \wedge \left(\bigwedge_{o,o' \in \mathsf{Obj}(\Sigma_\mathsf{s}) \cup \{c\}} \left(\bigwedge_{P \in \mathcal{F} \cap \mathsf{N_R}} \gamma_{P,o,o'}^i\right)\right),$$

where

$$\begin{aligned}
\gamma_{A,o}^i := &\left(\left(\left(o \in A^{(i-1)}\right) \wedge \bigwedge_{\psi \rhd \langle A,\{o\}\rangle^- \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \neg\psi^{(i-1)}\right) \rightarrow \left(o \in A^{(i)}\right)\right) \wedge\\
&\left(\neg\left(o \in A^{(i-1)}\right) \wedge \bigwedge_{\psi \rhd \langle A,\{o\}\rangle^+ \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \neg\psi^{(i-1)}\right) \rightarrow \neg\left(o \in A^{(i)}\right)\\
\gamma_{P,o,o'}^i := &\left(\left(\left((o,o') \in P^{(i-1)}\right) \wedge \bigwedge_{\psi \rhd \langle P,\{(o,o')\}\rangle^- \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \neg\psi^{(i-1)}\right) \rightarrow \left((o,o') \in P^{(i)}\right)\right) \wedge\\
&\left(\neg\left((o,o') \in P^{(i-1)}\right) \wedge \bigwedge_{\psi \rhd \langle P,\{(o,o')\}\rangle^+ \in \mathsf{eff}(\boldsymbol{\alpha}_i)} \neg\psi^{(i-1)}\right) \rightarrow \neg\left((o,o') \in P^{(i)}\right).
\end{aligned}$$

The overall reduction KB is given by

$$\mathcal{K}_{\mathrm{red}}^{\sigma} := \mathcal{T}_{\mathsf{sub}(\Sigma_\mathsf{s})}^{\sigma} \wedge \left(\bigwedge_{\varphi \text{ occurs in } \mathcal{K}} \varphi^{(0)}\right) \wedge \mathcal{A}_{\mathrm{eff}}^{(0)} \wedge \cdots \wedge \mathcal{A}_{\mathrm{eff}}^{(n)} \wedge \mathcal{A}_{\mathrm{min}}^{(0)} \wedge \cdots \wedge \mathcal{A}_{\mathrm{min}}^{(n)}. \tag{6.4}$$

Since we also want to use the reduction KB for rewriting the subjective projection query into an objective one, it is necessary to extend $\mathcal{K}_{\mathrm{red}}^{\sigma}$ with additional concept definitions for sub-concepts occurring within the scope of a $\mathbf{K}$ in the given projection query. To keep the size of the resulting objective rewriting result small an additional acyclic $\mathcal{ALCO}$-TBox is considered.

Let $\mathcal{T}_\mathsf{a}$ be an acyclic $\mathcal{ALCO}$-TBox such that

- all defined concept names in $\mathcal{T}_\mathsf{a}$ are not contained in $\mathcal{F}$, and
- all other non-defined concept names plus all role names in $\mathcal{T}_\mathsf{a}$ are contained in $\mathcal{F}$.

Let X be an $\mathcal{ALCO}$-concept or a Boolean $\mathcal{ALCO}$-KB over concept names and role names from $\mathcal{F}$ and defined concept names from $\mathcal{T}_\mathsf{a}$. For each sub-concept $C \in \mathsf{sub}(X) \cup \mathsf{sub}(\mathcal{T}_\mathsf{a})$ and each $i \in \{0,\dots,n\}$ a new concept name $T_C^{(i)}$ is introduced. With

$$\mathcal{T}^{\sigma}_{\mathsf{sub}(X,\mathcal{T}_\mathsf{a})} \tag{6.5}$$

we denote the conjunction of the following concept definitions:

- for all $C \in \mathsf{sub}(X) \cup \mathsf{sub}(\mathcal{T}_\mathsf{a})$ where C is not a defined name in $\mathcal{T}_\mathsf{a}$, and all $i \in \{0,\dots,n\}$ a definition of $T_C^{(i)}$ according to Figure 6.2;
- $T_A^{(i)} \equiv T_D^{(i)}$ for each definition $A \equiv D \in \mathcal{T}_\mathsf{a}$ and each $i \in \{0,\dots,n\}$.

Let Y be a Boolean $\mathcal{ALCO}$-KB or an $\mathcal{ALCO}$-concept. With $Y_{\mathcal{T}_\mathsf{a}}$ we denote the Boolean KB or concept obtained from Y be exhaustively replacing each occurrence of a defined name in $\mathcal{T}_\mathsf{a}$ in Y with the corresponding right-hand side of the definition.

In the next lemma we show that the models of $\mathcal{K}^{\sigma}_{\mathsf{red}}$ correctly encode the sequences of interpretations generated from models of $\mathcal{K}$ by σ. In the following we only talk about interpretations satisfying the SNA.

Lemma 6.52. *Let $\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_\mathsf{s})$ be the epistemic FO-DS induced by Σ_s. Let $\sigma = \boldsymbol{\alpha}_1\boldsymbol{\alpha}_2\dots\boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ be a sequence of ground actions,*

and let X, $\mathcal{T}_\mathsf{a}$ and

$$\mathcal{K}^{\sigma}_{red} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(X,\mathcal{T}_\mathsf{a})}$$

be as described above.

1. *For every sequence $\mathcal{I}_0,\dots,\mathcal{I}_n$ with $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ and $\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_1} \mathcal{I}_1 \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_2} \cdots \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_n} \mathcal{I}_n$, there exists a $\mathcal{J}$ such that $\mathcal{J} \models \mathcal{K}^{\sigma}_{red}$ and it holds that*

$$\mathcal{I}_j \models \psi_{\mathcal{T}_\mathsf{a}} \text{ iff } \mathcal{J} \models \psi^{(j)}, \text{for all } j \in \{0,\dots,n\}$$

   and for any Boolean $\mathcal{ALCO}$-KB ψ with $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\Sigma_\mathsf{s}) \cup \mathsf{sub}(X,\mathcal{T}_\mathsf{a})$.

2. *For every interpretation $\mathcal{J}$ with $\mathcal{J} \models \mathcal{K}^{\sigma}_{red}$, there exists $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ such that for the sequence $\mathcal{I}_0,\dots,\mathcal{I}_n$ with $\mathcal{I}_0 \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_1} \mathcal{I}_1 \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_2} \cdots \Rightarrow_{\mathfrak{D}}^{\boldsymbol{\alpha}_n} \mathcal{I}_n$ it holds that*

$$\mathcal{I}_j \models \psi_{\mathcal{T}_\mathsf{a}} \text{ iff } \mathcal{J} \models \psi^{(j)}, \text{for all } j \in \{0,\dots,n\}$$

   and for any Boolean $\mathcal{ALCO}$-KB ψ with $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\Sigma_\mathsf{s}) \cup \mathsf{sub}(X,\mathcal{T}_\mathsf{a})$.

Proof. Essentially, the proof works in the same way as the one given in [Baa+05b] (page 17, Theorem 14). □

Let $\mathsf{s}_\sigma = (\psi_1, \dots, \psi_n)$ be a sensing result of σ w.r.t. Σ_s. The corresponding reduction KB, denoted by $\mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma)$, is given by:

$$\mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) := {\psi_1}^{(0)} \wedge {\psi_2}^{(1)} \wedge \dots \wedge {\psi_n}^{(n-1)}.$$

Lemma 6.53. *Let Σ_s, σ be as above and s'_σ a sensing result of σ w.r.t. Σ_s.*
It holds that s'_σ is consistent w.r.t. Σ_s iff the Boolean $\mathcal{ALCO}$-KB $\mathcal{K}^\sigma_{red} \wedge \mathcal{K}_{sen}(\mathsf{s}'_\sigma)$ is consistent.

Proof. Let $\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_\mathsf{s})$.
"⇒":
Assume there exists an $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ such that $\mathsf{s}'_\sigma = \mathsf{s}_\sigma(\mathcal{I}_0)$. Let $\mathcal{I}_0, \dots, \mathcal{I}_n$ be such that

$$\mathcal{I}_0 \Rightarrow^{\alpha_1}_{\mathfrak{D}} \mathcal{I}_1 \Rightarrow^{\alpha_2}_{\mathfrak{D}} \cdots \Rightarrow^{\alpha_n}_{\mathfrak{D}} \mathcal{I}_n.$$

Let $\mathsf{s}'_\sigma = \mathsf{s}_\sigma(\mathcal{I}_0) = (\psi_1, \dots, \psi_n)$. For all $j \in \{0, \dots, n-1\}$ it holds by assumption that

$$\mathcal{I}_j \models \psi_{j+1}.$$

From Lemma 6.52.1 it follows that there exists a model $\mathcal{J}$ such that $\mathcal{J} \models \mathcal{K}^\sigma_{\mathrm{red}}$ and

$$\mathcal{J} \models {\psi_{j+1}}^{(j)} \text{ for all } j \in \{0, \dots, n-1\}.$$

This implies $\mathcal{J} \models \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}'_\sigma)$.
"⇐":
Assume $\mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}'_\sigma)$ is consistent with $\mathsf{s}'_\sigma = (\psi_1, \dots, \psi_n)$. There exists a model $\mathcal{J}$ such that

$$\mathcal{J} \models \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}'_\sigma).$$

It follows from Lemma 6.52.2 that there exists $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ such that for the sequence $\mathcal{I}_0, \dots, \mathcal{I}_n$ with

$$\mathcal{I}_0 \Rightarrow^{\alpha_1}_{\mathfrak{D}} \mathcal{I}_1 \Rightarrow^{\alpha_2}_{\mathfrak{D}} \cdots \Rightarrow^{\alpha_n}_{\mathfrak{D}} \mathcal{I}_n.$$

it holds that $\mathcal{I}_j \models \psi_{j+1}$ for all $j \in \{0, \dots, n-1\}$. Consequently, $\mathsf{s}'_\sigma = \mathsf{s}_\sigma(\mathcal{I}_0)$. □

Using the reduction KB for the sensing results we can now represent the knowledge state after a sequence of ground actions has been performed.

Lemma 6.54. *Let Σ_s, $\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})$, σ, X and $\mathcal{T}_\mathsf{a}$ be as described above.*
Furthermore, let $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ be an initial model and $(\mathcal{I}_\sigma, \mathcal{M})$ the epistemic interpretation with

$$(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^\sigma_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_\sigma, \mathcal{M}).$$

1. *For every interpretation $\mathcal{Y} \in \mathcal{M}$ there exists an interpretation $\mathcal{J}$ such that*

$$\mathcal{J} \models \mathcal{K}^\sigma_{red} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(X,\mathcal{T}_\mathsf{a})} \wedge \mathcal{K}_{sen}(\mathsf{s}'_\sigma)$$

   and $\mathcal{Y} \models \psi_{\mathcal{T}_\mathsf{a}}$ iff $\mathcal{J} \models \psi^{(n)}$ for any Boolean $\mathcal{ALCO}$-KB ψ with $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\Sigma_\mathsf{s}) \cup \mathsf{sub}(X, \mathcal{T}_\mathsf{a})$.

2. *For every interpretation $\mathcal{J}$ with*

$$\mathcal{J} \models \mathcal{K}^{\sigma}_{red} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(X,\mathcal{T}_{\mathsf{a}})} \wedge \mathcal{K}_{sen}(\mathsf{s}'_{\sigma})$$

there exists $\mathcal{Y} \in \mathcal{M}$ such that and $\mathcal{Y} \models \psi_{\mathcal{T}_{\mathsf{a}}}$ iff $\mathcal{J} \models \psi^{(n)}$ for any Boolean $\mathcal{ALCO}$-KB ψ with $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\Sigma_{\mathsf{s}}) \cup \mathsf{sub}(X, \mathcal{T}_{\mathsf{a}})$.

Proof. Let $\sigma = \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n$ for some $n \geq 0$, $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ and $(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), (\mathcal{I}_1, \mathcal{W}_1), \ldots, (\mathcal{I}_n, \mathcal{W}_n)$ the sequence of epistemic interpretations with

$$(\mathcal{I}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\boldsymbol{\alpha}_1}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{I}_1, \mathcal{W}_1) \Longrightarrow^{\boldsymbol{\alpha}_2}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} \cdots \Longrightarrow^{\boldsymbol{\alpha}_n}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{I}_n, \mathcal{W}_n).$$

1. Let $\mathcal{J}_n \in \mathcal{W}_n$. There exists a sequence $\mathcal{J}_0, \ldots, \mathcal{J}_n$ with

$$\mathcal{J}_j \Rightarrow^{\boldsymbol{\alpha}_{j+1}}_{\mathfrak{D}} \mathcal{J}_{j+1} \text{ for all } j \in \{0, \ldots, n-1\} \text{ and } \mathcal{J}_i \in \mathcal{W}_i \text{ for all } i \in \{0, \ldots, n\}.$$

   It follows that $\mathcal{I}_i \sim^{\boldsymbol{\alpha}_i}_{\mathsf{s}} \mathcal{J}_i$ for all $i \in \{0, \ldots, n\}$. Therefore, $\mathsf{s}_{\sigma}(\mathcal{I}_0) = \mathsf{s}_{\sigma}(\mathcal{J}_0)$.

   Lemma 6.52.1 implies that there is a model $\mathcal{J}$ with

$$\mathcal{J} \models \mathcal{K}^{\sigma}_{\mathrm{red}} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(X,\mathcal{T}_{\mathsf{a}})}$$

   such that for all $i \in \{0, \ldots, n\}$ it holds that $\mathcal{J}_i \models \psi_{\mathcal{T}_{\mathsf{a}}}$ iff $\mathcal{J} \models \psi^{(i)}$ for any Boolean $\mathcal{ALCO}$-KB ψ with $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(X, \mathcal{T}_{\mathsf{a}}) \cup \mathsf{sub}(\Sigma_{\mathsf{s}})$. Consequently,

$$\mathcal{J} \models \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_{\sigma}(\mathcal{J}_0)).$$

   With $\mathsf{s}_{\sigma}(\mathcal{I}_0) = \mathsf{s}_{\sigma}(\mathcal{J}_0)$ this implies the claim.

2. As for the other direction, Lemma 6.52.2 implies the claim.

□

Now, we are ready to define an algorithm for rewriting the projection query into an objective Boolean $\mathcal{ALCO}$-KB given a consistent sensing result of the action sequence under consideration. The *input* consists of the following components:

- an epistemic $\mathcal{ALCO}$-action theory $\Sigma_{\mathsf{s}} = (\mathcal{K} = (\mathcal{T}, \mathcal{A}), \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$,
- a ground action sequence $\sigma = \boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ for some $n \geq 0$,
- a Boolean $\mathcal{ALCOK}$-KB ψ (projection query)
- a consistent sensing result s_{σ} of σ w.r.t. Σ_{s}.

Let $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_{\mathsf{s}})$ and $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ such that $\mathsf{s}_{\sigma} = \mathsf{s}_{\sigma}(\mathcal{I})$ and let $(\mathcal{I}_{\sigma}, \mathcal{M})$ be the epistemic interpretation with $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{I}_{\sigma}, \mathcal{M})$. We assume that all names mentioned in the projection query are contained in $\mathcal{F}$ and $\mathsf{Obj}(\Sigma_{\mathsf{s}})$.

The *output* of the algorithm is an acyclic $\mathcal{ALCO}$-TBox $\mathcal{T}_{\mathsf{a}}$ and an (objective) Boolean $\mathcal{ALCO}$-KB φ such that $\varphi_{\mathcal{T}_{\mathsf{a}}}$ is equivalent to $[\![\psi, \kappa_{\mathcal{M}}]\!]$. The TBox $\mathcal{T}_{\mathsf{a}}$ is used to obtain a representation of $[\![\psi, \kappa_{\mathcal{M}}]\!]$ that is of polynomial size.

The execution starts with the pair consisting of the empty TBox and the projection query:

$$(\mathcal{T}_\mathsf{a} = \emptyset, \psi).$$

In each execution step either an epistemic sub-concept of the form $\mathbf{K}D$ or $\exists(\mathbf{K}P).D$, where D is subjective, or an epistemic sub-KB $\mathbf{K}\varrho$, where ϱ is objective, is chosen and rewritten into an objective expression by computing the image of the instance function determined by the given sensing result.

Let $\mathcal{T}_\mathsf{a}$ be an acyclic $\mathcal{ALCO}$-TBox such that all concept names and role names in $\mathcal{T}_\mathsf{a}$ are either defined concept names in $\mathcal{T}_\mathsf{a}$ or are contained in $\mathcal{F}$ and let $\mathbf{K}D$ be an $\mathcal{ALCOK}$-concept, where D is objective and mentions only concept names and role names from $\mathcal{F}$ and $\mathcal{T}_\mathsf{a}$ and only object names from $\mathsf{Obj}(\Sigma_\mathsf{s})$. We define a set $\kappa_{\kappa_{\mathsf{s}_\sigma}}(\mathbf{K}D) \subseteq \mathsf{Nom}(\Sigma_\mathsf{s})$, where $\mathsf{Nom}(\Sigma_\mathsf{s})$ is defined in (6.3), as follows

$$\begin{aligned}\kappa_{\mathsf{s}_\sigma}(\mathbf{K}D) :=& \Big\{\{o\} \,\Big|\, o \in \mathsf{Obj}(\Sigma_\mathsf{s}), \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(D,\mathcal{T}_\mathsf{a})} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models \big(o \in T^{(n)}_D\big)\Big\} \cup \\ &\Big\{\neg N \,\Big|\, \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(D,\mathcal{T}_\mathsf{a})} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models \big(c \in T^{(n)}_D\big)\Big\}.\end{aligned} \tag{6.6}$$

Let $P \in \mathcal{F}$ be a role name. We define $\kappa_{\mathsf{s}_\sigma}(X,P) \subseteq \mathsf{Nom}(\Sigma_\mathsf{s})$ for some $X \in \mathsf{Nom}(\Sigma_\mathsf{s})$ as follows as follows:

$$\begin{aligned}\kappa_{\mathsf{s}_\sigma}(\{o\},P) :=\ & \big\{\{o'\} \,\big|\, o' \in \mathsf{Obj}(\Sigma_\mathsf{s}), \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models (o,o') \in P^{(n)}\big\} \cup \\ & \big\{\neg N \,\big|\, \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models (o,c) \in P^{(n)}\big\} \text{ for all } o \in \mathsf{Obj}(\Sigma_\mathsf{s}); \\ \kappa_{\mathsf{s}_\sigma}(\neg N,P) :=\ & \big\{\{o'\} \,\big|\, o' \in \mathsf{Obj}(\Sigma_\mathsf{s}), \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models (c,o') \in P^{(n)}\big\} \cup \\ & \big\{\neg N \,\big|\, \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models (c,c) \in P^{(n)}\big\}.\end{aligned} \tag{6.7}$$

Let $\mathcal{T}_\mathsf{a}$ be as above and $\mathbf{K}\varrho$ a Boolean $\mathcal{ALCOK}$-KB, where ϱ is objective and mentions only concept and role names from $\mathcal{F}$ and $\mathcal{T}_\mathsf{a}$ and object names from $\mathsf{Obj}(\Sigma_\mathsf{s})$. We define

$$\kappa_{\mathsf{s}_\sigma}(\mathbf{K}\varrho) := \begin{cases} o \in \top & \text{if } \mathcal{K}^\sigma_{\mathrm{red}} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(\varrho,\mathcal{T}_\mathsf{a})} \wedge \mathcal{K}_{\mathrm{sen}}(\mathsf{s}_\sigma) \models \varrho^{(n)}; \\ o \in \bot & \text{otherwise.}\end{cases} \tag{6.8}$$

Next, using the instance function $\kappa_{\mathsf{s}_\sigma}$ we define the rewrite steps via a transition relation on pairs of an acyclic TBox and a Boolean $\mathcal{ALCOK}$-KB.

Definition 6.55. Let $\mathcal{T}_\mathsf{a}$ be an acyclic $\mathcal{ALCO}$-TBox where all concept names and role names in $\mathcal{T}_\mathsf{a}$ except for the defined names are contained in $\mathcal{F}$ and let φ be a Boolean $\mathcal{ALCOK}$-KB over $\mathcal{F}$ and defined names from $\mathcal{T}_\mathsf{a}$.

A *binary rewrite relation* "$\vdash_{\kappa_{\mathsf{s}_\sigma}}$" between pairs of the form $(\mathcal{T}_\mathsf{a}, \varphi)$ is defined as follows.

$(\mathcal{T}_\mathsf{a}, \varphi) \vdash^{\mathbf{K}D}_{\kappa_{\mathsf{s}_\sigma}} (\mathcal{T}'_\mathsf{a}, \varphi')$: iff $\mathbf{K}D \in \mathsf{sub}(\varphi)$ is a sub-concept, D is objective, $\mathcal{T}_\mathsf{a} = \mathcal{T}'_\mathsf{a}$, and φ' is obtained from φ by replacing each occurrence of $\mathbf{K}D$ in φ by the following concept

$$\bigsqcup \kappa_{\mathsf{s}_\sigma}(\mathbf{K}D).$$

$(\mathcal{T}_{\mathsf{a}}, \varphi) \vdash_{\kappa_{\mathsf{s}_\sigma}}^{\exists(\mathbf{K}P).D} (\mathcal{T}'_{\mathsf{a}}, \varphi')$: iff $\exists(\mathbf{K}P).D \in \mathsf{sub}(\varphi)$ is a sub-concept, D is objective,

$$\mathcal{T}'_{\mathsf{a}} := \mathcal{T}_{\mathsf{a}} \cup \{A_D \equiv D\},$$

where A_D is a new concept name not occurring in $\mathcal{T}_{\mathsf{a}}$ and in $\mathcal{F}$, and φ' is obtained from φ by replacing each occurrence of $\exists(\mathbf{K}P).D$ in φ by the concept

$$\bigsqcup_{X \in \mathsf{Nom}(\Sigma_{\mathsf{s}})} \Big(X \sqcap \exists P. \Big(\Big(\bigsqcup \kappa_{\mathsf{s}_\sigma}(X, P) \Big) \sqcap A_D \Big) \Big),$$

$(\mathcal{T}_{\mathsf{a}}, \varphi) \vdash_{\kappa_{\mathsf{s}_\sigma}}^{\mathbf{K}\varrho} (\mathcal{T}'_{\mathsf{a}}, \varphi')$: iff $\mathbf{K}\varrho$ is a sub-KB of φ, ϱ is objective, $\mathcal{T}_{\mathsf{a}} = \mathcal{T}'_{\mathsf{a}}$ and φ' is obtained from be replacing each occurrence of $\mathbf{K}\varrho$ in φ by $\kappa_{\mathsf{s}_\sigma}(\mathbf{K}\varrho)$.

▲

The rewrite steps follow the definition of the rewriting operator in Figure 6.1. To rewrite an epistemic value restriction of the form $\forall(\mathbf{K}P).D$ we replace it with $\neg\exists(\mathbf{K}P).\neg D$ and use the transition rule for existential restrictions.

Starting with the pair $(\emptyset, \psi)$ the algorithm performs possible rewrite steps as defined above until a **K**-free Boolean $\mathcal{ALCO}$-KB is reached. There can be more than one possible rewrite step we can perform on one pair. However, it can be shown that all possible choices lead to the same outcome. We write

$$(\emptyset, \psi) \vdash_{\kappa_{\mathsf{s}_\sigma}}^{*} (\mathcal{T}_{\mathsf{a}}, \varphi)$$

to express that $(\mathcal{T}_{\mathsf{a}}, \varphi)$ is obtained from $(\emptyset, \psi)$ via a sequence of rewrite steps. In each step the number of **K** symbols decreases by one. Thus, the number of rewrite steps we need to take to reach the final pair $(\mathcal{T}_{\mathsf{a}}, \varphi)$, where φ is objective, is bounded by the number of **K** symbols occurring in the initial projection query ψ.

Lemma 6.56. *Let Σ_{s}, $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})$, σ, ψ and s_σ be as above. Furthermore, let $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ with $\mathsf{s}_\sigma = \mathsf{s}_\sigma(\mathcal{I})$, $\mathcal{M}$ the epistemic state with $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})}^{\sigma} (\mathcal{I}_\sigma, \mathcal{M})$ and $\mathcal{T}_{\mathsf{a}}$ an acyclic $\mathcal{ALCO}$-TBox and φ a Boolean $\mathcal{ALCO}$-KB with $(\emptyset, \psi) \vdash_{\kappa_{\mathsf{s}_\sigma}}^{*} (\mathcal{T}_{\mathsf{a}}, \varphi)$. For all $\mathcal{J} \in \mathcal{M}$ it holds that*

$$\mathcal{J} \models \varphi_{\mathcal{T}_{\mathsf{a}}} \text{ iff } \mathcal{J} \models [\![\psi, \kappa_{\mathcal{M}}]\!].$$

Proof. Let $(\mathcal{T}_1, \phi_1)$ and $(\mathcal{T}_2, \phi_2)$ be two pairs reached from $(\emptyset, \psi)$ such that

$$(\emptyset, \psi) \vdash_{\kappa_{\mathsf{s}_\sigma}}^{*} (\mathcal{T}_1, \phi_1) \vdash_{\kappa_{\mathsf{s}_\sigma}}^{X} (\mathcal{T}_2, \phi_2)$$

where X is an epistemic concept or KB. For an arbitrary $\mathcal{J} \in \mathcal{M}$ we show that

$$(\mathcal{J}, \mathcal{M}) \mid\!\models (\phi_1)_{\mathcal{T}_1} \text{ iff } (\mathcal{J}, \mathcal{M}) \mid\!\models (\phi_2)_{\mathcal{T}_2}. \tag{6.9}$$

Assume $X = \mathbf{K}D \in \mathsf{sub}(\phi_1)$, where D is an objective concept. We have $\mathcal{T}_1 = \mathcal{T}_2$ and ϕ_2 is obtained from ϕ_1 by replacing each occurrence of $\mathbf{K}D$ in ϕ_1 by $\bigsqcup \kappa_{\mathsf{s}_\sigma}(\mathbf{K}D)$. We show that

$$\kappa_{\mathsf{s}_\sigma}(\mathbf{K}D) = \kappa_{\mathcal{M}}(\mathbf{K}(D_{\mathcal{T}_1})),$$

where $\mathcal{M}$ is the epistemic state as given in the claim. Let $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$ and $|\sigma| = n$. Using the definition of $\kappa_{\mathsf{s}_\sigma}(\mathbf{K}D)$ and Lemma 6.54 it follows that $\{o\} \in \kappa_{\mathsf{s}_\sigma}(\mathbf{K}D)$

iff $\mathcal{K}^\sigma_{\text{red}} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(D,\mathcal{T}_1)} \wedge \mathcal{K}_{\text{sen}}(\mathsf{s}_\sigma) \models o \in T^{(n)}_D$

iff $\mathcal{J} \models \varphi^{(n)}$ with $\varphi = (o \in D)$ for every model $\mathcal{J}$ of $\mathcal{K}^\sigma_{\text{red}} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(D,\mathcal{T}_1)} \wedge \mathcal{K}_{\text{sen}}(\mathsf{s}_\sigma)$

iff $\mathcal{Y} \models o \in D_{\mathcal{T}_1}$ for every $\mathcal{Y} \in \mathcal{M}$ (by Lemma 6.54)

iff $o \in (\mathbf{K}(D_{\mathcal{T}_1}))^{\mathcal{M}}$

iff $\{o\} \in \kappa_{\mathcal{M}}(\mathbf{K}(D_{\mathcal{T}_1}))$.

Using analogous arguments it can be shown that $\neg N \in \kappa_{\mathsf{s}_\sigma}(\mathbf{K}D)$ iff $\neg N \in \kappa_{\mathcal{M}}(\mathbf{K}(D_{\mathcal{T}_1}))$. Since D and $D_{\mathcal{T}_1}$ are objective, it follows that

$$\bigsqcup \kappa_{\mathsf{s}_\sigma}(\mathbf{K}D) = [\![\mathbf{K}(D_{\mathcal{T}_1}), \kappa_{\mathcal{M}}]\!].$$

With Lemma 6.48 we obtain

$$X^{\mathcal{Y},\mathcal{M}}_{\mathcal{T}_1} = (\mathbf{K}(D_{\mathcal{T}_1}))^{\mathcal{Y},\mathcal{M}} = ([\![\mathbf{K}(D_{\mathcal{T}_1}), \kappa_{\mathcal{M}}]\!])^{\mathcal{Y}} = \left(\bigsqcup \kappa_{\mathsf{s}_\sigma}(\mathbf{K}D)\right)^{\mathcal{Y}}.$$

Thus, X and its substitute in ϕ_2 have the same instances under $\mathcal{M}$ w.r.t. $\mathcal{T}_1$. The claim (6.9) follows. The remaining cases where X is of the form $\exists(\mathbf{K}P).D$ or $\mathbf{K}\varrho$ according to Definition 6.55 can be handled in an analogous way.

Let $(\mathcal{T}_\mathsf{a}, \varphi)$ be such that φ is objective and $(\emptyset, \psi) \vdash^*_{\kappa_{\mathsf{s}_\sigma}} (\mathcal{T}_\mathsf{a}, \varphi)$. By induction on the number of rewrite steps and (6.9) it follows that

$$(\mathcal{J}, \mathcal{M}) \mid\!\models \psi \text{ iff } (\mathcal{J}, \mathcal{M}) \mid\!\models \varphi_{\mathcal{T}_\mathsf{a}} \text{ iff } \mathcal{J} \models \varphi_{\mathcal{T}_\mathsf{a}}$$

for all $\mathcal{J} \in \mathcal{M}$. Lemma 6.50 implies $\mathcal{J} \models \varphi_{\mathcal{T}_\mathsf{a}}$ iff $\mathcal{J} \models [\![\psi, \kappa_{\mathcal{M}}]\!]$, for all $\mathcal{J} \in \mathcal{M}$. □

Let $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory, $\sigma \in \boldsymbol{A}^*$ with $|\sigma| = n$ a sequence and ψ a projection query. The *algorithm for deciding the epistemic projection problem*, denoted by

$$\mathsf{proj}(\Sigma_\mathsf{s}, \sigma, \psi),$$

consists of the following steps

1. The set of all consistent sensing results of σ w.r.t. Σ_s is computed: for all sensing results s_σ of σ w.r.t. $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ the Boolean $\mathcal{ALCO}$-KB

$$\mathcal{K}^\sigma_{\text{red}} \wedge \mathcal{K}_{\text{sen}}(\mathsf{s}_\sigma) \tag{6.10}$$

is computed and checked for consistency.

2. For each consistent sensing result s_σ of σ w.r.t. Σ_s the pair $(\mathcal{T}_\mathsf{a}, \varphi)$, where φ is objective and $(\emptyset, \psi) \vdash^*_{\kappa_{\mathsf{s}_\sigma}} (\mathcal{T}_\mathsf{a}, \varphi)$ holds, is computed. For each pair the following entailment check is performed

$$\mathcal{K}^\sigma_{\text{red}} \wedge \mathcal{T}^\sigma_{\mathsf{sub}(\varphi,\mathcal{T}_\mathsf{a})} \wedge \mathcal{K}_{\text{sen}}(\mathsf{s}_\sigma) \models \varphi^{(n)}. \tag{6.11}$$

If for all pairs $(\mathcal{T}_{\mathsf{a}}, \varphi)$ the entailment (6.11) is true, then the answer of the projection problem is "yes", written as $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{True}$, and "no" otherwise, written as $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{False}$.

Lemma 6.57. *Let Σ_{s}, σ and ψ be as above. It holds that* $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{True}$ *iff ψ is valid after doing σ in Σ_{s}.*

Proof. "$\Rightarrow$": Assume $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{True}$. Let $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ be an arbitrary model. We have to show that

$$(\mathcal{I}_\sigma, \mathcal{W}) \Vdash \psi$$

where $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{I}_\sigma, \mathcal{W})$. Let s_σ be the sensing result of σ in $\mathcal{I}$. It follows that $\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma)$ is consistent. Let $(\mathcal{T}_{\mathsf{a}}, \varphi)$ be the pair of an acyclic $\mathcal{ALCO}$-TBox and an objective Boolean $\mathcal{ALCO}$-KB such that $(\emptyset, \psi) \vdash^{*}_{\mathcal{K}_{\mathsf{s}_\sigma}} (\mathcal{T}_{\mathsf{a}}, \varphi)$. Lemma 6.54 implies that there is a model $\mathcal{J}$ of $\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(\varphi, \mathcal{T}_{\mathsf{a}})} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma)$ such that $\mathcal{J} \models \varphi^{(n)}$ iff $\mathcal{I}_\sigma \models \varphi_{\mathcal{T}_{\mathsf{a}}}$. Since we have $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{True}$ by assumption, it holds that

$$\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(\varphi, \mathcal{T}_{\mathsf{a}})} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma) \models \varphi^{(n)}.$$

It follows that $\mathcal{I}_\sigma \models \varphi_{\mathcal{T}_{\mathsf{a}}}$. Lemma 6.56 implies $\mathcal{I}_\sigma \models [\![\psi, \kappa_{\mathcal{W}}]\!]$. Lemma 6.50 implies that $(\mathcal{I}_\sigma, \mathcal{W}) \Vdash \psi$.

"$\Leftarrow$":

Assume ψ is valid after doing σ in Σ_{s}. We have to show that $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{True}$. To the contrary assume $\mathsf{proj}(\Sigma_{\mathsf{s}}, \sigma, \psi) = \textsc{False}$. Thus, there exists a sensing result s_σ of σ w.r.t. Σ_{s} such that $\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma)$ is consistent, but for the pair $(\mathcal{T}_{\mathsf{a}}, \varphi)$, where $(\emptyset, \psi) \vdash^{*}_{\mathcal{K}_{\mathsf{s}_\sigma}} (\mathcal{T}_{\mathsf{a}}, \varphi)$ and φ is objective, it holds that

$$\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(\varphi, \mathcal{T}_{\mathsf{a}})} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma) \not\models \varphi^{(n)}. \tag{6.12}$$

Since $\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma)$ is consistent, Lemma 6.53 implies that s_σ is consistent w.r.t. Σ_{s}, i.e. there exists a model $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ such that s_σ is the sensing result of σ in $\mathcal{I}$. Let $(\mathcal{I}_\sigma, \mathcal{W})$ be the epistemic interpretation satisfying $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{I}_\sigma, \mathcal{W})$. (6.12) implies that there is a model $\mathcal{J} \models \mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{T}^{\sigma}_{\mathsf{sub}(\varphi, \mathcal{T}_{\mathsf{a}})} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma)$ such that $\mathcal{J} \not\models \varphi^{(n)}$. Lemma 6.54 implies that there exists $\mathcal{Y} \in \mathcal{W}$ such that $\mathcal{Y} \not\models \varphi_{\mathcal{T}_{\mathsf{a}}}$. With Lemma 6.56 it follows that $\mathcal{Y} \not\models [\![\psi, \kappa_{\mathcal{W}}]\!]$. Lemma 6.50 yields $(\mathcal{Y}, \mathcal{W}) \not\Vdash \psi$. $\mathcal{Y} \in \mathcal{W}$ implies that there exists a model $\mathcal{Y}_0 \in \mathcal{M}(\mathcal{K})$ such that $(\mathcal{Y}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{Y}, \mathcal{W})$. Thus, $(\mathcal{Y}, \mathcal{W}) \not\Vdash \psi$ is a contradiction to the assumption that ψ is valid after doing σ in Σ_{s}. □

The algorithm described above yields a decision procedure for the projection problem with an ExpTime upper bound. For an action sequence σ of length n there are 2^n possible sensing results. For each sensing result s_σ the Boolean $\mathcal{ALCO}$-KB $\mathcal{K}^{\sigma}_{\mathsf{red}} \wedge \mathcal{K}_{\mathsf{sen}}(\mathsf{s}_\sigma)$ is of polynomial size and the consistency check is in ExpTime (Corollary 6.17). Thus, exponentially many ExpTime checks are required to compute the set of all consistent sensing results. For each consistent sensing result the objective rewriting of the projection query given by $(\mathcal{T}_{\mathsf{a}}, \varphi)$ with $(\emptyset, \psi) \vdash^{*}_{\mathcal{K}_{\mathsf{s}_\sigma}} (\mathcal{T}_{\mathsf{a}}, \varphi)$ can be computed in ExpTime by making polynomially many calls to an ExpTime decision procedure that decides entailment in $\mathcal{ALCO}$: the number of rewrite steps is linearly bounded by the number of $\mathbf{K}$ symbols in the projection query ψ. There is exactly one rewrite step for each **K** in ψ. To rewrite an epistemic concept $|\mathsf{Obj}(\Sigma_{\mathsf{s}})| + 1$ many entailment checks are required, for an epistemic role $(|\mathsf{Obj}(\Sigma_{\mathsf{s}})| + 1) \times (|\mathsf{Obj}(\Sigma_{\mathsf{s}})| + 1)$ many and for an epistemic KB only one entailment check is needed.

The matching lower bound comes from the ExpTime-hard consistency problem of $\mathcal{ALCO}$-KBs under the SNA (see 6.17).

Theorem 6.58. *Projection in epistemic $\mathcal{ALCO}$-action theories is* ExpTime*-complete.*

6.5 Summary and Related Work

We have introduced an action formalism based on the DL $\mathcal{ALCO}$ with sensing actions. It can be viewed as a fragment of the epistemic Situation Calculus $\mathcal{ES}$ [LL04; LL11]. We have shown that the projection problem is ExpTime-complete. Thus, it is not harder than projection in the non-epistemic case. The epistemic DL $\mathcal{ALCOK}$ we have chosen to formulate projection queries is quite expressive. The knowledge modality can be applied to axioms, concepts and also to roles. Due to the semantics that guarantees a unique epistemic model of the initial knowledge base, it is possible to use a similar reduction to non-modal reasoning as the one presented in [LL04] within the chosen DL $\mathcal{ALCO}$.

Another DL-based action formalism with sensing has been introduced in [De +97]. It uses the epistemic DL $\mathcal{ALCK}$ to axiomatize action effects by viewing certain role names as actions, that relate a starting state to an end state. However, the frame problem is not solved in this formalism.

Another decidable fragment of the epistemic Situation Calculus has been obtained in [LL14]. The fragment is not based on syntactic restrictions as ours. Decidability is achieved by using a weaker inference mechanism.

Chapter 7

Verification of Knowledge-Based Programs

In this chapter we investigate the computational properties of temporal verification of ConGolog programs over actions defined in an epistemic $\mathcal{ALCO}$-action theory and with subjective tests formulated as Boolean $\mathcal{ALCOK}$-KBs. As temporal specification language we consider CTL* over Boolean $\mathcal{ALCOK}$-KBs.

Example 7.1. As an example we consider the control program of an agent whose task is to identify and repair faults of a device named dev. The goal is to safely turn dev on after identifying and repairing the faults of dev. The basic abilities of the agent are described in the epistemic $\mathcal{ALCO}$-action theory in Example 6.20. In addition there is the purely physical ground action `raise-alarm` that just sets a notification flag to true and is defined as a local effect action. The action `skip` does not cause any changes. The program in Figure 7.1

```
while $\mathsf{dev} \sqsubseteq \exists(\mathbf{K}\mathit{HasFault}).(\mathbf{K}\mathit{Fault})$ do
   $\mathsf{pick}(x) \rightarrow \mathbf{K}((\mathsf{dev}, x) \sqsubseteq \mathit{HasFault})$?; $\mathtt{repair}(\mathsf{dev}, x)$;
end;
while $\neg\mathbf{K}(\mathsf{dev} \sqsubseteq \forall \mathit{HasFault}.\neg\mathbf{K}\mathit{Fault})$ do
   $\mathsf{pick}(x) \rightarrow \mathbf{K}(x \sqsubseteq \mathit{Fault}) \wedge \neg\mathbf{Kw}((\mathsf{dev}, x) \sqsubseteq \mathit{HasFault})$?;
      $\mathtt{sense\text{-}fault}(\mathsf{dev}, x)$;
      if $\mathbf{K}((\mathsf{dev}, x) \sqsubseteq \mathit{HasFault})$ then $\mathtt{repair}(\mathsf{dev}, x)$ else $\mathtt{skip}$;
end;
$\mathtt{turn\text{-}on}(\mathsf{dev})$; $\mathtt{sense\text{-}on}(\mathsf{dev})$;
if $\mathbf{K}(\mathsf{dev} \sqsubseteq \neg \mathit{On})$ then $\mathtt{raise\text{-}alarm}$ else $\mathtt{skip}$;
```

Figure 7.1: Program for fault detection and repair

describes the following behavior: with the first while-loop the agent repairs one by one in an arbitrary order all the faults dev is known to have. The second loop deals with all those known faults for which it is unknown whether dev has them or not. As long as the agent does not know that dev has no known fault, a known fault x is chosen non-deterministically for which it is unknown whether dev has it or not. The agent then senses whether dev has this fault and repairs it if necessary. After completing the second loop the agent turns the device on and checks if this was successful and if not an alarm is raised.

Thus, with this program we also want to achieve that the agent is aware of the fact that its knowledge about faults is incomplete and that its abilities to identify and repair them are limited. The following specifications describe some desirable properties of the program

- the program always terminates;
- eventually it is known that dev is on or it is known that dev has an unknown critical fault;
- the TBox of the underlying action theory is always known, i.e. the TBox is always part the agent's knowledge state

▲

The remainder of this chapter is organized as follows. In Section 7.1, we formally define knowledge-based programs and the verification problem for (possibly epistemic) temporal properties. In Section 7.2, we investigate the complexity of the verification problem for programs over unconditional ground actions. In case an action does not have conditional effects the outcome is immediately observable to the agent because the action causes the same changes in every possible world. Under this restriction we single out two fragments where the verification of subjective temporal properties is ExpTime-complete. This lowers the complexity by one exponential compared to the verification problem for $\mathcal{ALCO}$-ConGolog over local effect actions (Theorem 4.18). Decidability of the verification problem for programs over ground actions with conditional effects is shown in Section 7.3. In Section 7.4, we revisit the guarded pick operator. We are able to identify restrictions on the guards such that the verification problem becomes decidable. A summary of the results can be found in Section 7.5.

7.1 Knowledge-Based Programs and Temporal Properties

First, we define syntax and semantics of knowledge-based $\mathcal{ALCOK}$-ConGolog programs.

Syntax and Semantics of $\mathcal{ALCOK}$-ConGolog

We define programs over actions defined in an epistemic $\mathcal{ALCO}$-action theory and with tests formulated as *subjective* Boolean $\mathcal{ALCOK}$-KBs. The programming constructs are the same as for ConGolog programs over FO-DSs (Definition 2.41).

Definition 7.2. Let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory and $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}, \sim_{\mathsf{s}})$ the induced epistemic FO-DS. A *program expression δ over $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$* is built according to the following grammar:

$$\delta ::= \langle\rangle \mid \alpha(\bar{t}) \mid \psi? \mid (\delta;\delta) \mid (\delta|\delta) \mid (\delta\|\delta) \mid \mathsf{pick}(\bar{x}) \rightarrow \psi?;\delta,$$

where $\langle\rangle$ is the empty program, $\alpha(\bar{t}) \in \mathsf{Act}$ is an action term, ψ stands for a subjective Boolean $\mathcal{ALCOK}$-KB over concept names and role names from $\mathcal{F}$ and $\bar{x}$ is a tuple of variable names. δ is called *pick-free* if no guarded pick expressions occur in δ, and δ is called *closed* if all variable names are bound by a guarded pick.

A *knowledge-based $\mathcal{ALCOK}$-ConGolog program* is of the form

$$\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta),$$

where Σ_{s} is an epistemic $\mathcal{ALCO}$-action theory and δ a closed program expression over $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$. ▲

The definition of the program semantics is analogous to the semantics of ConGolog over FO-DSs with the only difference that program states now contain an epistemic interpretation instead of only an ordinary FO interpretation. Ground actions are only executed if they are known to be possible, i.e. the preconditions are satisfied in all possible worlds of the knowledge state. Together with the restriction to subjective tests we achieve that the choice of the next action only depends on the current knowledge state of the program state.

Definition 7.3. Let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory and $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \mathsf{Act}, \mathcal{E}, \mathsf{Pre}, \sim_{\mathsf{s}})$ the induced epistemic FO-DS.

A *program state over* $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$ is a tuple of the form

$$\langle (\mathcal{I}, \mathcal{W}), \sigma, \rho \rangle, \text{ where}$$

$(\mathcal{I}, \mathcal{W})$ is an epistemic interpretation, $\sigma \in \mathsf{ground}(\mathsf{Act})^*$ a ground action sequence and ρ a closed program expression over $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$.

The *set of all program states over* $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$ is denoted by $\mathsf{States}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$.

The set $\mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$ denotes the *set of all final program states over* $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$ and is defined by induction on the size of program expressions as the smallest set satisfying the following conditions:

1. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \langle\rangle \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$;
2. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \psi? \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$, if $(\mathcal{I}, \mathcal{W}) \mid\models \psi$;
3. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta^* \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$;
4. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_1; \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$,
   if $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$ and $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$;
5. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_1 | \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$,
   if $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$ or $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$;
6. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_1 \| \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$,
   if $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_1 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$ and $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta_2 \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$;
7. $\langle (\mathcal{I}, \mathcal{W}), \sigma, \mathsf{pick}(\bar{x}) \to \psi?; \delta \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$,
   if $(\mathcal{I}, \mathcal{W}) \mid\models \psi^{\nu}$ and $\langle (\mathcal{I}, \mathcal{W}), \sigma, \delta^{\nu} \rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$ for some ν.

A *transition relation*

$$\to_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})} \subseteq \mathsf{States}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})) \times \mathsf{States}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}))$$

is defined by induction on the size of program expressions as the smallest set satisfying the following conditions:

1. $\langle(\mathcal{I},\mathcal{W}),\sigma,\boldsymbol{\alpha}\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\langle\rangle\rangle$,
   if $(\mathcal{I},\mathcal{W}) \Longrightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}_{\mathbf{K}}} (\mathcal{I}',\mathcal{W}')$ and $\mathcal{J} \succ_{\mathsf{poss}} \boldsymbol{\alpha}$ for all $\mathcal{J}\in\mathcal{W}$;
2. $\langle\mathcal{I},\mathcal{W},\sigma,\delta^*\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta';\delta^*\rangle$,
   if $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta'\rangle$;
3. $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1;\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_1';\delta_2\rangle$,
   if $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_1'\rangle$;
4. $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1;\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_2'\rangle$,
   if $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1\rangle \in \mathsf{Final}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}}))$ and $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_2'\rangle$;
5. $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1|\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta'\rangle$,
   if $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma,\delta'\rangle$ or $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma,\delta'\rangle$
6. $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1\|\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_1'\|\delta_2\rangle$,
   if $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_1'\rangle$;
7. $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_1\|\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_1\|\delta_2'\rangle$,
   if $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta_2\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta_2'\rangle$;
8. $\langle(\mathcal{I},\mathcal{W}),\sigma,\mathsf{pick}(\vec{x}) \to \psi?;\delta\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta'\rangle$,
   if $\mathcal{I},\mathcal{W} \mid\!\models \psi^{\nu}$ and$\langle(\mathcal{I},\mathcal{W}),\sigma,\delta^{\nu}\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma\cdot\boldsymbol{\alpha},\delta'\rangle$ for some ν.

The *set of all failure states over* $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})$, denoted by $\mathsf{Fail}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}}))$, is defined as follows:

$$\mathsf{Fail}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})) := \{\langle(\mathcal{I},\mathcal{W}),\sigma,\delta\rangle \in \mathsf{States}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})) \mid \text{ (i) and (ii)}\}$$

(i) $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta\rangle \notin \mathsf{Final}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}}))$

(ii) there is no $q \in \mathsf{States}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}}))$ with $\langle(\mathcal{I},\mathcal{W}),\sigma,\delta\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} q$.

▲

The next program state via $\to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})}$ is reached by executing the next ground action that is *known* to be possible.

Lemma 7.4. *Let* $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle$ *and* $\langle\mathcal{I}',\mathcal{W}',\sigma',\rho'\rangle$ *be two program states over the epistemic FO-DS* $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}}) = (\mathcal{M}(\mathcal{K}),\mathcal{F},\mathsf{Act},\mathcal{E},\mathsf{Pre},\sim_{\mathrm{s}})$*. It holds that if*

$$\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \to_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathrm{s}})} \langle(\mathcal{I}',\mathcal{W}'),\sigma',\rho'\rangle,$$

then there exists $\boldsymbol{\alpha} \in \mathsf{ground}(\mathsf{Act})$ *such that*

$$\sigma' = \sigma\cdot\boldsymbol{\alpha} \textit{ and } \mathcal{J} \succ_{\mathsf{poss}} \boldsymbol{\alpha} \textit{ for all } \mathcal{J}\in\mathcal{W}.$$

The handling of final and failure states is done in the same way as in the non-epistemic case. The *termination and failure extension* of $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$, denoted by

$$\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) \uplus \{\epsilon, \mathfrak{f}\},$$

is defined as in Definition 2.45 regarding $\mathcal{E}$ and $\succ_{\mathsf{poss}}$, and the termination and failure actions do not have sensing results, i.e. we have $\mathcal{I} \sim_{\mathsf{s}}^{\epsilon} \mathcal{J}$ and $\mathcal{I} \sim_{\mathsf{s}}^{\mathfrak{f}} \mathcal{J}$ for all interpretations $\mathcal{I}$ and $\mathcal{J}$. Both actions are definable as local effect actions in an extended epistemic $\mathcal{ALCO}$-action theory, denoted by $\Sigma_{\mathsf{s}} \uplus \{\epsilon, \mathfrak{f}\}$, that is obtained from Σ_{s} by adding the literals $(\mathsf{prog} \sqsubseteq \neg \mathit{Final})$ and $(\mathsf{prog} \sqsubseteq \neg \mathit{Fail})$ to the initial KB and by defining preconditions, effects and sensing results of ϵ and $\mathfrak{f}$ as follows:

$$\begin{aligned}
&\mathsf{pre}(\epsilon) = \mathsf{pre}(\mathfrak{f}) := \emptyset,\\
&\mathsf{eff}(\epsilon) := \big\{\langle \mathit{Final}, \{\mathsf{prog}\}\rangle^{+}\big\}, \mathsf{eff}(\mathfrak{f}) := \big\{\langle \mathit{Fail}, \{\mathsf{prog}\}\rangle^{+}\big\},\\
&\mathsf{sense}(\epsilon) = \mathsf{sense}(\mathfrak{f}) := \textsc{True}.
\end{aligned}$$

It holds that

$$\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) \uplus \{\epsilon, \mathfrak{f}\} = \mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}} \uplus \{\epsilon, \mathfrak{f}\}).$$

Using the epistemic FO-DS $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}} \uplus \{\epsilon, \mathfrak{f}\})$ we are now ready to define the transition system of a knowledge-based program. It is a straightforward adaption of the definition for ordinary ConGolog programs over FO-DSs.

Definition 7.5. Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program. We write $\Sigma_{\mathsf{s}}^{\mathsf{e}}$ as abbreviation of $\Sigma_{\mathsf{s}} \uplus \{\epsilon, \mathfrak{f}\}$ and write $\mathcal{K}^{\mathsf{e}}$ to denote the initial KB of the extended action theory $\Sigma_{\mathsf{s}}^{\mathsf{e}}$.

The *transition system induced by* $\mathcal{P}$, denoted by

$$\mathfrak{I}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}}),$$

consists of

- the set of *states*

$$Q_{\mathcal{P}} := \{\langle (\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \in \mathsf{States}(\Sigma_{\mathsf{s}}^{\mathsf{e}}) \mid \text{ the symbols } \epsilon, \mathfrak{f}, \mathit{Final}, \mathit{Fail} \text{ do not occur in } \rho\};$$

- the set of *initial states* given by

$$I_{\mathcal{P}} := \{\langle (\mathcal{I}, \mathcal{W}), \langle\rangle, \delta\rangle \mid \mathcal{W} = \mathcal{M}(\mathcal{K}^{\mathsf{e}})\}, \text{ and}$$

- the *transition relation* $\hookrightarrow_{\mathcal{P}} \subseteq Q_{\mathcal{P}} \times Q_{\mathcal{P}}$ that is defined as the smallest set satisfying the following conditions:
  - $\langle (\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \hookrightarrow_{\mathcal{P}} \langle (\mathcal{J}, \mathcal{M}), \sigma \cdot \boldsymbol{\alpha}, \xi\rangle$,
    if $\langle (\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \rightarrow_{\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}^{\mathsf{e}})} \langle (\mathcal{J}, \mathcal{M}), \sigma \cdot \boldsymbol{\alpha}, \xi\rangle$;
  - $\langle (\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \hookrightarrow_{\mathcal{P}} \langle (\mathcal{J}, \mathcal{M}), \sigma \cdot \epsilon, \langle\rangle\rangle$,
    if $\langle (\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \in \mathsf{Final}(\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}^{\mathsf{e}}))$ and $(\mathcal{I}, \mathcal{W}) \Longrightarrow_{\mathfrak{D}_{\mathsf{K}}}^{\epsilon} (\mathcal{J}, \mathcal{M})$;

- $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \hookrightarrow_{\mathcal{P}} \langle(\mathcal{J},\mathcal{M}),\sigma\cdot\mathfrak{f},\rho\rangle$,
  if $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \in \mathsf{Fail}(\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}^{\mathsf{e}}))$ and $(\mathcal{I},\mathcal{W}) \Longrightarrow^{\mathfrak{f}}_{\mathfrak{D}_{\mathbf{K}}} (\mathcal{J},\mathcal{M})$.

- the *labeling function* $\lambda_{\mathcal{P}} : \langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \mapsto (\mathcal{I},\mathcal{W})$ for each $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \in Q_{\mathcal{P}}$.

▲

The partition of the set of all infinite paths in the transition system into non-terminating and non-failing executions, terminating executions and failing executions (Definition 2.47 and Lemma 3.2) is carried over to the epistemic case in the obvious way. Note that for all states $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \in Q_{\mathfrak{T}}$ reachable from an initial state it holds that

$$\mathcal{I} \models \mathsf{prog} \sqsubseteq \mathit{Final} \text{ iff } (\mathcal{I},\mathcal{W}) \mid\!\models \mathbf{K}(\mathsf{prog} \sqsubseteq \mathit{Final}) \text{ and}$$
$$\mathcal{I} \models \mathsf{prog} \sqsubseteq \mathit{Fail} \text{ iff } (\mathcal{I},\mathcal{W}) \mid\!\models \mathbf{K}(\mathsf{prog} \sqsubseteq \mathit{Fail}).$$

Verification Problem

To specify the correctness of knowledge-based $\mathcal{ALCOK}$-ConGolog programs we define the temporal logic called $\mathcal{ALCOK}$-CTL*.

The *syntax* of *$\mathcal{ALCOK}$-CTL* state formulas* and *$\mathcal{ALCOK}$-CTL* path formulas* is defined in the same way as for FO-CTL* (Definition 2.37) but with Boolean $\mathcal{ALCOK}$-KBs in place of FO sentences.

An $\mathcal{ALCOK}$-CTL* state or path formula is called *subjective,* if all Boolean $\mathcal{ALCOK}$-KBs occurring in it are subjective, and *objective* if all KBs are objective.

The *semantics* is defined in terms of transition systems where states are labeled with epistemic interpretations. Let $\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$ be a transition system, where each state is labeled with an epistemic interpretation, and let $q \in Q_{\mathfrak{T}}$ be a state and π a path in $\mathfrak{T}$.

Satisfaction of an $\mathcal{ALCOK}$-CTL state formula Φ in $\mathfrak{T}, q$,* denoted by $\mathfrak{T}, q \models \Phi$, and *satisfaction of an $\mathcal{ALCOK}$-CTL* path formula* Ψ in $\mathfrak{T}, \pi$, denoted $\mathfrak{T}, \pi \models \Psi$, is defined as in Definition 2.38 except for the atomic case where the state formula is of the form $\Phi = \psi$ for some Boolean $\mathcal{ALCOK}$-KB ψ. In this case we have

$$\mathfrak{T}, q \models \psi \text{ iff } (\mathcal{I},\mathcal{W}) \mid\!\models \psi \text{ where } (\mathcal{I},\mathcal{W}) = \lambda_{\mathfrak{T}}(\mathrm{q}).$$

Definition 7.6 (verification problem). Let $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program and Φ an $\mathcal{ALCOK}$-CTL* state formula. We say that Φ *is valid in* $\mathcal{P}$ iff $\mathfrak{T}_{\mathcal{P}}, q_0 \models \Phi$ holds for all initial states $q_0 \in I_{\mathcal{P}}$ of the transition system $\mathfrak{T}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ induced by $\mathcal{P}$. Φ *is said to be satisfiable in* $\mathcal{P}$ iff $\mathfrak{T}_{\mathcal{P}}, q_0 \models \Phi$ for some initial state $q_0 \in I_{\mathcal{P}}$.

The *verification problem* is the problem of determining whether or not Φ is valid in $\mathcal{P}$. ▲

For instance, it can be expressed that the TBox $\mathcal{T}$ (considered as a conjunction of concept inclusions) of the underlying action theories is always known: $\mathsf{AG}(\mathbf{K}\mathcal{T})$. This means the TBox is always part of the knowledge state of the agent and is available to evaluate the tests in the program. The following subjective formula describes a desirable outcome of the program in Example 7.1:

$$\mathsf{AF}(\mathbf{K}(\mathsf{dev} \sqsubseteq \mathit{On}) \vee \mathbf{K}(\mathsf{dev} \sqsubseteq (\exists \mathit{HasFault}.(\mathit{CriticalFault} \sqcap \neg\mathbf{K}\mathit{Fault})))).$$

Bisimulation and Abstraction for Programs over Ground Actions

We consider knowledge-based $\mathcal{ALCOK}$-ConGolog programs $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_s), \delta)$, where δ is pick-free and Σ_s is an epistemic $\mathcal{ALCO}$-action theory for a finite set of ground actions. For such programs we define *propositional abstractions* based on context-bisimulations in the same way as for ordinary ConGolog programs.

The definition of a *guarded action* (Definition 3.8) over $\mathfrak{D}_{\mathbf{K}}(\Sigma_s \uplus \{\epsilon, \mathfrak{f}\})$ is carried over from the non-epistemic case in the obvious way. Let $\Sigma_s = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$,

$$\mathfrak{a} = \psi_1?; \cdots ; \psi_n?; \boldsymbol{\alpha}$$

a guarded action over $\mathfrak{D}_{\mathbf{K}}(\Sigma_s \uplus \{\epsilon, \mathfrak{f}\})$ for some $n \geq 0$ and let $(\mathcal{I}, \mathcal{W})$ be an epistemic interpretation. We say that $\mathfrak{a}$ *is executable in* $(\mathcal{I}, \mathcal{W})$ iff

$$(\mathcal{I}, \mathcal{W}) \models \psi_i \text{ for all } i = 1, \ldots, n \quad \text{and } \mathcal{J} \models \mathsf{pre}(\boldsymbol{\alpha}) \text{ for all } \mathcal{J} \in \mathcal{W}.$$

Let δ be a pick-free program expression over $\mathfrak{D}_{\mathbf{K}}(\Sigma_s)$. The functions $\mathsf{head}(\cdot)$, $\mathsf{tail}(\cdot,\cdot)$ and the *set of all reachable sub-programs* $\mathsf{sub}(\delta)$ are defined analogously to Definition 3.9, 3.10 and 3.11. Lemma 3.12 and Theorem 3.15 are true for the epistemic case as well. We leave this without a proof.

We define static types of epistemic interpretations. Let $(\mathcal{I}, \mathcal{W})$ be an epistemic interpretation and $\mathcal{C}$ an $\mathcal{ALCOK}$-context. The *static type of* $(\mathcal{I}, \mathcal{W})$ *w.r.t.* $\mathcal{C}$, denoted by $\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W})$, is given by

$$\mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W}) := \{\psi \in \mathcal{C} \mid (\mathcal{I}, \mathcal{W}) \Vdash \psi\}.$$

The *set of all static types w.r.t.* $\mathcal{C}$, denoted by $\mathfrak{S}_{\mathcal{C}}$, is given by

$$\mathfrak{S}_{\mathcal{C}} := \{\mathfrak{s} \subseteq \mathcal{C} \mid \text{ there exists an } (\mathcal{I}, \mathcal{W}) \text{ with } \mathfrak{s} = \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W})\}.$$

Next, we define the relevant context of a knowledge-based program.

Definition 7.7. Let $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_s), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program, where δ is pick-free and $\Sigma_s = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ with $\boldsymbol{A}$ being a finite set of ground actions. The *relevant* $\mathcal{ALCOK}$*-context* $\mathcal{C}_{\mathcal{P}}$ for $\mathcal{P}$ is given by

$$\begin{aligned}
\mathcal{C}_{\mathcal{P}} := \{\psi, \neg\psi \mid\ & \psi \rhd \langle F, \{\bar{o}\}\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}) \text{ for some } \boldsymbol{\alpha} \in \boldsymbol{A}, \text{ or} \\
& \psi = \mathsf{sense}(\boldsymbol{\alpha}) \text{ for some } \boldsymbol{\alpha} \in \boldsymbol{A}, \text{ or} \\
& \psi = \mathbf{K}\big(\textstyle\bigwedge \mathsf{pre}(\boldsymbol{\alpha})\big) \text{ for some } \boldsymbol{\alpha} \in \boldsymbol{A}, \text{ or} \\
& \psi \text{ is an axiom in } \mathcal{K}, \text{ or} \\
& \psi? \text{ is a test occurring in } \delta, \text{ or} \\
& \psi = \mathbf{K}(\mathsf{prog} \sqsubseteq \mathit{Final}) \text{ or } \psi = \mathbf{K}(\mathsf{prog} \sqsubseteq \mathit{Fail})\}.
\end{aligned}$$

An $\mathcal{ALCOK}$-context $\mathcal{C}$ is called *a proper context for* $\mathcal{P}$ iff $\mathcal{C}_{\mathcal{P}} \subseteq \mathcal{C}$. ▲

Executability of a guarded action can be defined on the level of static types.

Definition 7.8. Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be as in Definition 7.7, $\mathcal{C}$ a proper context for $\mathcal{P}$, $\mathcal{C}_{\mathsf{K}}$ the *subjective sub-context of* $\mathcal{C}$ with

$$\mathcal{C}_{\mathsf{K}} := \{\psi \in \mathcal{C} \mid \psi \text{ is subjective}\},$$

and let $\mathfrak{s} \in \mathfrak{S}_{\mathcal{C}_{\mathsf{K}}}$ be a static type and $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $\rho \in \mathsf{sub}(\delta)$ a guarded action.

We say that $\mathfrak{a}$ *is executable in* $\mathfrak{s}$ iff $\psi_i \in \mathfrak{s}$ for all $i = 1, \ldots, n$ and $\mathsf{K}\big(\bigwedge \mathsf{pre}(\boldsymbol{\alpha})\big) \in \mathfrak{s}$. ▲

The following lemma is a direct consequence of the definitions above. It is the epistemic counterpart of Lemma 3.20.

Lemma 7.9. *Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be as in Definition 7.7, $\mathcal{C}_{\mathsf{K}}$ the subjective sub-context of a proper context $\mathcal{C}$ for $\mathcal{P}$, $(\mathcal{I}, \mathcal{W})$ an epistemic interpretation and $\mathfrak{a} \in \mathsf{head}(\rho)$ for some $\rho \in \mathsf{sub}(\delta)$. It holds that $\mathfrak{a}$ is executable in $(\mathcal{I}, \mathcal{W})$ iff it is executable in $\mathsf{s\text{-}type}_{\mathcal{C}_{\mathsf{K}}}(\mathcal{I}, \mathcal{W})$.*

The definition of a $\mathcal{C}$-bisimulation extends naturally to $\mathcal{ALCOK}$-contexts and transition systems where states are labeled with epistemic interpretations.

Definition 7.10. Let $\mathcal{C}$ be an $\mathcal{ALCOK}$-context, AP a finite set of atomic propositions such that a bijection $\iota_{\mathcal{C}} : \mathcal{C} \to \mathsf{AP}$ between $\mathcal{C}$ and AP exists, and let $\mathfrak{I} = (Q_{\mathfrak{I}}, I_{\mathfrak{I}}, \hookrightarrow_{\mathfrak{I}}, \lambda_{\mathfrak{I}})$ be a transition system, where $\lambda_{\mathfrak{I}}$ labels each state with an epistemic interpretation, and let

$$\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$$

be a propositional transition system over AP. A binary relation $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ is called *$\mathcal{C}$-bisimulation* iff the following conditions are satisfied:

- $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ implies $\lambda_{\mathfrak{T}}(q_{\mathfrak{T}}) = \{\iota_{\mathcal{C}}(\psi) \mid \psi \in \mathsf{s\text{-}type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W})\}$, where $\lambda_{\mathfrak{I}}(q_{\mathfrak{I}}) = (\mathcal{I}, \mathcal{W})$.
- If $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and there is a transition $q_{\mathfrak{I}} \hookrightarrow_{\mathfrak{I}} q'_{\mathfrak{I}}$, then there exists a transition $q_{\mathfrak{T}} \hookrightarrow_{\mathfrak{T}} q'_{\mathfrak{T}}$ such that $q'_{\mathfrak{I}} \simeq_{\mathcal{C}} q'_{\mathfrak{T}}$.
- If $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and there is a transition $q_{\mathfrak{T}} \hookrightarrow_{\mathfrak{T}} q'_{\mathfrak{T}}$, then there exists a transition $q_{\mathfrak{I}} \hookrightarrow_{\mathfrak{I}} q'_{\mathfrak{I}}$ such that $q'_{\mathfrak{I}} \simeq_{\mathcal{C}} q'_{\mathfrak{T}}$.

We say that $\mathfrak{I}$ and $\mathfrak{T}$ are *$\mathcal{C}$-bisimilar* iff there exists a $\mathcal{C}$-bisimulation $\simeq_{\mathcal{C}} \subseteq Q_{\mathfrak{I}} \times Q_{\mathfrak{T}}$ such that

- for all $q_{\mathfrak{I}} \in I_{\mathfrak{I}}$ there exists $q_{\mathfrak{T}} \in I_{\mathfrak{T}}$ such that $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$ and
- for all $q_{\mathfrak{T}} \in I_{\mathfrak{T}}$ there exists $q_{\mathfrak{I}} \in I_{\mathfrak{I}}$ such that $q_{\mathfrak{I}} \simeq_{\mathcal{C}} q_{\mathfrak{T}}$.

▲

We are now ready to define the notion of a propositional abstraction of a knowledge-based program that preserves temporal properties.

Definition 7.11. Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be as in Definition 7.7, $\mathcal{C}$ a proper context for $\mathcal{P}$, and AP a finite set of atomic propositions and $\mathfrak{T} = (Q_{\mathfrak{T}}, I_{\mathfrak{T}}, \hookrightarrow_{\mathfrak{T}}, \lambda_{\mathfrak{T}})$ a propositional transition system over AP.

We say that $\mathfrak{T}$ is a *bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. $\mathcal{C}$* iff the following conditions are satisfied

- There exists a bijection $\iota_{\mathcal{C}} : \mathcal{C} \to \mathsf{AP}$ between $\mathcal{C}$ and AP.
- The transition system $\mathfrak{I}_{\mathcal{P}}$ induced by $\mathcal{P}$ and the propositional transition system $\mathfrak{T}$ are $\mathcal{C}$-bisimilar.

▲

We have that $\mathcal{C}$-bisimilar transition systems are indistinguishable w.r.t. the valid temporal formulas. This is analogous to Lemma 3.24.

Lemma 7.12. *Let $\mathcal{C}$ be an $\mathcal{ALCOK}$-context, $\mathfrak{I}$ a transition system where states are labeled with epistemic interpretations, $\mathfrak{T}$ a propositional transition system over a finite set of atomic propositions* AP *with a bijection $\iota_{\mathcal{C}} : \mathcal{C} \to \mathsf{AP}$ such that $\mathfrak{I}$ and $\mathfrak{T}$ are $\mathcal{C}$-bisimilar.*

For an $\mathcal{ALCOK}$-CTL* state formula Φ over Boolean KBs from $\mathcal{C}$ it holds that

$$\Phi \textit{ is valid in } \mathfrak{I} \textit{ iff } \iota_{\mathcal{C}}(\Phi) \textit{ is valid in } \mathfrak{T},$$

where $\iota_{\mathcal{C}}(\Phi)$ is the propositional CTL* state formula obtained from Φ by replacing all Boolean KBs occurring in Φ by their image in $\iota_{\mathcal{C}}$.

7.2 Programs over Unconditional Ground Actions

In this section we investigate the complexity of the verification problem for knowledge-based $\mathcal{ALCOK}$-ConGolog programs over ground actions with only unconditional effects.

We call $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}), \delta)$ a *knowledge-based $\mathcal{ALCOK}$-ConGolog program over unconditional ground actions* iff it satisfies the following restrictions:

- $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ is an epistemic $\mathcal{ALCO}$-action theory, where $\boldsymbol{A}$ is a finite set of ground actions, and for all $\boldsymbol{\alpha} \in \boldsymbol{A}$ the effects in $\mathsf{eff}(\boldsymbol{\alpha})$ are *unconditional* local effects of the form
$$\left\langle P, \{(o, o')\}\right\rangle^{\pm} \text{ or } \langle A, \{o\}\rangle^{\pm};$$
- δ is a *pick-free* program expression over $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})$.

The size of a program $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}), \delta)$ is given by the size of Σ_{s} and the size of δ. The hardness proof for $\mathcal{ALCO}$-ConGolog based on a local effect $\mathcal{ALCO}$-action theory (Section 4.4) also works in this case.

Corollary 7.13. *The verification problem for $\mathcal{ALCOK}$-CTL* properties and knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions without sensing is* 2EXPTIME-*hard.*

Proof. The reduction in Section 4.4 of the consistency problem of an $\mathcal{ALCO}$-KB with nominal schemas w.r.t. a finite set of object names to the verification problem works without any modifications. We only need to extend the non-epistemic $\mathcal{ALCO}$-action theory

$$\Sigma_{\mathcal{S},\mathsf{Obj}} = (\mathcal{K}_{\mathcal{S},\mathsf{Obj}}, \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}, \mathsf{eff}, \mathsf{pre})$$

with sensing results. We define $\mathsf{sense}(\boldsymbol{\alpha}) := \textsc{True}$ for all $\boldsymbol{\alpha} \in \boldsymbol{A}_{\mathcal{S},\mathsf{Obj}}$. Thus, all actions are purely physical ones and do not provide any additional observations. With this extension the action theory $\Sigma_{\mathcal{S},\mathsf{Obj}}$, the program expression $\delta_{\mathcal{S},\mathsf{Obj}}$ and the temporal property $\Phi_{\mathcal{S},\mathsf{Obj}}$ defined in Section 4.4 satisfy all the above mentioned restrictions. □

If we allow sensing but restrict the temporal property to a subjective one, then we still have 2EXPTIME-hardness.

Corollary 7.14. *The verification problem for subjective $\mathcal{ALCOK}$-CTL* temporal properties and knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions is* 2EXPTIME-*hard.*

Proof. Again, the reduction given in Section 4.4 only needs to be slightly modified. We define $\mathsf{sense}(\boldsymbol{\alpha}) := \textsc{True}$ for all $\boldsymbol{\alpha} \in \{\boldsymbol{A}_{\mathcal{S},\mathsf{Obj}} \setminus \{\mathtt{finished}(s)\}\}$ and

$$\mathsf{sense}(\mathtt{finished}(s)) := \bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D}.$$

Now we can use the following subjective temporal property for the reduction:

$$\Phi_{\mathcal{S},\mathsf{Obj}} := \mathsf{AG}\left(\mathbf{K}(s \,\dot{\in}\, AllGrounded) \rightarrow \mathbf{K}\left(\bigwedge_{C \sqsubseteq D \in \mathcal{S}} \widehat{C} \sqsubseteq \widehat{D}\right)\right).$$

□

7.2.1 Deciding the Verification Problem with Sensing

Next, we show that the problem is in 2EXPTIME. For simplicity we only consider temporal properties over Boolean $\mathcal{ALCOK}$-KBs that are either objective or subjective. In the following we consider a proper $\mathcal{ALCOK}$-context $\mathcal{C}$ for the given program $\mathcal{P}$ that can be partitioned into a subjective and objective part:

$$\mathcal{C} = \mathcal{C}_{\mathbf{K}} \uplus \mathcal{C}_{\mathsf{o}},$$

where $\mathcal{C}_{\mathbf{K}}$ consists of all subjective Boolean $\mathcal{ALCOK}$-KBs and $\mathcal{C}_{\mathsf{o}}$ consists of all relevant objective Boolean $\mathcal{ALCO}$-KBs. The abstraction is based on the *set of all relevant local effects* of actions in $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ given by

$$\mathsf{Lit}(\Sigma_{\mathsf{s}}) := \left(\bigcup_{\boldsymbol{\alpha} \in \boldsymbol{A}} \mathsf{eff}(\boldsymbol{\alpha})\right) \cup \left\{\langle Final, \{\mathsf{prog}\}\rangle^{+}, \langle Fail, \{\mathsf{prog}\}\rangle^{+}\right\}.$$

With $\mathsf{D\text{-}Types}(\mathcal{C}_{\mathsf{o}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$ we denote the *set of all dynamic types w.r.t.* $\mathcal{C}_{\mathsf{o}}$ *and* $\mathsf{Lit}(\Sigma_{\mathsf{s}})$ according to Definition 4.7. An *abstraction* of a program state consists of the following components:

- a dynamic type $\mathfrak{t} \in \mathsf{D\text{-}Types}(\mathcal{C}_{\mathsf{o}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$ as an abstraction of the initial model describing the environment the program is executed in;
- a subset $\mathfrak{t}_{\mathsf{s}} \subseteq \mathfrak{t}$ describing the part of the environment that was observed so far by means of sensing;

- a set of local effects $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ as the accumulated effects of the action history;
- the remaining program $\rho \in \mathsf{sub}(\delta)$.

The static type of the abstract state w.r.t. the *objective* context $\mathcal{C}_\mathsf{o}$ is uniquely determined by the dynamic type $\mathfrak{t}$ of the world state and the accumulated effects L. Next, we show that the sensing result of an action sequence $\sigma \in \boldsymbol{A}^*$ in an interpretation $\mathcal{I}$ can be represented as a subset of $\mathcal{C}_\mathsf{o} \times 2^{\mathsf{Lit}(\Sigma_\mathsf{s})}$.

Definition 7.15. Let $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory, where $\mathsf{eff}(\boldsymbol{\alpha})$ is a set of unconditional local effects for each ground action $\boldsymbol{\alpha} \in \boldsymbol{A}$. First, we extend $\mathsf{eff}(\cdot)$ to sequences in $\boldsymbol{A}^*$.

For some $\sigma \in \boldsymbol{A}^*$ the *set of effects* $\mathsf{eff}(\sigma)$ is defined inductively as follows:

$$\begin{aligned}&\mathsf{eff}(\langle\rangle) := \emptyset\\ &\mathsf{eff}(\sigma' \cdot \boldsymbol{\alpha}) := \big(\mathsf{eff}(\sigma') \setminus \neg\mathsf{eff}(\boldsymbol{\alpha})\big) \cup \mathsf{eff}(\boldsymbol{\alpha}) \text{ with } \sigma' \in \boldsymbol{A}^* \text{ and } \boldsymbol{\alpha} \in \boldsymbol{A}.\end{aligned}$$

Let $\mathcal{C}$ be an $\mathcal{ALCO}$-context such that $\{\mathsf{sense}(\boldsymbol{\alpha}), \neg\mathsf{sense}(\boldsymbol{\alpha})\} \subseteq \mathcal{C}$ holds for all $\boldsymbol{\alpha} \in \boldsymbol{A}$, $\mathsf{Lit}(\Sigma_\mathsf{s})$ the set of all relevant effects, $\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_n \in \boldsymbol{A}^*$ for some $n \geq 0$ an action sequence and

$$\mathsf{s}_\sigma = (\psi_1, \ldots, \psi_n)$$

a sensing result of σ w.r.t. Σ_s. The *corresponding subset of* $\mathcal{C} \times 2^{\mathsf{Lit}(\Sigma_\mathsf{s})}$ is defined as follows:

$$\mathfrak{t}_\mathsf{s}(\mathsf{s}_\sigma) := \{(\psi_1, \emptyset)\} \cup \{(\psi_i, \mathsf{eff}(\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_{i-1})) \mid i \in \{2, \ldots, n\}\}.$$

▲

It is easy to see that the sensing result of an action sequence uniquely determines the resulting knowledge state.

Lemma 7.16. *Let* $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$*,* $\mathcal{C}$ *and* $\mathsf{Lit}(\Sigma_\mathsf{s})$ *be as above,*

$$\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_\mathsf{s})$$

the induced epistemic FO-DS, $\sigma \in \boldsymbol{A}^*$ *an action sequence,* $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ *and* $(\mathcal{I}_\sigma, \mathcal{M})$ *the epistemic interpretation with* $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_\mathsf{K}} (\mathcal{I}_\sigma, \mathcal{M})$*. It holds that*

$$\mathcal{M} = \{\mathcal{J}^{\mathsf{eff}(\sigma)} \mid \mathcal{J} \in \mathcal{M}(\mathcal{K}), \mathfrak{t}_\mathsf{s}(\mathsf{s}_\sigma(\mathcal{I})) = \mathfrak{t}_\mathsf{s}(\mathsf{s}_\sigma(\mathcal{J}))\}.$$

Given the current sensing result $\mathfrak{t}_\mathsf{s} \subseteq \mathcal{C}_\mathsf{o} \times 2^{\mathsf{Lit}(\Sigma_\mathsf{s})}$, the accumulated effects L and the initial KB $\mathcal{K}$ we construct a representation of the current knowledge state in form of a reduction KB. With this reduction KB, which is a Boolean $\mathcal{ALCO}$-KB, the static type of the knowledge state w.r.t. $\mathcal{C}_\mathsf{K}$ can be determined. The reduction KB has the same structure as the KB for checking realizability of dynamic types in presence of local effects (Lemma 4.12) and the one used for solving the epistemic projection problem (Lemma 6.54).

Definition 7.17. Let $\mathfrak{t}_\mathsf{s} \subseteq \mathcal{C}_\mathsf{o} \times 2^{\mathsf{Lit}(\Sigma_\mathsf{s})}$ be a representation of the sensing results, $\mathcal{F}$ the set of all concept names and role names occurring in $\mathcal{C}_\mathsf{o}$ and $\mathsf{Lit}(\Sigma_\mathsf{s})$, $\mathsf{Obj}(\Sigma_\mathsf{s})$ the set of all object names mentioned in $\mathcal{C}_\mathsf{o}$ and $\mathsf{Lit}(\Sigma_\mathsf{s})$, and $\mathsf{sub}(\mathcal{C}_\mathsf{o})$ the set of all sub-concepts occurring in $\mathcal{C}_\mathsf{o}$.

We define a Boolean $\mathcal{ALCO}$-KB

$$\mathcal{K}_{\mathrm{red}}^{\mathsf{t_s}}$$

that uses the following fresh concept, role and object names: for each non-contradictory set $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$, each $F \in \mathcal{F}$ and each concept $C \in \mathsf{sub}(\mathcal{C}_\mathsf{o})$ there are concept/role names $F^{(\mathsf{L})}$ and concept names $T_C^{(\mathsf{L})}$; a concept name N_c representing the set of all named elements including c; and there is a distinguished object name $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$.

Let $\psi \in \mathcal{C}_\mathsf{o}$ and $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ a non-contradictory set of effects. The copy $\psi^{(\mathsf{L})}$ is obtained from ψ be replacing each $C \sqsubseteq D$ in ψ by $T_C^{(\mathsf{L})} \sqsubseteq T_D^{(\mathsf{L})}$, each $o \mathrel{\mathsf{E}} C$ by $o \mathrel{\mathsf{E}} T_C^{(\mathsf{L})}$ and each $(o,o') \mathrel{\mathsf{E}} P$ by $(o,o') \mathrel{\mathsf{E}} P^{(\mathsf{L})}$.

The TBox

$$\mathcal{T}_{\mathsf{sub}(\mathcal{C}_\mathsf{o})}$$

consists of the following definitions

- for each new name $T_C^{(\mathsf{L})}$ a definition as defined in Figure 4.1 but with N_c instead of N, and
- the definition $N_c \equiv \bigsqcup_{o \in \mathsf{Obj}(\Sigma_\mathsf{s}) \cup \{c\}} \{o\}$.

The ABox

$$\mathcal{A}_{\mathsf{eff}}$$

consists of all assertions of the ABox given in (4.5). Finally, we define

$$\mathcal{K}_{\mathrm{red}}^{\mathsf{t_s}} := \mathcal{T}_{\mathsf{sub}(\mathcal{C}_\mathsf{o})} \wedge \mathcal{A}_{\mathsf{eff}} \wedge \bigwedge_{\varphi \text{ in } \mathcal{K}} \varphi^{(\emptyset)} \wedge \bigwedge_{(\psi,\mathsf{L}) \in \mathsf{t_s}} \psi^{(\mathsf{L})}.$$

We also allow for an additional acyclic TBox similar to the one in (6.5). Let $\mathcal{T}_\mathsf{a}$ be an acyclic $\mathcal{ALCO}$-TBox such that

- all defined concept names in $\mathcal{T}_\mathsf{a}$ are not contained in $\mathcal{F}$, and
- all other non-defined concept names and all role names in $\mathcal{T}_\mathsf{a}$ are contained in $\mathcal{F}$.

Let X be an $\mathcal{ALCO}$-concept or a Boolean $\mathcal{ALCO}$-KB over concept names and role names from $\mathcal{F}$ and defined concept names from $\mathcal{T}_\mathsf{a}$. For each sub-concept $C \in \mathsf{sub}(X) \cup \mathsf{sub}(\mathcal{T}_\mathsf{a})$ and each non-contradictory set $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ a new concept name $T_C^{(\mathsf{L})}$ is introduced. With

$$\mathcal{T}_{\mathsf{sub}(X,\mathcal{T}_\mathsf{a})} \tag{7.1}$$

we denote the conjunction of the following concept definitions:

- for all $C \in \mathsf{sub}(X) \cup \mathsf{sub}(\mathcal{T}_\mathsf{a})$ where C is not a defined name in $\mathcal{T}_\mathsf{a}$, and all non-contradictory sets $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ a definition of $T_C^{(\mathsf{L})}$ according to Figure 4.1 as above;
- $T_A^{(\mathsf{L})} \equiv T_D^{(\mathsf{L})}$ for each definition $A \equiv D \in \mathcal{T}_\mathsf{a}$ and each non-contradictory set $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$.

Let Y be a Boolean $\mathcal{ALCO}$-KB or an $\mathcal{ALCO}$-concept. With $Y_{\mathcal{T}_\mathsf{a}}$ we denote the Boolean KB or concept obtained from Y be exhaustively replacing each occurrence of a defined name in $\mathcal{T}_\mathsf{a}$ in Y with the corresponding right-hand side of the definition. ▲

The following lemma describes the essential property of the reduction KB analogous to Lemma 6.52.

Lemma 7.18. *Let* $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$*,* $\mathcal{C}_{\mathsf{o}}$*,* $\mathsf{Lit}(\Sigma_{\mathsf{s}})$*,* $\mathcal{F}$*,* $\mathsf{Obj}(\Sigma_{\mathsf{s}})$ *and* $\mathfrak{t}_{\mathsf{s}} \subseteq \mathcal{C}_{\mathsf{o}} \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$ *be as above, and* $\mathcal{T}_{\mathsf{a}}$ *an acyclic* $\mathcal{ALCO}$*-TBox such that all defined concept names in* $\mathcal{T}_{\mathsf{a}}$ *are not contained in* $\mathcal{F}$*, and all other non-defined concept names and all role names in* $\mathcal{T}_{\mathsf{a}}$ *are contained in* $\mathcal{F}$*, and let* X *be an* $\mathcal{ALCO}$*-concept or a Boolean* $\mathcal{ALCO}$*-KB over concept names and role names from* $\mathcal{F}$ *and defined concept names from* $\mathcal{T}_{\mathsf{a}}$*, and object names from* $\mathsf{Obj}(\Sigma_{\mathsf{s}})$*.*

1. *For every* $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ *there exists a* $\mathcal{J}$ *such that*

$$\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C}_{\mathsf{o}})} \wedge \mathcal{A}_{\mathsf{eff}} \wedge \bigwedge_{\varphi \text{ in } \mathcal{K}} \varphi^{(\emptyset)} \wedge \mathcal{T}_{\mathsf{sub}(X,\mathcal{T}_{\mathsf{a}})}$$

   and it holds that

$$\mathcal{I}^{\mathsf{L}} \models \psi_{\mathcal{T}_{\mathsf{a}}} \text{ iff } \mathcal{J} \models \psi^{(\mathsf{L})}\text{, for all non-contradictory } \mathsf{L} \subseteq \mathsf{Lit}(\Sigma_{\mathsf{s}})$$

   and for any Boolean $\mathcal{ALCO}$*-KB* ψ *with* $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\mathcal{C}_{\mathsf{o}}) \cup \mathsf{sub}(X, \mathcal{T}_{\mathsf{a}})$*.*

2. *For every interpretation* $\mathcal{J}$ *with*

$$\mathcal{J} \models \mathcal{T}_{\mathsf{sub}(\mathcal{C}_{\mathsf{o}})} \wedge \mathcal{A}_{\mathsf{eff}} \wedge \bigwedge_{\varphi \text{ in } \mathcal{K}} \varphi^{(\emptyset)} \wedge \mathcal{T}_{\mathsf{sub}(X,\mathcal{T}_{\mathsf{a}})},$$

   there exists $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ *such that it holds that*

$$\mathcal{I}^{\mathsf{L}} \models \psi_{\mathcal{T}_{\mathsf{a}}} \text{ iff } \mathcal{J} \models \psi^{(\mathsf{L})}\text{, for all non-contradictory } \mathsf{L} \subseteq \mathsf{Lit}(\Sigma_{\mathsf{s}})$$

   and for any Boolean $\mathcal{ALCO}$*-KB* ψ *with* $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\mathcal{C}_{\mathsf{o}}) \cup \mathsf{sub}(X, \mathcal{T}_{\mathsf{a}})$*.*

Proof. Essentially, the proof works in the same way as the one given in [Baa+05b] (page 17, Theorem 14). □

The next lemma is a direct consequence.

Lemma 7.19. *Let* $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$*,* $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$*,* $\mathcal{C}_{\mathsf{o}}$*,* $\mathsf{Lit}(\Sigma_{\mathsf{s}})$*,* $\mathcal{F}$*,* X *and* $\mathcal{T}_{\mathsf{a}}$ *be as above. Furthermore let* $\sigma \in \boldsymbol{A}^*$ *be an action sequence,* $\mathcal{I} \in \mathcal{M}(\mathcal{K})$*,* $\mathfrak{t}_{\mathsf{s}} = \mathfrak{t}_{\mathsf{s}}(\mathsf{s}_{\sigma}(\mathcal{I}))$ *and* $(\mathcal{I}_{\sigma}, \mathcal{M})$ *the epistemic interpretation with* $(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathsf{K}}} (\mathcal{I}_{\sigma}, \mathcal{M})$*.*

1. *For every* $\mathcal{Y} \in \mathcal{M}$ *there exists a* $\mathcal{J}$ *such that* $\mathcal{J} \models \mathcal{K}^{\mathfrak{t}_{\mathsf{s}}}_{red} \wedge \mathcal{T}_{\mathsf{sub}(X,\mathcal{T}_{\mathsf{a}})}$ *and it holds that*

$$\mathcal{Y} \models \psi_{\mathcal{T}_{\mathsf{a}}} \text{ iff } \mathcal{J} \models \psi^{(\mathsf{L})} \text{ with } \mathsf{L} = \mathsf{eff}(\sigma)$$

   for any Boolean $\mathcal{ALCO}$*-KB* ψ *with* $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\mathcal{C}_{\mathsf{o}}) \cup \mathsf{sub}(X, \mathcal{T}_{\mathsf{a}})$*.*

2. *For every interpretation* $\mathcal{J}$ *with* $\mathcal{J} \models \mathcal{K}^{\mathfrak{t}_{\mathsf{s}}}_{red} \wedge \mathcal{T}_{\mathsf{sub}(X,\mathcal{T}_{\mathsf{a}})}$*, there exists* $\mathcal{Y} \in \mathcal{M}$ *such that it holds that*

$$\mathcal{Y} \models \psi_{\mathcal{T}_{\mathsf{a}}} \text{ iff } \mathcal{J} \models \psi^{(\mathsf{L})} \text{ with } \mathsf{L} = \mathsf{eff}(\sigma)$$

   for any Boolean $\mathcal{ALCO}$*-KB* ψ *with* $\mathsf{sub}(\psi) \subseteq \mathsf{sub}(\mathcal{C}_{\mathsf{o}}) \cup \mathsf{sub}(X, \mathcal{T}_{\mathsf{a}})$*.*

We can now define an instance function based on entailments of $\mathcal{K}^{\mathsf{t_s}}_{\text{red}}$ analogous to (6.6), (6.7) and (7.4).

Definition 7.20. Let $\mathsf{t_s} \subseteq \mathcal{C}_o \times 2^{\mathsf{Lit}(\Sigma_s)}$, $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_s)$ a non-contradictory set, $\mathcal{F}$ the set of all relevant concept and role names and $\mathsf{Obj}(\Sigma_s)$ the set of all relevant object names. Furthermore let $\mathcal{T}_a$ be an acyclic $\mathcal{ALCO}$-TBox such that all concept names and role names in $\mathcal{T}_a$ are either defined concept names in $\mathcal{T}_a$ or are contained in $\mathcal{F}$ and let $\mathbf{K}D$ be an $\mathcal{ALCOK}$-concept, where D is objective and mentions only concept names and role names from $\mathcal{F}$ and $\mathcal{T}_a$ and only object names from $\mathsf{Obj}(\Sigma_s)$.

We define a set $\kappa_{\mathsf{t_s},\mathsf{L}}(\mathbf{K}D) \subseteq \mathsf{Nom}(\Sigma_s)$, where $\mathsf{Nom}(\Sigma_s)$ is defined in (6.3), as follows

$$\begin{aligned}\kappa_{\mathsf{t_s},\mathsf{L}}(\mathbf{K}D) :=& \Big\{\{o\} \;\Big|\; o \in \mathsf{Obj}(\Sigma_s), \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \wedge \mathcal{T}_{\mathsf{sub}(D,\mathcal{T}_a)} \models \big(o \in T^{(n)}_D\big)\Big\} \cup \\ &\Big\{\neg N \;\Big|\; \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \wedge \mathcal{T}_{\mathsf{sub}(D,\mathcal{T}_a)} \models \big(c \in T^{(n)}_D\big)\Big\}.\end{aligned} \tag{7.2}$$

Let $P \in \mathcal{F}$ be a role name. We define $\kappa_{\mathsf{s}_\sigma}(X, P) \subseteq \mathsf{Nom}(\Sigma_s)$ for some $X \in \mathsf{Nom}(\Sigma_s)$ as follows as follows:

$$\begin{aligned}\kappa_{\mathsf{t_s},\mathsf{L}}(\{o\}, P) :=& \ \big\{\{o'\} \;\big|\; o' \in \mathsf{Obj}(\Sigma_s), \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \models (o, o') \in P^{(n)}\big\} \cup \\ &\ \big\{\neg N \;\big|\; \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \models (o, c) \in P^{(n)}\big\} \text{ for all } o \in \mathsf{Obj}(\Sigma_s); \\ \kappa_{\mathsf{t_s},\mathsf{L}}(\neg N, P) :=& \ \big\{\{o'\} \;\big|\; o' \in \mathsf{Obj}(\Sigma_s), \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \models (c, o') \in P^{(n)}\big\} \cup \\ &\ \big\{\neg N \;\big|\; \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \models (c, c) \in P^{(n)}\big\}.\end{aligned} \tag{7.3}$$

Let $\mathcal{T}_a$ be as above and $\mathbf{K}\varrho$ a Boolean $\mathcal{ALCOK}$-KB, where ϱ is objective and mentions only concept and role names from $\mathcal{F}$ and $\mathcal{T}_a$ and object names from $\mathsf{Obj}(\Sigma_s)$. We define

$$\kappa_{\mathsf{t_s},\mathsf{L}}(\mathbf{K}\varrho) := \begin{cases}\textsc{True} & \text{if } \mathcal{K}^{\mathsf{t_s}}_{\text{red}} \wedge \mathcal{T}_{\mathsf{sub}(\varrho,\mathcal{T}_a)} \models \varrho^{(n)}; \\ \textsc{False} & \text{otherwise.}\end{cases} \tag{7.4}$$

▲

Based on the instance function $\kappa_{\mathsf{t_s},\mathsf{L}}$ as defined above a corresponding rewrite relation, denoted by

$$\vdash_{\mathsf{t_s},\mathsf{L}},$$

for subjective Boolean $\mathcal{ALCOK}$-KBs is defined as in Definition 6.55. As a consequence of Lemma 7.19 we have that Lemma 6.56 holds for $\vdash_{\mathsf{t_s},\mathsf{L}}$ as well.

Thus, given $\mathsf{t_s}$ and L as an abstraction of the current knowledge state the static type w.r.t. the subjective context $\mathcal{C}_\mathbf{K}$ can be determined by rewriting the KBs in $\mathcal{C}_\mathbf{K}$ into objective ones and then checking whether they are entailed by $\mathcal{K}^{\mathsf{t_s}}_{\text{red}}$.

We are now ready to define the abstract transition system.

Definition 7.21. Let $\mathcal{P} = (\mathfrak{D}_\mathbf{K}(\Sigma_s), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program over unconditional ground actions with $\Sigma_s = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$, $\mathfrak{T}_\mathcal{P}$ the induced transition system, and let $\mathcal{C} = \mathcal{C}_\mathbf{K} \uplus \mathcal{C}_o$ be a proper context for $\mathcal{P}$ that consists of subjective or objective Boolean $\mathcal{ALCOK}$-KBs, and AP a finite set of atomic propositions with a bijection $\iota_\mathcal{C} : \mathcal{C} \to \mathsf{AP}$.

The *abstraction of $\mathfrak{I}_{\mathcal{P}}$ w.r.t. $\mathcal{C}$* is a propositional transition system

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$$

over AP, where

- $Q_{\mathfrak{T}_{\mathcal{P}}} := \{(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho) \mid \mathfrak{t} \in \mathsf{D\text{-}Types}(\mathcal{C}_\mathsf{o}, \mathsf{Lit}(\Sigma_\mathsf{s})), \mathfrak{t}_\mathsf{s} \subseteq \mathfrak{t}, \mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s}), \rho \in \mathsf{sub}(\delta)\}$;
- $I_{\mathfrak{T}_{\mathcal{P}}} := \left\{(\mathfrak{t}, \emptyset, \emptyset, \delta) \in Q_{\mathfrak{T}_{\mathcal{P}}} \;\middle|\; (\varphi, \emptyset) \in \mathfrak{t}, \forall \varphi.\varphi \text{ occurring in the initial KB of } \Sigma_\mathsf{s} \uplus \{\epsilon, \mathfrak{f}\}\right\}$;
- for each state $(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}}$ the label set $\lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho)$ is defined as follows
  - for each $\psi \in \mathcal{C}_\mathsf{o}$ it holds that $\iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho)$ iff $(\psi, \mathsf{L}) \in \mathfrak{t}$;
  - for each $\psi \in \mathcal{C}_\mathsf{K}$ it holds that $\iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho)$ iff for the objective rewriting $(\mathcal{T}_\mathsf{a}, \varphi)$ with $(\emptyset, \psi) \vdash^*_{\mathfrak{t}_\mathsf{s}, \mathsf{L}} (\mathcal{T}_\mathsf{a}, \varphi)$ it holds that $\mathcal{K}^{\mathfrak{t}_\mathsf{s}}_{\mathsf{red}} \wedge \mathcal{T}_{\mathsf{sub}(\varphi, \mathcal{T}_\mathsf{a})} \models \varphi^{(\mathsf{L})}$;
- $\hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} := \left\{\big((\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho), (\mathfrak{t}', \mathfrak{t}'_\mathsf{s}, \mathsf{L}', \rho')\big) \in Q_{\mathfrak{T}_{\mathcal{P}}} \times Q_{\mathfrak{T}_{\mathcal{P}}} \;\middle|\; \mathfrak{t} = \mathfrak{t}', \text{ (i) or (ii) }\right\}$ with
  (i) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ such that
    - $\mathfrak{a}$ is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C}_\mathsf{K} \mid \iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho)\}$, and
    - $\mathsf{L}' = (\mathsf{L} \setminus \neg\mathsf{eff}(\boldsymbol{\alpha})) \cup \mathsf{eff}(\boldsymbol{\alpha})$ and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$ and
    - $\mathfrak{t}'_\mathsf{s} = \mathfrak{t}_\mathsf{s} \cup (\{(\mathsf{sense}(\boldsymbol{\alpha}), \mathsf{L}), (\neg\mathsf{sense}(\boldsymbol{\alpha}), \mathsf{L})\} \cap \mathfrak{t})$.

  (ii) there is *no* guarded action contained in $\mathsf{head}(\rho)$ that is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C}_\mathsf{K} \mid \iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \rho)\}$ and we have
  $$\mathsf{L}' = (\mathsf{L} \setminus \neg\mathsf{eff}(\mathfrak{f})) \cup \mathsf{eff}(\mathfrak{f})$$
  and $\rho = \rho'$, and $\mathfrak{t}'_\mathsf{s} = \mathfrak{t}_\mathsf{s}$.

▲

To determine the subjective label of an abstract state the representation of the knowledge state in form of the reduction KB is used. Since the KBs in $\mathcal{C}_\mathsf{K}$ are subjective, the resulting rewriting is interpreted in the same way in each possible world (Lemma 6.49). Abstract states a progressed by choosing an executable guarded action, adding the respective effects, choosing the remaining program and by updating the sensing result. In the second component of each state we keep track of all observations made throughout the execution. If $\boldsymbol{\alpha} \in \boldsymbol{A}$ is executed and L are the currently accumulated effects, then either $(\mathsf{sense}(\boldsymbol{\alpha}), \mathsf{L})$ or $(\neg\mathsf{sense}(\boldsymbol{\alpha}), \mathsf{L})$ is contained in the dynamic type representing the world state. After doing $\boldsymbol{\alpha}$ the pair contained in $\mathfrak{t}$ is added to the sensing result.

We show that $\mathfrak{T}_{\mathcal{P}}$ is bisimilar propositional abstraction of $\mathcal{P}$. Let $\mathfrak{I}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ be the transition system induced by $\mathcal{P}$ and $\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$ the abstraction of $\mathfrak{I}_{\mathcal{P}}$ w.r.t. $\mathcal{C}$. We define a relation

$$\simeq_{\mathcal{C}} \subseteq Q_{\mathcal{P}} \times Q_{\mathfrak{T}_{\mathcal{P}}}$$

such that

$$\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \simeq_{\mathcal{C}} (\mathfrak{t}, \mathfrak{t}_\mathsf{s}, \mathsf{L}, \theta) \text{ iff the following conditions are satisfied}$$

- there exists $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \in I_{\mathcal{P}}$ such that $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \hookrightarrow_{\mathcal{P}}{}^* \langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle$ and $\mathsf{t} = \text{d-type}^{\text{loc}}_{\mathcal{C}_{\text{o}}}(\mathcal{I}_0)$, $\mathcal{I} = \mathcal{I}_0{}^{\mathsf{L}}$ and $\mathsf{t}_{\mathsf{s}} = \mathsf{t}_{\mathsf{s}}(\mathsf{s}_\sigma(\mathcal{I}_0))$, and
- $\rho = \theta$.

Lemma 7.22. *$\simeq_{\mathcal{C}}$ is a $\mathcal{C}$-bisimulation.*

Proof. Let

$$\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_{\mathsf{s}}).$$

Let $\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \in Q_{\mathcal{P}}$ and $(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \theta) \in Q_{\mathfrak{T}_{\mathcal{P}}}$ such that $\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \simeq_{\mathcal{C}} (\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \theta)$. Let $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ such that

- $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \hookrightarrow_{\mathcal{P}}{}^* \langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle$ and
- $\mathsf{t} = \text{d-type}^{\text{loc}}_{\mathcal{C}_{\text{o}}}(\mathcal{I}_0)$,
- $\mathcal{I} = \mathcal{I}_0{}^{\mathsf{L}}$ and $\mathsf{t}_{\mathsf{s}} = \mathsf{t}_{\mathsf{s}}(\mathsf{s}_\sigma(\mathcal{I}_0))$.

It follows that

$$(\mathcal{I}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}} (\mathcal{I}, \mathcal{W}) \text{ and } \mathsf{L} = \mathsf{eff}(\sigma).$$

First, we show that

$$\lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \theta) = \{\iota_{\mathcal{C}}(\psi) \mid \psi \in \text{s-type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W})\}. \tag{7.5}$$

For all $\psi \in \mathcal{C}_{\text{o}}$ it is easy to see that

$$\begin{aligned}
&\psi \in \text{s-type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W}) \text{ iff } \psi \in \text{s-type}_{\mathcal{C}_{\text{o}}}(\mathcal{I}) \text{ iff}\\
&\psi \in \text{s-type}_{\mathcal{C}_{\text{o}}}(\mathcal{I}_0{}^{\mathsf{L}}) \text{ iff}\\
&(\psi, \mathsf{L}) \in \mathsf{t} = \text{d-type}^{\text{loc}}_{\mathcal{C}_{\text{o}}}(\mathcal{I}_0) \text{ iff}\\
&\iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \theta).
\end{aligned}$$

Let $\psi \in \mathcal{C}_{\mathbf{K}}$. It holds that $\psi \in \text{s-type}_{\mathcal{C}}(\mathcal{I}, \mathcal{W})$

iff $(\mathcal{I}, \mathcal{W}) \models \psi$

iff $\mathcal{I} \models [\![\psi, \kappa_{\mathcal{W}}]\!]$

iff $\mathcal{J} \models [\![\psi, \kappa_{\mathcal{W}}]\!]$ for all $\mathcal{J} \in \mathcal{W}$ (Lemma 6.49)

iff $\mathcal{J} \models \varphi_{\mathcal{T}_{\mathsf{a}}}$ for all $\mathcal{J} \in \mathcal{W}$ for the objective rewriting $(\emptyset, \psi) \vdash^*_{\mathsf{t}_{\mathsf{s}},\mathsf{L}} (\mathcal{T}_{\mathsf{a}}, \varphi)$ due to Lemma 6.56 and 7.19 and Definition 7.20

iff $\mathcal{K}^{\mathsf{t}_{\mathsf{s}}}_{\text{red}} \wedge \mathcal{T}_{\mathsf{sub}(\varphi, \mathcal{T}_{\mathsf{a}})} \models \varphi^{(\mathsf{L})}$ for the objective rewriting $(\emptyset, \psi) \vdash^*_{\mathsf{t}_{\mathsf{s}},\mathsf{L}} (\mathcal{T}_{\mathsf{a}}, \varphi)$

iff $\iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \theta)$.

It remains to be shown that for each successor of $\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle$ there is a successor of $(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \theta)$ such that both successor or in $\simeq_{\mathcal{C}}$-relation, and vice versa. Assume $\sigma \in \boldsymbol{A}$. Consider a transition

$$\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \hookrightarrow_{\mathcal{P}} \langle(\mathcal{J}, \mathcal{M}), \sigma \cdot \boldsymbol{\alpha}, \zeta\rangle$$

with $\boldsymbol{\alpha} \in \boldsymbol{A}$. It follows that there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots ; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ that is executable in $(\mathcal{I}, \mathcal{W})$ and $\zeta \in \mathsf{tail}(\mathfrak{a}, \rho)$. It follows that

$$\mathcal{J} = \mathcal{I}^{\mathsf{eff}(\boldsymbol{\alpha})} = {\mathcal{I}_0}^{\mathsf{E}} \text{ with } \mathsf{E} = (\mathsf{L} \setminus \neg\mathsf{eff}(\boldsymbol{\alpha})) \cup \mathsf{eff}(\boldsymbol{\alpha}). \tag{7.6}$$

Since $\mathfrak{a}$ is executable in $(\mathcal{I}, \mathcal{W})$, (7.5) implies that $\mathfrak{a}$ is executable in

$$\{\psi \in \mathcal{C}_{\mathsf{K}} \mid \iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \rho)\}.$$

It follows that

$$(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \rho) \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} (\mathsf{t}, \mathsf{t}'_{\mathsf{s}}, \mathsf{E}, \zeta) \text{ with } \mathsf{t}'_{\mathsf{s}} = \mathsf{t}_{\mathsf{s}} \cup (\{(\mathsf{sense}(\boldsymbol{\alpha}), \mathsf{L}), (\neg\mathsf{sense}(\boldsymbol{\alpha}), \mathsf{L})\} \cap \mathsf{t}).$$

It is easy to show that

$$\langle(\mathcal{J}, \mathcal{M}), \sigma \cdot \boldsymbol{\alpha}, \zeta\rangle \simeq_{\mathcal{C}} (\mathsf{t}, \mathsf{t}'_{\mathsf{s}}, \mathsf{E}, \zeta).$$

The other direction is analogous. We omit the remaining cases. The proof works following the same lines as in the proof of Lemma 4.16. □

Thus, the abstraction defined in Definition 7.21 is a bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. the chosen proper context $\mathcal{C}$. It is finite and effectively computable. The property can be verified via propositional model checking.

Theorem 7.23. *The verification problem for $\mathcal{ALCOK}$-CTL* properties and knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions is* 2ExpTime*-complete.*

Proof. The set of all dynamic types $\mathsf{D\text{-}Types}(\mathcal{C}_{\mathsf{o}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$ can be computed in 2ExpTime and there are at most double exponentially many types. Thus, there are at most double exponentially many abstract states. For each abstract state of the form $(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \rho)$ the corresponding reduction KB $\mathcal{K}^{\mathsf{t}_{\mathsf{s}}}_{\text{red}}$ is of exponential size. The label set $\lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{t}, \mathsf{t}_{\mathsf{s}}, \mathsf{L}, \rho)$ is computable with a polynomially of entailment checks of exponential size. Therefore, for each state the label set is computable in 2ExpTime. Checking whether or not there is a transition between two abstract states given the label sets can be done in polynomial time. The computation of the bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. the chosen proper context $\mathcal{C}$ can be done in double exponential time. To decide the verification problem a propositional CTL* model checker is called with a transition system of double exponential size. Since the model checking algorithm requires polynomial time in the size of the transition system, the model checker requires double exponential time w.r.t. the size of the input. □

7.2.2 Lowering the Complexity

In this section we consider additional restrictions on the verification problem of knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions that admit a decision procedure in ExpTime. One such fragment is obtained by disallowing sensing results and by restricting the specification to $\mathcal{ALCOK}$-CTL* formulas over only subjective Boolean $\mathcal{ALCOK}$-KBs. In the other fragment actions may have sensing results but have no physical effects.

The Case without Sensing and only Subjective Temporal Properties

First, we consider knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$, where the underlying action theory $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ in addition has the following property

$$\mathsf{sense}(\boldsymbol{\alpha}) = \textsc{True} \text{ for all } \boldsymbol{\alpha} \in \boldsymbol{A}.$$

Furthermore, we consider only $\mathcal{ALCOK}$-CTL* state formulas over *subjective* Boolean $\mathcal{ALCOK}$-KBs.

The abstraction is constructed based on a proper context $\mathcal{C} = \mathcal{C}_{\mathsf{o}} \uplus \mathcal{C}_{\mathsf{K}}$, where the objective part $\mathcal{C}_{\mathsf{o}}$ only consists of the axioms contained in the initial KB $\mathcal{K}$. For the decision which action to execute next only $\mathcal{C}_{\mathsf{K}}$ is relevant. Since the temporal property also only refers to the knowledge state, it is sufficient to construct a $\mathcal{C}_{\mathsf{K}}$-bimilar abstraction. Intuitively, the actual world state is irrelevant for the verification. In this fragment there is no interaction between the world state and the knowledge state, because there are no conditional effects, i.e. the outcome of an action is immediately observable, and no observations can be made through sensing.

For the abstraction it is sufficient to keep track of the accumulated effects and the remaining program expression. Thus, we obtain a transition system with at most exponentially many states.

Definition 7.24. Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions with $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ and $\mathsf{sense}(\boldsymbol{\alpha}) = \textsc{True}$ for all $\boldsymbol{\alpha} \in \boldsymbol{A}$, and let $\mathcal{C}_{\mathsf{K}}$ be the subjective part of a proper context for $\mathcal{P}$ and AP a set of atomic propositions with a bijection $\iota_{\mathcal{C}_{\mathsf{K}}} : \mathcal{C}_{\mathsf{K}} \to \mathsf{AP}$.

Let $\mathfrak{I}_{\mathcal{P}}$ be the transition system induced by $\mathcal{P}$. The *abstraction of* $\mathfrak{I}_{\mathcal{P}}$ *w.r.t.* $\mathcal{C}_{\mathsf{K}}$ is a propositional transition system

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$$

over AP, where

- $Q_{\mathfrak{T}_{\mathcal{P}}} := \mathsf{Lit}(\Sigma_{\mathsf{s}}) \times \mathsf{sub}(\delta)$;
- $I_{\mathfrak{T}_{\mathcal{P}}} := \{(\emptyset, \delta)\}$;
- for each $(\mathsf{L}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}}$ we have

  $$\lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{L}, \rho) := \{\iota_{\mathcal{C}_{\mathsf{K}}}(\psi) \mid \psi \in \mathcal{C}_{\mathsf{K}}, \mathsf{proj}({\Sigma_{\mathsf{s}}}^{\mathsf{L}}, \boldsymbol{\alpha}^{\mathsf{L}}, \psi) = \textsc{True}\}, \text{ where}$$

  $\boldsymbol{\alpha}^{\mathsf{L}}$ is a ground action term with all object names mentioned in L as arguments, and ${\Sigma_{\mathsf{s}}}^{\mathsf{L}} = (\mathcal{K}, \{\boldsymbol{\alpha}^{\mathsf{L}}\}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ is an epistemic $\mathcal{ALCO}$-action theory with $\mathsf{pre}(\boldsymbol{\alpha}^{\mathsf{L}}) = \emptyset$, $\mathsf{eff}(\boldsymbol{\alpha}^{\mathsf{L}}) = \mathsf{L}$, $\mathsf{sense}(\boldsymbol{\alpha}^{\mathsf{L}}) = \textsc{True}$ and $\mathcal{K}$ is the initial KB that includes the assertions

  $$(\mathsf{prog} \sqsubseteq \neg \mathit{Final}) \text{ and } (\mathsf{prog} \sqsubseteq \neg \mathit{Fail}).$$

- $\hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} := \big\{\big((\mathsf{L}, \rho), (\mathsf{L}', \rho')\big) \in Q_{\mathfrak{T}_{\mathcal{P}}} \times Q_{\mathfrak{T}_{\mathcal{P}}} \text{ (i) or (ii) } \big\}$ with

  (i) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots ; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ such that

- $\mathfrak{a}$ is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C}_{\mathsf{K}} \mid \iota_{\mathcal{C}_{\mathsf{K}}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{L}, \rho)\}$, and
- $\mathsf{L}' = (\mathsf{L} \setminus \neg\mathsf{eff}(\boldsymbol{\alpha})) \cup \mathsf{eff}(\boldsymbol{\alpha})$ and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$;

(ii) there is *no* guarded action contained in $\mathsf{head}(\rho)$ that is executable in the static type $\mathfrak{s} = \{\psi \in \mathcal{C}_{\mathsf{K}} \mid \iota_{\mathcal{C}_{\mathsf{K}}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{L}, \rho)\}$ and we have

$$\mathsf{L}' = (\mathsf{L} \setminus \neg\mathsf{eff}(\mathfrak{f})) \cup \mathsf{eff}(\mathfrak{f})$$

and $\rho = \rho'$.

▲

The transition system only has a single initial state given by $(\emptyset, \delta)$. To determine the (subjective) label of an abstract state of the form (L, δ) we check for each $\psi \in \mathcal{C}_{\mathsf{K}}$ whether or not ψ is known after an update of the initial KB $\mathcal{K}$ with L. This check is described as an epistemic projection problem with a single action that has L as its set of effects. The transition relation on abstract states is defined as usual.

We show that $\mathfrak{T}_{\mathcal{P}}$ is a $\mathcal{C}_{\mathsf{K}}$-bisimilar abstraction of $\mathfrak{I}_{\mathcal{P}}$. Let $\mathfrak{I}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ be the transition system induced by $\mathcal{P}$ and $\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$ the abstraction of $\mathfrak{I}_{\mathcal{P}}$ w.r.t. $\mathcal{C}_{\mathsf{K}}$. We define a relation

$$\simeq_{\mathcal{C}_{\mathsf{K}}} \subseteq Q_{\mathcal{P}} \times Q_{\mathfrak{T}_{\mathcal{P}}}$$

such that

$$\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \simeq_{\mathcal{C}_{\mathsf{K}}} (\mathsf{L}, \theta) \text{ iff the following conditions are satisfied}$$

- there exists $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \in I_{\mathcal{P}}$ such that $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \hookrightarrow_{\mathcal{P}}^* \langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle$ and $\mathcal{I} = {\mathcal{I}_0}^{\mathsf{L}}$;
- $\rho = \theta$.

Lemma 7.25. *$\simeq_{\mathcal{C}_{\mathsf{K}}}$ is a $\mathcal{C}_{\mathsf{K}}$-bisimulation.*

Proof. Let

$$\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_{\mathsf{s}}).$$

Let $\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \in Q_{\mathcal{P}}$ and $(\mathsf{L}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}}$ such that $\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \simeq_{\mathcal{C}_{\mathsf{K}}} (\mathsf{L}, \rho)$. Let $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$ such that

- $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \hookrightarrow_{\mathcal{P}}^* \langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle$ and
- $\mathcal{I} = {\mathcal{I}_0}^{\mathsf{L}}$.

It follows that

$$(\mathcal{I}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow_{\mathfrak{D}_{\mathsf{K}}}^{\sigma} (\mathcal{I}, \mathcal{W}) \text{ and } \mathsf{L} = \mathsf{eff}(\sigma).$$

Since the actions in σ only have unconditional effects and provide no sensing results, it follows that

$$\mathcal{W} = \{\mathcal{J}^{\mathsf{L}} \mid \mathcal{J} \in \mathcal{M}(\mathcal{K})\}. \tag{7.7}$$

We show that

$$\lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{L},\rho) = \{\iota_{\mathcal{C}_{\mathbf{K}}}(\psi) \mid \psi \in \text{s-type}_{\mathcal{C}_{\mathbf{K}}}(\mathcal{I},\mathcal{W})\}.$$

Let ${\Sigma_{\mathsf{s}}}^{\mathsf{L}} = (\mathcal{K}, \{\boldsymbol{\alpha}^{\mathsf{L}}\}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ with $\mathsf{pre}(\boldsymbol{\alpha}^{\mathsf{L}}) = \emptyset$, $\mathsf{eff}(\boldsymbol{\alpha}^{\mathsf{L}}) = \mathsf{L}$ and $\mathsf{sense}(\boldsymbol{\alpha}^{\mathsf{L}}) = \textsc{True}$.
It holds that $\iota_{\mathcal{C}_{\mathbf{K}}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathsf{L},\rho)$ for some $\psi \in \mathcal{C}_{\mathbf{K}}$

iff $\mathsf{proj}({\Sigma_{\mathsf{s}}}^{\mathsf{L}}, \boldsymbol{\alpha}^{\mathsf{L}}, \psi) = \textsc{True}$

iff for all $\mathcal{J}_0 \in \mathcal{M}(\mathcal{K})$ we have $(\mathcal{Y},\mathcal{M}) \Vdash \psi$ with $(\mathcal{J}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\boldsymbol{\alpha}^{\mathsf{L}}}_{\mathfrak{D}_{\mathbf{K}}} (\mathcal{Y},\mathcal{M})$

iff $(\mathcal{Y},\mathcal{M}) \Vdash \psi$ with $\mathcal{M} = \{\mathcal{J}^{\mathsf{L}} \mid \mathcal{J} \in \mathcal{M}(\mathcal{K})\}$ for some $\mathcal{Y} \in \mathcal{M}$ (since ψ is subjective and by definition of ${\Sigma_{\mathsf{s}}}^{\mathsf{L}}$)

iff $\psi \in \text{s-type}_{\mathcal{C}_{\mathbf{K}}}(\mathcal{I},\mathcal{W})$ (with (7.7)).

We omit the proof that both states have matching successor that are $\simeq_{\mathcal{C}_{\mathbf{K}}}$-related. A successor $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle$ of is obtained by choosing the next possible ground action $\boldsymbol{\alpha}$ from the head of ρ, by updating all possible worlds in $\mathcal{W}$ with the unconditional effects in $\mathsf{eff}(\boldsymbol{\alpha})$ and by choosing a remaining program expression. In the same way the abstract state (L,ρ) is progressed: accumulating L and the effects of the next possible action and updating ρ. Since the subjective static type of $(\mathcal{I},\mathcal{W})$ and the subjective label of (L,ρ) coincide, the same actions are possible in both states. It follows that the respective successor states are $\simeq_{\mathcal{C}_{\mathbf{K}}}$-related as well. □

It follows that $\mathfrak{I}_{\mathcal{P}}$ and $\mathfrak{T}_{\mathcal{P}}$ are $\mathcal{C}_{\mathbf{K}}$-bisimilar. Thus, $\mathfrak{T}_{\mathcal{P}}$ is a finite bisimilar propositional abstraction of $\mathcal{P}$ w.r.t. $\mathcal{C}_{\mathbf{K}}$. CTL* properties over Boolean $\mathcal{ALCOK}$-KBs from $\mathcal{C}_{\mathbf{K}}$ can be verified by checking whether they are modeled by $\mathfrak{T}_{\mathcal{P}}$.

Theorem 7.26. *The verification problem for $\mathcal{ALCOK}$-CTL* formulas over subjective Boolean $\mathcal{ALCOK}$-KBs and knowledge-based $\mathcal{ALCOK}$-ConGolog programs over unconditional ground actions without sensing is* ExpTime*-complete.*

Proof. The abstract transition system $\mathfrak{T}_{\mathcal{P}}$ defined above has at most $|\mathsf{Lit}(\Sigma_{\mathsf{s}}) \times \mathsf{sub}(\delta)|$ many states. Therefore there are at most exponentially many states that can be enumerated in ExpTime as a consequence of Theorem 3.15. To determine the labels we have to solve an instance of the epistemic projection problem for each state and each relevant subjective KB $\psi \in \mathcal{C}_{\mathbf{K}}$. This can be done in ExpTime (Theorem 6.58). Checking whether there is a transition between two labeled states can be done in polynomial time in the size of the input. Thus, $\mathfrak{T}_{\mathcal{P}}$ is of exponential size and can be computed in exponential time. To verify a given CTL* state formula Φ over subjective KBs from the relevant context $\mathcal{C}_{\mathbf{K}}$ a propositional CTL* model checker is called with $\mathfrak{T}_{\mathcal{P}}$ and Φ as input. The model checking problem can be solved in polynomial time w.r.t. the size of the transition system and in exponential time in the size of the temporal formula. Thus, the model checking task can be solved in exponential time in the size of the input.

The ExpTime lower bound comes from the consistency problem of $\mathcal{ALCO}$-KBs. □

The Case with purely Sensing Actions

Next, we consider the fragment where actions have no physical effects at all. The program only describes observations made by the agent. We consider knowledge-based $\mathcal{ALCOK}$-ConGolog programs of the form $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ with the following restrictions:

- $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ is an epistemic $\mathcal{ALCO}$-action theory for a finite set of ground actions $\boldsymbol{A}$ with $\mathsf{eff}(\boldsymbol{\alpha}) = \emptyset$ for all $\boldsymbol{\alpha} \in \boldsymbol{A}$;
- δ is a *pick-free* program expression over $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$.

Again, we consider only proper $\mathcal{ALCOK}$-contexts of $\mathcal{P}$ of the form $\mathcal{C} = \mathcal{C}_{\mathsf{K}} \uplus \mathcal{C}_{\mathsf{o}}$, where $\mathcal{C}_{\mathsf{K}}$ is the subjective part and $\mathcal{C}_{\mathsf{o}}$ the objective one. We do not consider Boolean $\mathcal{ALCOK}$-KBs that are neither objective nor subjective.

Intuitively, executing a purely sensing action is nothing more than just adding new objective axioms to the knowledge base of the agent. In the abstraction of the program the state of the world is represented as its static type w.r.t. $\mathcal{C}_{\mathsf{o}}$. Since actions do not have any effects on the world state, this part remains unchanged. The current knowledge state is just a subset of the static type of the world state.

We define an abstract propositional transition system as follows.

Definition 7.27. Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog programs over purely sensing actions with $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ where $\boldsymbol{A}$ is a finite set of ground actions, and let $\mathcal{C} = \mathcal{C}_{\mathsf{K}} \uplus \mathcal{C}_{\mathsf{o}}$ be a proper context for $\mathcal{P}$ and AP a set of atomic propositions with a bijection $\iota_{\mathcal{C}} : \mathcal{C} \to \mathsf{AP}$.

Let $\mathfrak{I}_{\mathcal{P}}$ be the transition system induced by $\mathcal{P}$. The *abstraction of* $\mathfrak{I}_{\mathcal{P}}$ *w.r.t.* $\mathcal{C}$ is a propositional transition system

$$\mathfrak{T}_{\mathcal{P}} = (Q_{\mathfrak{T}_{\mathcal{P}}}, I_{\mathfrak{T}_{\mathcal{P}}}, \hookrightarrow_{\mathfrak{T}_{\mathcal{P}}}, \lambda_{\mathfrak{T}_{\mathcal{P}}})$$

over AP, where

- $Q_{\mathfrak{T}_{\mathcal{P}}} := \{(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho) \in \mathfrak{S}_{\mathcal{C}_{\mathsf{o}}} \times \mathfrak{S}_{\mathcal{C}_{\mathsf{o}}} \times \mathsf{sub}(\delta) \mid \mathfrak{s}_{\mathsf{s}} \subseteq \mathfrak{s}\}$;
- $I_{\mathfrak{T}_{\mathcal{P}}} := \{(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}} \mid \varphi \in \mathfrak{s} \text{ for all } \varphi \in \mathcal{K} \cup \{(\mathsf{prog} \sqsubseteq \neg \mathit{Final}), (\mathsf{prog} \sqsubseteq \neg \mathit{Fail})\},$
  $\mathfrak{s}_{\mathsf{s}} = \mathcal{K} \cup \{(\mathsf{prog} \sqsubseteq \neg \mathit{Final}), (\mathsf{prog} \sqsubseteq \neg \mathit{Fail})\},$
  $\rho = \delta\}$;
- for each $(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho) \in Q_{\mathfrak{T}_{\mathcal{P}}}$ we have

  $$\lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho) := \{\iota_{\mathcal{C}}(\psi) \mid \psi \in \mathfrak{s}\} \cup \{\iota_{\mathcal{C}}(\psi) \mid \psi \in \mathcal{C}_{\mathsf{K}}, \mathsf{proj}(\Sigma_{\mathsf{s}}(\mathfrak{s}_{\mathsf{s}}), \langle\rangle, \psi) = \text{TRUE}\},$$

  where $\Sigma_{\mathsf{s}}(\mathfrak{s}_{\mathsf{s}}) = (\mathfrak{s}_{\mathsf{s}}, \emptyset, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$;
- $\hookrightarrow_{\mathfrak{T}_{\mathcal{P}}} := \Big\{\big((\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho), (\mathfrak{s}', \mathfrak{s}'_{\mathsf{s}}, \rho')\big) \in Q_{\mathfrak{T}_{\mathcal{P}}} \times Q_{\mathfrak{T}_{\mathcal{P}}} \;\Big|\; \text{(i) or (ii) or (ii)} \Big\}$ with
  (i) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ with $\boldsymbol{\alpha} \in \boldsymbol{A}$ such that
    - $\mathfrak{s} = \mathfrak{s}'$, and
    - $\mathfrak{a}$ is executable in the static type $\{\psi \in \mathcal{C}_{\mathsf{K}} \mid \iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho)\}$, and

  - $\mathfrak{s}'_{\mathsf{s}} = \mathfrak{s}_{\mathsf{s}} \cup (\{\mathsf{sense}(\boldsymbol{\alpha}), \neg\mathsf{sense}(\boldsymbol{\alpha})\} \cap \mathfrak{s})$, and
  - $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$;

(ii) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \epsilon \in \mathsf{head}(\rho)$ such that
  - $\mathfrak{a}$ is executable in the static type $\{\psi \in \mathcal{C}_{\mathsf{K}} \mid \iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho)\}$, and
  - $\mathfrak{s}' = \mathfrak{s} \setminus \{(\mathsf{prog} \sqsubseteq \neg \mathit{Final})\} \cup \{(\mathsf{prog} \sqsubseteq \mathit{Final})\}$ and
  - $\mathfrak{s}'_{\mathsf{s}} = \mathfrak{s}_{\mathsf{s}} \setminus \{(\mathsf{prog} \sqsubseteq \neg \mathit{Final})\} \cup \{(\mathsf{prog} \sqsubseteq \mathit{Final})\}$, and
  - $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$;

(iii) there is *no* guarded action contained in $\mathsf{head}(\rho)$ that is executable in the static type $\{\psi \in \mathcal{C}_{\mathsf{K}} \mid \iota_{\mathcal{C}}(\psi) \in \lambda_{\mathfrak{T}_{\mathcal{P}}}(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho)\}$ and we have

$$\mathfrak{s}' = \mathfrak{s} \setminus \{(\mathsf{prog} \sqsubseteq \neg \mathit{Fail})\} \cup \{(\mathsf{prog} \sqsubseteq \mathit{Fail})\}$$
$$\mathfrak{s}'_{\mathsf{s}} = \mathfrak{s}_{\mathsf{s}} \setminus \{(\mathsf{prog} \sqsubseteq \neg \mathit{Fail})\} \cup \{(\mathsf{prog} \sqsubseteq \mathit{Fail})\}$$

and $\rho = \rho'$.

▲

Initially, the knowledge base

$$\mathcal{K} \cup \{(\mathsf{prog} \sqsubseteq \neg \mathit{Final}), (\mathsf{prog} \sqsubseteq \neg \mathit{Fail})\}$$

describes everything that is known. It is the representation of the initial knowledge in the transition system. For an abstract state of the form $(\mathfrak{s}, \mathfrak{s}_{\mathsf{s}}, \rho)$ the corresponding subjective label is obtained by answering epistemic projection queries w.r.t. the empty action sequence, where $\mathfrak{s}_{\mathsf{s}}$ is treated as the current knowledge base.

A labeled abstract state is progressed by choosing a next possible action determined by the subjective label, by adding the sensed Boolean KB to the knowledge state and by updating the remaining program. In case the termination or failing action is executed also the static type of the world state is updated.

It can be shown that $\mathfrak{I}_{\mathcal{P}}$ and $\mathfrak{T}_{\mathcal{P}}$ are $\mathcal{C}$-bisimilar. $\mathfrak{T}_{\mathcal{P}}$ is of exponential size and can be computed in exponential time. This leads to the following result. We omit further details of the proof.

Theorem 7.28. *The verification problem for $\mathcal{ALCOK}$-CTL* formulas and knowledge-based $\mathcal{ALCOK}$-ConGolog programs over purely sensing actions is* ExpTime*-complete.*

7.3 Programs over Ground Actions with Conditional Effects

In this section we consider the case where actions might have conditional effects. We prove decidability of the verification problem for programs over conditional ground actions.

We consider knowledge-based $\mathcal{ALCOK}$-ConGolog programs of the form $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}), \delta)$, where

- $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ is an unrestricted epistemic $\mathcal{ALCO}$-action theory for a finite set of ground actions $\boldsymbol{A}$, and

- δ is a pick-free program expression over $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})$.

In addition we have given a proper $\mathcal{ALCOK}$-context $\mathcal{C} = \mathcal{C}_{\mathsf{o}} \uplus \mathcal{C}_{\mathbf{K}}$ for $\mathcal{P}$ partitioned into a objective and subjective part as before.

Due to conditional effects of an action it might happen that the outcome of an action is unknown. An action execution can lead to different changes on different possible worlds. This requires an abstraction technique that takes the different types of the possible worlds into account. The general idea is to finitely represent the knowledge state in terms of the possible static types w.r.t. an appropriate objective context. This objective context is constructed such that from the set of all possible *objective* static types the *subjective* static type of the represented knowledge state can be derived. To represent how the possible worlds evolve we consider the corresponding dynamic types.

First, we define the set of all instance functions with a domain that consists of all epistemic concepts, roles and KBs occurring in the context.

Definition 7.29. Let $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be an $\mathcal{ALCOK}$-ConGolog program as described above, $\mathcal{C}$ a proper $\mathcal{ALCOK}$-context for $\mathcal{P}$, $\mathcal{F}$ the set of all relevant concept and role names in $\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})$, $\mathsf{Obj}(\Sigma_{\mathsf{s}})$ the set of all relevant object names mentioned in Σ_{s} and $\mathsf{Nom}(\Sigma_{\mathsf{s}})$ the set of all nominal concepts as defined in (6.3).

With $\mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$ we denote the set of all (sub-)concepts of the form $\mathbf{K}D$ and with $\mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$ the set of all Boolean sub-KBs of the form $\mathbf{K}\varphi$ mentioned in $\mathcal{C}$. The *set of all relevant instance functions w.r.t. $\mathcal{C}$*, denoted by $\mathfrak{F}_{\mathcal{C}}$, consists of all total functions κ of the form

$$\kappa : \mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C}) \cup (\mathsf{Nom}(\Sigma_{\mathsf{s}}) \times (\mathcal{F} \cap \mathsf{N_R})) \cup \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C}) \to 2^{\mathsf{Nom}(\Sigma_{\mathsf{s}})} \cup \{\textsc{True}, \textsc{False}\}, \tag{7.8}$$

where

- each $\mathbf{K}D \in \mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$ is mapped to a set $\kappa(\mathbf{K}D) \subseteq \mathsf{Nom}(\Sigma_{\mathsf{s}})$,
- each pair $(X, P) \in (\mathsf{Nom}(\Sigma_{\mathsf{s}}) \times (\mathcal{F} \cap \mathsf{N_R}))$ to a set $\kappa(X, P) \subseteq \mathsf{Nom}(\Sigma_{\mathsf{s}})$, and
- each $\mathbf{K}\varphi \in \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$ to $\kappa(\mathbf{K}\varphi) \in \{\textsc{True}, \textsc{False}\}$.

▲

We sometimes also consider partial instance functions that are only defined for a subset of the domain specified in (7.8).

Consider an epistemic concept or KB of the form $\mathbf{K}X$ and an instance function κ. The operator $[\![\cdot,\cdot]\!]$ (Definition 6.47) first rewrites X and then applies the instance function to $\mathbf{K}[\![X, \kappa]\!]$. We define a modified version of the rewriting operator that produces an equivalent result but applies the instance function directly to $\mathbf{K}X$ without first applying the rewriting operator to X. This is done to ensure that an instance function with a domain specified in (7.8) is sufficient to obtain an objective rewriting.

Definition 7.30. Let $\mathcal{C}$ and $\mathfrak{F}_{\mathcal{C}}$ be as above, and let C be a sub-concept occurring in $\mathcal{C}$, ψ a Boolean sub-KB occurring in $\mathcal{C}$ and $\kappa \in \mathfrak{F}_{\mathcal{C}}$. The *rewriting of C w.r.t. κ*, denoted by $||C, \kappa||$, is an $\mathcal{ALCO}$-concept defined by induction on the structure of C in exactly the same way $[\![C, \kappa]\!]$ is defined except for the case where C is of the form $\mathbf{K}D$. In this case we have

$$||\mathbf{K}D, \kappa|| := \bigsqcup \kappa(\mathbf{K}D).$$

Similarly, the *rewriting of ψ w.r.t. κ*, denoted by $\|\psi,\kappa\|$, is defined inductively in the same way as $[\![\psi,\kappa]\!]$ except for the case where ψ is of the form $\mathbf{K}\varphi$. In this case we have

$$\|\mathbf{K}\varphi,\kappa\| := \kappa(\mathbf{K}\varphi).$$

▲

The result $\|C,\kappa\|$ and $\|\psi,\kappa\|$ is well-defined also for partial instance functions κ, if κ is defined for all sub-concepts $\mathbf{K}D$ in C or in ψ, respectively, for all sub-KBs of ψ of the form $\mathbf{K}\varphi$ and for all pairs $(X,P)\in(\mathsf{Nom}(\Sigma_\mathsf{s})\times(\mathcal{F}\cap\mathsf{N_R}))$, where $\mathbf{K}P$ occurs in C or ψ, respectively. Both operators $[\![\cdot,\cdot]\!]$ and $\|\cdot,\cdot\|$ give the same result if instantiated with the instance function for a knowledge state.

Lemma 7.31. *Let $\mathcal{P}=(\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}),\delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program over ground actions, $\mathfrak{I}_\mathcal{P}=(Q_\mathcal{P},I_\mathcal{P},\hookrightarrow_\mathcal{P},\lambda_\mathcal{P})$ the induced transition system, $\mathcal{C}$ a proper $\mathcal{ALCOK}$-context for $\mathcal{P}$ and $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle\in Q_\mathcal{P}$ a program state reachable from an initial state in $\mathfrak{I}_\mathcal{P}$. Furthermore, let $\kappa_\mathcal{W}$ be the instance function for $\mathcal{W}$ (Def. 6.46) and let $\kappa'_\mathcal{W}$ be the restriction of $\kappa_\mathcal{W}$ to the domain according to (7.8).*

For each sub-concept C and Boolean sub-KB φ occurring in $\mathcal{C}$ it holds that

$$[\![C,\kappa_\mathcal{W}]\!]=\|C,\kappa'_\mathcal{W}\| \text{ and } [\![\varphi,\kappa_\mathcal{W}]\!]=\|\varphi,\kappa'_\mathcal{W}\|.$$

The instance functions in $\mathfrak{F}_\mathcal{C}$ provide enough information to rewrite concepts and KBs occurring in $\mathcal{C}$ into equivalent objective formulas. To obtain the relevant objective context we consider all possible rewritings with instance functions from $\mathfrak{F}_\mathcal{C}$. The result is called *knowledge closure*.

Definition 7.32. Let $\mathcal{C}=\mathcal{C}_\mathsf{o}\uplus\mathcal{C}_\mathbf{K}$ be an $\mathcal{ALCOK}$-context over $\mathcal{F}$, where all Boolean KBs in $\mathcal{C}_\mathsf{o}$ are objective and all Boolean KBs in $\mathcal{C}_\mathbf{K}$ are subjective, and let $\mathsf{Obj}(\Sigma_\mathsf{s})$ be the finite set of all relevant object names, and $\mathfrak{F}_\mathcal{C}$ the set of all relevant instance functions w.r.t. $\mathcal{C}$. Furthermore, let $c\in\mathsf{N_O}\setminus\mathsf{Obj}(\Sigma_\mathsf{s})$ be an arbitrary but fixed object name not occurring in $\mathsf{Obj}(\Sigma_\mathsf{s})$.

The *knowledge closure of $\mathcal{C}$*, denoted by $\widehat{\mathcal{C}}$, is the following object context

$$\begin{aligned}\widehat{\mathcal{C}} := \mathcal{C}_\mathsf{o} \cup &\\ &\{(\neg)(\|\psi,\kappa\|)\mid \psi\in\mathcal{C}_\mathbf{K},\kappa\in\mathfrak{F}_\mathcal{C}\}\cup\\ &\big\{(\neg)\big((o,o')\sqsubseteq P\big)\,\big|\, P\in\mathcal{F}\cap\mathsf{N_R},(o,o')\in(\mathsf{Obj}(\Sigma_\mathsf{s})\cup\{c\})\times(\mathsf{Obj}(\Sigma_\mathsf{s})\cup\{c\})\big\}\cup\\ &\big\{(\neg)(o\sqsubseteq(\|D,\kappa\|))\,\big|\,\mathbf{K}D\in\mathsf{sub}^\mathsf{C}_\mathbf{K}(\mathcal{C}),\kappa\in\mathfrak{F}_\mathcal{C},o\in\mathsf{Obj}(\Sigma_\mathsf{s})\cup\{c\}\big\}\cup\\ &\big\{(\neg)(\|\varphi,\kappa\|)\,\big|\,\mathbf{K}\varphi\in\mathsf{sub}^\mathsf{KB}_\mathbf{K}(\mathcal{C}),\kappa\in\mathfrak{F}_\mathcal{C}\big\}.\end{aligned}$$

▲

We show that the static type of an epistemic interpretation $(\mathcal{I},\mathcal{W})$ w.r.t. $\mathcal{C}$ is uniquely determined by the set of static types of the worlds in $\mathcal{W}$ w.r.t. the knowledge closure $\widehat{\mathcal{C}}$. First, we define an instance function for a given set of static types w.r.t. $\widehat{\mathcal{C}}$.

Definition 7.33. Let $\mathcal{C}=\mathcal{C}_\mathsf{o}\uplus\mathcal{C}_\mathbf{K}$, $\mathsf{Obj}(\Sigma_\mathsf{s})$, $\mathcal{F}$ and $\widehat{\mathcal{C}}$ be as above and let $\mathfrak{S}_{\widehat{\mathcal{C}}}$ be the set of all static types w.r.t. $\widehat{\mathcal{C}}$ and $\mathcal{S}\subseteq\mathfrak{S}_{\widehat{\mathcal{C}}}$ a subset.

The *corresponding instance function* $\kappa_{\mathcal{S}} \in \mathfrak{F}_{\mathcal{C}}$ is defined as follows. Let $\{o\} \in \mathsf{Nom}(\Sigma_{\mathsf{s}})$ with $o \in \mathsf{Obj}(\Sigma_{\mathsf{s}})$ and $P \in \mathcal{F} \cap \mathsf{N_R}$. We define

$$\begin{aligned}
\kappa_{\mathcal{S}}(\{o\},P) := {} & \{\{a\} \in \mathsf{Nom}(\Sigma_{\mathsf{s}}) \mid a \in \mathsf{Obj}(\Sigma_{\mathsf{s}}), ((o,a) \sqsubseteq P) \in \mathfrak{s} \text{ for all } \mathfrak{s} \in \mathcal{S}\} \cup \\
& \{\neg N \mid ((o,c) \sqsubseteq P) \in \mathfrak{s} \text{ for all } \mathfrak{s} \in \mathcal{S}\}. \\
\kappa_{\mathcal{S}}(\neg N,P) := {} & \{\{a\} \in \mathsf{Nom}(\Sigma_{\mathsf{s}}) \mid a \in \mathsf{Obj}(\Sigma_{\mathsf{s}}), ((c,a) \sqsubseteq P) \in \mathfrak{s} \text{ for all } \mathfrak{s} \in \mathcal{S}\} \cup \\
& \{\neg N \mid ((c,c) \sqsubseteq P) \in \mathfrak{s} \text{ for all } \mathfrak{s} \in \mathcal{S}\}.
\end{aligned}$$

On $\mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$ and $\mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$ the function $\kappa_{\mathcal{S}}$ is defined by induction on the nesting depth of $\mathbf{K}$ symbols. Let $\mathbf{K}C \in \mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$, where C is objective. We define

$$\begin{aligned}
\kappa_{\mathcal{S}}(\mathbf{K}C) := {} & \Big\{\{a\} \in \mathsf{Nom}(\Sigma_{\mathsf{s}}) \mid a \in \mathsf{Obj}(\Sigma_{\mathsf{s}}), (a \sqsubseteq C) \in \bigcap \mathcal{S}\Big\} \\
& \Big\{\neg N \;\Big|\; (c \sqsubseteq C) \in \bigcap \mathcal{S} \text{ for some } c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_{\mathsf{s}})\Big\}.
\end{aligned}$$

Let $\mathbf{K}D \in \mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$ and assume $\kappa_{\mathcal{S}}$ is already defined for all concepts $\mathbf{K}C$ occurring in D and for all pairs in $(\mathsf{Nom}(\Sigma_{\mathsf{s}}) \times (\mathcal{F} \cap \mathsf{N_R}))$. We define

$$\begin{aligned}
\kappa_{\mathcal{S}}(\mathbf{K}D) := {} & \Big\{\{a\} \in \mathsf{Nom}(\Sigma_{\mathsf{s}}) \;\Big|\; a \in \mathsf{Obj}(\Sigma_{\mathsf{s}}), (a \sqsubseteq (\|D,\kappa_{\mathcal{S}}\|)) \in \bigcap \mathcal{S}\Big\} \\
& \Big\{\neg N \;\Big|\; (c \sqsubseteq (\|D,\kappa_{\mathcal{S}}\|)) \in \bigcap \mathcal{S} \text{ for some } c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_{\mathsf{s}})\Big\}.
\end{aligned}$$

Let $\mathbf{K}\varrho \in \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$, where ϱ is objective. We define

$$\kappa_{\mathcal{S}}(\mathbf{K}\varrho) := \begin{cases} \textsc{True} & \text{if } \varrho \in \bigcap \mathcal{S}; \\ \textsc{False} & \text{otherwise.} \end{cases}$$

Let $\mathbf{K}\varphi \in \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$ and assume $\kappa_{\mathcal{S}}$ is defined for all concepts of the form $\mathbf{K}D$ occurring in φ, for all pairs $(\mathsf{Nom}(\Sigma_{\mathsf{s}}) \times (\mathcal{F} \cap \mathsf{N_R}))$ and for all Boolean KBs of the form $\mathbf{K}\varrho$ occurring in φ. We define

$$\kappa_{\mathcal{S}}(\mathbf{K}\varphi) := \begin{cases} \textsc{True} & \text{if } \|\varphi,\kappa_{\mathcal{S}}\| \in \bigcap \mathcal{S}; \\ \textsc{False} & \text{otherwise.} \end{cases}$$

▲

It can be shown that $\kappa_{\mathcal{S}}$ is a well-defined instance function with $\kappa_{\mathcal{S}} \in \mathfrak{F}_{\mathcal{C}}$.

Lemma 7.34. *Let $\mathcal{C}$ and $\widehat{\mathcal{C}}$ be as above and $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle$ a reachable program state in the transition system induced by $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}),\delta)$ and let $\mathcal{S} := \big\{\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathcal{J}) \;\big|\; \mathcal{J} \in \mathcal{W}\big\}$. Furthermore, let $\kappa_{\mathcal{W}}$ be the instance function for $\mathcal{W}$ and $\kappa'_{\mathcal{W}}$ the restriction of $\kappa_{\mathcal{W}}$ to the domain*

$$\mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C}) \cup (\mathsf{Nom}(\Sigma_{\mathsf{s}}) \times (\mathcal{F} \cap \mathsf{N_R})) \cup \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C}).$$

It holds that

$$\kappa_{\mathcal{S}} = \kappa'_{\mathcal{W}}.$$

Proof. Let $(\{o\},P) \in \mathsf{Nom}(\Sigma_{\mathsf{s}}) \times (\mathcal{F} \cap \mathsf{N_R})$ with $o \in \mathsf{N_O}$ and let $a \in \mathsf{Obj}(\Sigma_{\mathsf{s}})$. It holds that

$\{a\} \in \kappa_{\mathcal{W}}(\{o\}, P)$

iff $(o, a) \in (\mathbf{K}P)^{\mathcal{W}}$

iff $\mathcal{J} \models (o, a) \sqsubseteq P$ for all $\mathcal{J} \in \mathcal{W}$

iff $((o, a) \sqsubseteq P) \in \mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathcal{J})$ for all $\mathcal{J} \in \mathcal{W}$ (by definition of $\widehat{\mathcal{C}}$)

iff $((o, a) \sqsubseteq P) \in \mathfrak{s}$ for all $\mathfrak{s} \in \mathcal{S}$

iff $\{a\} \in \kappa_{\mathcal{S}}(\{o\}, P)$.

Furthermore, it holds that $\neg N \in \kappa_{\mathcal{W}}(\{o\}, P)$

iff there exists $b \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that $(o, b) \in (\mathbf{K}P)^{\mathcal{W}}$

iff $(o, a) \in (\mathbf{K}P)^{\mathcal{W}}$ for all $a \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ (by Lemma 6.42)

iff there exists $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that $((o, c) \sqsubseteq P) \in \mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathcal{J})$ for all $\mathcal{J} \in \mathcal{W}$ (by definition of $\widehat{\mathcal{C}}$)

iff there exists $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that $((o, c) \sqsubseteq P) \in \mathfrak{s}$ for all $\mathfrak{s} \in \mathcal{S}$

iff $\neg N \in \kappa_{\mathcal{S}}(\{o\}, P)$.

With similar arguments it can be shown that $\kappa_{\mathcal{W}}(\neg N, P) = \kappa_{\mathcal{S}}(\neg N, P)$. For the subset $\mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C}) \cup \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$ of the domain the proof works by induction on the nesting depth of $\mathbf{K}$-symbols. For the base case let $\mathbf{K}D \in \mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$ with D being objective. It holds that $(a \sqsubseteq D) \in \widehat{\mathcal{C}}$ for all $a \in \mathsf{Obj}(\Sigma_\mathsf{s})$ and $(c \sqsubseteq D) \in \widehat{\mathcal{C}}$ for some $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ by definition of $\widehat{\mathcal{C}}$. With $\mathcal{S} = \left\{\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathcal{J}) \,\middle|\, \mathcal{J} \in \mathcal{W}\right\}$ it directly follows that $\kappa_{\mathcal{S}}(\mathbf{K}D) = \kappa_{\mathcal{W}}(\mathbf{K}D)$.

For the induction step consider an epistemic concept $\mathbf{K}C \in \mathsf{sub}^{\mathsf{C}}_{\mathbf{K}}(\mathcal{C})$. For all sub-concepts $\mathbf{K}D$ occurring in C we assume by induction $\kappa_{\mathcal{S}}(\mathbf{K}D) = \kappa_{\mathcal{W}}(\mathbf{K}D)$ and for all roles $(X, P) \in (\mathsf{Nom}(\Sigma_\mathsf{s}) \times (\mathcal{F} \cap \mathsf{N_R}))$ we have $\kappa_{\mathcal{S}}(X, P) = \kappa_{\mathcal{W}}(X, P)$ as shown above. It follows that

$$\|C, \kappa_{\mathcal{S}}\| = \|C, \kappa_{\mathcal{W}}\| \overset{\text{L. 7.31}}{=} [\![C, \kappa_{\mathcal{W}}]\!] \tag{7.9}$$

and it holds that $(a \sqsubseteq \|C, \kappa_{\mathcal{S}}\|) \in \widehat{\mathcal{C}}$ for all $a \in \mathsf{Obj}(\Sigma_\mathsf{s})$ and $(c \sqsubseteq \|C, \kappa_{\mathcal{S}}\|) \in \widehat{\mathcal{C}}$ for some $c \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ by definition of $\widehat{\mathcal{C}}$. Using (7.9) and Lemma 6.48 it can be shown that

$$\kappa_{\mathcal{S}}(\mathbf{K}C) = \kappa_{\mathcal{W}}(\mathbf{K}C).$$

Similarly, one can show that $\kappa_{\mathcal{S}}(\mathbf{K}\psi) = \kappa_{\mathcal{W}}(\mathbf{K}\psi)$ is true for all $\mathbf{K}\psi \in \mathsf{sub}^{\mathsf{KB}}_{\mathbf{K}}(\mathcal{C})$. □

We can now define satisfaction of a Boolean KB from the context in an abstraction of an epistemic interpretation in terms of static types w.r.t. the knowledge closure of the context. Let $\mathcal{S} \subseteq \mathfrak{S}_{\widehat{\mathcal{C}}}$, $\mathfrak{s} \in \mathcal{S}$ and $\psi \in \mathcal{C}$. *Satisfaction of ψ in $(\mathfrak{s}, \mathcal{S})$*, denoted by $(\mathfrak{s}, \mathcal{S}) \models_{\mathcal{C}} \psi$, is defined by:

$$(\mathfrak{s}, \mathcal{S}) \models_{\mathcal{C}} \psi \text{ iff } \|\psi, \kappa_{\mathcal{S}}\| \in \mathfrak{s}.$$

Lemma 7.35. *Let $\mathcal{C}$ and $\widehat{\mathcal{C}}$ be as above and $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle$ a reachable program state in the transition system induced by $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}),\delta)$ and let $\mathcal{S} := \{\text{s-type}_{\widehat{\mathcal{C}}}(\mathcal{J}) \mid \mathcal{J} \in \mathcal{W}\}$, $\mathfrak{s} :=$ $\text{s-type}_{\widehat{\mathcal{C}}}(\mathcal{I})$ and $\psi \in \mathcal{C}$. It holds that*

$$(\mathcal{I},\mathcal{W}) \Vdash \psi \textit{ iff } (\mathfrak{s},\mathcal{S}) \models_{\mathcal{C}} \psi.$$

We introduce a few more auxiliary notions needed for the abstraction. The abstraction of a program state consists of the dynamic type of each possible world, the set of accumulated effects for each individual dynamic type and the remaining program expression.

Definition 7.36. Let $\mathcal{P} = (\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}),\delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program over ground actions with $\Sigma_{\mathsf{s}} = (\mathcal{K},\boldsymbol{A},\mathsf{pre},\mathsf{eff},\mathsf{sense})$, where $\boldsymbol{A}$ is a finite set of ground actions, and let $\mathcal{C} = \mathcal{C}_{\mathsf{O}} \uplus \mathcal{C}_{\mathsf{K}}$ be a proper $\mathcal{ALCOK}$-context $\mathcal{P}$, $\widehat{\mathcal{C}}$ the knowledge closure of $\mathcal{C}$ and $\text{D-Types}(\widehat{\mathcal{C}},\mathsf{Lit}(\Sigma_{\mathsf{s}}))$ the set of all dynamic types w.r.t. $\widehat{\mathcal{C}}$ and $\mathsf{Lit}(\Sigma_{\mathsf{s}})$.

An *abstract program state of $\mathcal{P}$ w.r.t. $\mathcal{C}$* is a tuple of the form

$$(\mathfrak{d},\mathfrak{K},\rho)$$

with $\mathfrak{K} \subseteq \text{D-Types}(\widehat{\mathcal{C}},\mathsf{Lit}(\Sigma_{\mathsf{s}})) \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$, $\mathfrak{d} \in \mathfrak{K}$ and $\rho \in \mathsf{sub}(\delta)$. For a tuple

$$(\mathfrak{t},\mathsf{L}) \in \text{D-Types}(\widehat{\mathcal{C}},\mathsf{Lit}(\Sigma_{\mathsf{s}})) \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$$

we define the corresponding static type

$$\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t},\mathsf{L}) := \{\psi \mid (\psi,\mathsf{L}) \in \mathfrak{t}\}.$$

Let $(\mathfrak{d},\mathfrak{K},\rho)$ be an abstract program state and

$$\mathfrak{s} := \text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}) \text{ and } \mathcal{S} := \{\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t},\mathsf{L}) \mid (\mathfrak{t},\mathsf{L}) \in \mathfrak{K}\}.$$

We say that a guarded action $\psi_1?;\cdots;\psi_n?;\boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ is executable in $(\mathfrak{d},\mathfrak{K})$ iff $(\mathfrak{s},\mathcal{S}) \models_{\mathcal{C}} \psi_i$ for all $i = 1,\ldots,n$ and $(\mathfrak{s},\mathcal{S}) \models_{\mathcal{C}} \mathsf{K}\big(\bigwedge \mathsf{pre}(\boldsymbol{\alpha})\big)$.

Let $\boldsymbol{\alpha} \in \boldsymbol{A}$ and $\mathfrak{s} \in \mathfrak{S}_{\widehat{\mathcal{C}}}$. The *effects of executing $\boldsymbol{\alpha}$ in a world of static type* $\mathfrak{s}$, denoted by $\mathsf{eff}_{\widehat{\mathcal{C}}}(\mathfrak{s},\boldsymbol{\alpha})$, is defined as the following subset of $\mathsf{Lit}(\Sigma_{\mathsf{s}})$:

$$\mathsf{eff}_{\widehat{\mathcal{C}}}(\mathfrak{s},\boldsymbol{\alpha}) := \big\{\langle F,X\rangle^{\pm} \bigm| \psi \rhd \langle F,X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}), \psi \in \mathfrak{s}\big\}.$$

The sensing compatibility relation is lifted to the level of static types as well. We define

$$\sim_{\mathsf{s}}^{\widehat{\mathcal{C}}} := \big\{(\mathfrak{s},\boldsymbol{\alpha},\mathfrak{s}') \in \mathfrak{S}_{\widehat{\mathcal{C}}} \times \boldsymbol{A} \times \mathfrak{S}_{\widehat{\mathcal{C}}} \bigm| \mathsf{sense}(\boldsymbol{\alpha}) \in \mathfrak{s} \text{ iff } \mathsf{sense}(\boldsymbol{\alpha}) \in \mathfrak{s}'\big\}.$$

Next, we define the lifting of the transition relation induced by $\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}})$ to the abstraction of epistemic interpretations. Let $\boldsymbol{\alpha} \in \boldsymbol{A}$, $\mathfrak{K},\mathfrak{K}' \subseteq \text{D-Types}(\widehat{\mathcal{C}},\mathsf{Lit}(\Sigma_{\mathsf{s}})) \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$ and $\mathfrak{d} \in \mathfrak{K}$ and $\mathfrak{d}' \in \mathfrak{K}'$. We say that the *execution of $\boldsymbol{\alpha}$ in* $(\mathfrak{d},\mathfrak{K})$ *transforms* $(\mathfrak{d},\mathfrak{K})$ *into* $(\mathfrak{d}',\mathfrak{K}')$, written as

$$(\mathfrak{d},\mathfrak{K}) \Longrightarrow_{\widehat{\mathcal{C}}}^{\boldsymbol{\alpha}} (\mathfrak{d}',\mathfrak{K}'),$$

iff the following conditions are satisfied

(P1) $\mathfrak{d}' = (\mathfrak{t}, \mathsf{L}')$ with $\mathfrak{d} = (\mathfrak{t}, \mathsf{L})$ and

$$\mathsf{L}' = \big(\mathsf{L} \setminus \neg\mathsf{eff}_{\widehat{\mathcal{C}}}(\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{d}), \boldsymbol{\alpha})\big) \cup \mathsf{eff}_{\widehat{\mathcal{C}}}(\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{d}), \boldsymbol{\alpha});$$

(P2) it holds that $(\mathfrak{t}', \mathsf{E}') \in \mathfrak{K}'$ iff there exists $(\mathfrak{t}', \mathsf{E}) \in \mathfrak{K}$ such that

$$\big(\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{d}), \boldsymbol{\alpha}, \mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{E})\big) \in \sim_{\mathsf{s}}^{\widehat{\mathcal{C}}}$$

and

$$\mathsf{E}' = \big(\mathsf{E} \setminus \neg\mathsf{eff}_{\widehat{\mathcal{C}}}(\mathfrak{s}', \boldsymbol{\alpha})\big) \cup \mathsf{eff}_{\widehat{\mathcal{C}}}(\mathfrak{s}', \boldsymbol{\alpha})$$

with $\mathfrak{s}' = \mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{E})$.

The extension of $\Longrightarrow_{\widehat{\mathcal{C}}}^{\boldsymbol{\alpha}}$ for some $\boldsymbol{\alpha} \in \boldsymbol{A}$ to a relation $\Longrightarrow_{\widehat{\mathcal{C}}}^{\sigma}$ for some sequence $\sigma \in \boldsymbol{A}^*$ is defined in the usual way.

We also extend the definition of $\mathsf{eff}_{\widehat{\mathcal{C}}}(\mathfrak{s}, \boldsymbol{\alpha})$ to dynamic types and sequences of ground actions. Let $\mathfrak{t} \in \mathsf{d\text{-}type}_{\widehat{\mathcal{C}}}^{\mathsf{loc}}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$ and $\sigma \in \boldsymbol{A}^*$. The *set of effects of executing σ in* $\mathfrak{t}$, denoted by $\mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \sigma)$, is defined by induction on the length of σ as follows:

$$\mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \langle\rangle) := \emptyset$$

and for $\theta \in \boldsymbol{A}^*$, $\mathsf{L}_\theta = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \theta)$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$ we define

$$\mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \theta \cdot \boldsymbol{\alpha}) := \mathsf{L}_\theta \setminus \neg\mathsf{eff}_{\widehat{\mathcal{C}}}(\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{L}_\theta), \boldsymbol{\alpha}) \cup \mathsf{eff}_{\widehat{\mathcal{C}}}(\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{L}_\theta), \boldsymbol{\alpha}).$$

Furthermore, the sensing compatibility relation is extended to dynamic types and sequences of actions as well. Let $\sigma \in \boldsymbol{A}^*$ and $\mathfrak{t}, \mathfrak{t}' \in \mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$. *Sensing compatibility of* $\mathfrak{t}$ *and* $\mathfrak{t}'$ *w.r.t.* σ, denoted by $\mathfrak{t} \asymp_{\langle\rangle}^{\widehat{\mathcal{C}}} \mathfrak{t}'$, is defined by induction on the length of σ as follows:

$$\mathfrak{t} \asymp_{\langle\rangle}^{\widehat{\mathcal{C}}} \mathfrak{t}'$$

is true for arbitrary pairs of dynamic types; and if σ is of the form $\theta \cdot \boldsymbol{\alpha}$ we have

$$\mathfrak{t} \asymp_{\theta\cdot\boldsymbol{\alpha}}^{\widehat{\mathcal{C}}} \mathfrak{t}' \text{ iff } \mathfrak{t} \asymp_{\theta}^{\widehat{\mathcal{C}}} \mathfrak{t}' \text{ and } \big(\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \theta)), \boldsymbol{\alpha}, \mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \theta))\big) \in \sim_{\mathsf{s}}^{\widehat{\mathcal{C}}}.$$

▲

Given a pair

$$(\mathfrak{t}, \mathsf{L}) \in \mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_{\mathsf{s}})) \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$$

The type $\mathfrak{t}$ represents the dynamic type of an initial world and L is the set of accumulated effects of the actions executed so far in worlds of this type. L represents the current progress and points to the current static type denoted by $\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{L})$. Note that the abstract transition relation $\Longrightarrow_{\widehat{\mathcal{C}}}^{\boldsymbol{\alpha}}$ only updates the set of accumulated effects attached to a dynamic type. The dynamic type itself is not subject to any changes.

We show that the lifting of executability and transformation of ground actions to the level of types respects the underlying action semantics.

Lemma 7.37. *Let $\mathcal{P}$, Σ_{s} and $\widehat{\mathcal{C}}$ be as in Definition 7.36,*

$$\mathfrak{D}_{\mathsf{K}}(\Sigma_{\mathsf{s}}) = (\mathfrak{D} = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}), \sim_{\mathsf{s}})$$

the epistemic FO-DS induced by Σ_{s}, and $\mathfrak{T}_{\mathcal{P}} = (Q_{\mathcal{P}}, I_{\mathcal{P}}, \hookrightarrow_{\mathcal{P}}, \lambda_{\mathcal{P}})$ the transition system induced by $\mathcal{P}$, and let $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle \in Q_{\mathcal{P}}$ be a program state and $\langle(\mathcal{I}_0,\mathcal{M}(\mathcal{K})),\langle\rangle,\delta\rangle \in I_{\mathcal{P}}$ such that

$$\langle(\mathcal{I}_0,\mathcal{M}(\mathcal{K})),\langle\rangle,\delta\rangle \hookrightarrow_{\mathcal{P}}^{*} \langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle.$$

1. *It holds that*
$$\mathcal{I} = \mathcal{I}_0^{\,\mathsf{L}_\sigma} \textit{ with } \mathsf{L}_\sigma = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0),\sigma);$$
and for each $\mathcal{J} \in \mathcal{W}$ there exists a model $\mathcal{J}_0 \in \mathcal{M}(\mathcal{K})$ such that
$$\mathcal{J} = \mathcal{J}_0^{\,\mathsf{E}_\sigma} \textit{ with } \mathsf{E}_\sigma = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0),\sigma) \textit{ with } \textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\sigma} \textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0)$$

2. *Let $(\mathfrak{d}_{\mathcal{I}},\mathfrak{K}_{\mathcal{W}})$ be the abstraction of $(\mathcal{I},\mathcal{W})$ in $\langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle$ given by*
$$\mathfrak{d}_{\mathcal{I}} = (\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0),\mathsf{L}_\sigma) \textit{ with } \mathsf{L}_\sigma = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0),\sigma);$$
$$\mathfrak{K}_{\mathcal{W}} = \left\{ (\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0),\mathsf{E}_\sigma) \;\middle|\; \begin{array}{l} \mathcal{J}_0 \in \mathcal{M}(\mathcal{K}), \mathsf{E}_\sigma = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0),\sigma), \\ \textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\sigma} \textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0) \end{array} \right\}.$$
$(\mathfrak{d}_{\mathcal{I}},\mathfrak{K}_{\mathcal{W}})$ *has the following properties*
   a) *It holds that* $\textsf{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}_{\mathcal{I}}) = \textsf{s-type}_{\widehat{\mathcal{C}}}(\mathcal{I})$ *and*
$$\left\{\textsf{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}') \;\middle|\; \mathfrak{d}' \in \mathfrak{K}_{\mathcal{W}}\right\} = \left\{\textsf{s-type}_{\widehat{\mathcal{C}}}(\mathcal{J}) \mid \mathcal{J} \in \mathcal{W}\right\}.$$
   b) *A guarded action $\mathfrak{a} \in \mathsf{head}(\rho)$ is executable in $(\mathcal{I},\mathcal{W})$ iff $\mathfrak{a}$ is executable in $(\mathfrak{d}_{\mathcal{I}},\mathfrak{K}_{\mathcal{W}})$.*
   c) *Let $\mathfrak{d}_0 = (\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0),\emptyset)$ and $\mathfrak{K}_0 = \left\{(\textsf{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0),\emptyset) \;\middle|\; \mathcal{J}_0 \in \mathcal{M}(\mathcal{K})\right\}$. It holds that*
$$(\mathfrak{d}_0,\mathfrak{K}_0) \Longrightarrow^{\sigma}_{\widehat{\mathcal{C}}} (\mathfrak{d}_{\mathcal{I}},\mathfrak{K}_{\mathcal{W}}).$$

Proof. 1. We have
$$\langle(\mathcal{I}_0,\mathcal{M}(\mathcal{K})),\langle\rangle,\delta\rangle \hookrightarrow_{\mathcal{P}}^{*} \langle(\mathcal{I},\mathcal{W}),\sigma,\rho\rangle,$$
which implies $(\mathcal{I}_0,\mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathsf{K}}} (\mathcal{I},\mathcal{W})$. The proof is by induction on the length of σ. The base case with $\sigma = \langle\rangle$ is trivial. Assume σ is of the form $\theta \cdot \boldsymbol{\alpha}$. There exists an epistemic interpretation $(\mathcal{Y}_\theta,\mathcal{M}_\theta)$ such that
$$(\mathcal{I}_0,\mathcal{M}(\mathcal{K})) \Longrightarrow^{\theta}_{\mathfrak{D}_{\mathsf{K}}} (\mathcal{Y}_\theta,\mathcal{M}_\theta) \Longrightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}_{\mathsf{K}}} (\mathcal{I},\mathcal{W}).$$
It follows that $\mathcal{Y}_\theta \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{I}$. Thus, we have
$$\mathcal{I} = \mathcal{Y}_\theta^{\,\mathsf{L}_{\boldsymbol{\alpha}}} \text{ with } \mathsf{L}_{\boldsymbol{\alpha}} = \{\langle F,X\rangle^{\pm} \mid \psi \rhd \langle F,X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}), \mathcal{Y}_\theta \models \psi\}.$$

By induction we have

$$\mathcal{Y}_\theta = {\mathcal{I}_0}^{\mathsf{L}_\theta} \text{ with } \mathsf{L}_\theta = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \theta).$$

It follows that $\text{s-type}_{\widehat{\mathcal{C}}}(\mathcal{Y}_\theta) = \text{s-type}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \mathsf{L}_\theta)$ and hence

$$\mathsf{L}_{\boldsymbol{\alpha}} = \mathsf{eff}_{\widehat{\mathcal{C}}}(\text{s-type}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \mathsf{L}_\theta), \boldsymbol{\alpha}).$$

It follows that

$$\mathcal{I} = {\mathcal{I}_0}^{\mathsf{L}_\theta \setminus \neg \mathsf{L}_{\boldsymbol{\alpha}} \cup \mathsf{L}_{\boldsymbol{\alpha}}} \text{ with } (\mathsf{L}_\theta \setminus \neg \mathsf{L}_{\boldsymbol{\alpha}} \cup \mathsf{L}_{\boldsymbol{\alpha}}) = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \theta \cdot \boldsymbol{\alpha}).$$

Let $\mathcal{J} \in \mathcal{W}$. It follows that there exists $\mathcal{J}_\theta \in \mathcal{M}_\theta$ such that

$$\mathcal{Y}_\theta \sim^{\boldsymbol{\alpha}}_{\mathsf{s}} \mathcal{J}_\theta \text{ and } \mathcal{J}_\theta \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{J}.$$

By induction there exists $\mathcal{J}_0 \in \mathcal{M}(\mathcal{K})$ such that

$$\mathcal{J}_\theta = {\mathcal{J}_0}^{\mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \theta)} \text{ and } \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \preceq^{\widehat{\mathcal{C}}}_{\theta} \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0).$$

Using $\mathcal{J}_\theta \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{J}$ it can be shown that

$$\mathcal{J} = {\mathcal{J}_0}^{\mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \theta \cdot \boldsymbol{\alpha})}.$$

Now, $\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \preceq^{\widehat{\mathcal{C}}}_{\theta} \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0)$ and $\mathcal{J}_\theta \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{J}$ yields

$$\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \preceq^{\widehat{\mathcal{C}}}_{\theta \cdot \boldsymbol{\alpha}} \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0).$$

2. $\mathfrak{d}_{\mathcal{I}}$ and $\mathfrak{K}_{\mathcal{W}}$ as described in the claim are unique and always exist. 2a follows from 1. and Definition 7.36, and 2b is a consequence of 2a and 7.35.

   The proof of 2c is by induction on the length of σ. The case $\sigma = \langle\rangle$ is trivial. For the induction step assume σ to be of the form $\theta \cdot \boldsymbol{\alpha}$ with $\theta \in \boldsymbol{A}^*$ and $\boldsymbol{\alpha} \in \boldsymbol{A}$. There exists a state $\langle(\mathcal{Y}, \mathcal{N}), \theta, \zeta\rangle \in Q_{\mathfrak{T}_{\mathcal{P}}}$ such that

$$\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \hookrightarrow_{\mathcal{P}}{}^* \langle(\mathcal{Y}, \mathcal{N}), \theta, \zeta\rangle \hookrightarrow_{\mathcal{P}} \langle(\mathcal{I}, \mathcal{W}), \theta \cdot \boldsymbol{\alpha}, \rho\rangle$$

   Let

$$\mathfrak{d}_{\mathcal{Y}} := (\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \mathsf{L}_\theta) \text{ with } \mathsf{L}_\theta = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \theta);$$
$$\mathfrak{K}_{\mathcal{N}} = \left\{ (\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \mathsf{E}_\theta) \;\middle|\; \begin{array}{l} \mathcal{J}_0 \in \mathcal{M}(\mathcal{K}), \mathsf{E}_\theta = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \theta), \\ \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \preceq^{\widehat{\mathcal{C}}}_{\theta} \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0) \end{array} \right\}.$$

   The induction hypothesis yields $(\mathfrak{d}_0, \mathfrak{K}_0) \Longrightarrow^{\theta}_{\widehat{\mathcal{C}}} (\mathfrak{d}_{\mathcal{Y}}, \mathfrak{K}_{\mathcal{N}})$. We show that

$$(\mathfrak{d}_{\mathcal{Y}}, \mathfrak{K}_{\mathcal{N}}) \Longrightarrow^{\boldsymbol{\alpha}}_{\widehat{\mathcal{C}}} (\mathfrak{d}_{\mathcal{I}}, \mathfrak{K}_{\mathcal{W}}).$$

We have $\mathcal{Y} \Rightarrow^{\boldsymbol{\alpha}}_{\mathfrak{D}} \mathcal{I}$. Let $\mathsf{L}_{\boldsymbol{\alpha}} = \left\{ \langle F,X\rangle^{\pm} \,\middle|\, \psi \rhd \langle F,X\rangle^{\pm} \in \mathsf{eff}(\boldsymbol{\alpha}) \text{ and } \mathcal{Y} \models \psi \right\}$. It follows that

$$\mathsf{L}_{\boldsymbol{\alpha}} = \mathsf{eff}(\text{s-type}_{\widehat{\mathcal{C}}}(\mathcal{Y}), \boldsymbol{\alpha}) \overset{2a}{=} \mathsf{eff}(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}_{\mathcal{Y}}), \boldsymbol{\alpha})$$

Lemma 4.5 yields $\mathcal{I} = \mathcal{Y}^{\mathsf{L}_{\boldsymbol{\alpha}}}$. With $\mathcal{Y} = {\mathcal{I}_0}^{\mathsf{L}_\theta}$ and Lemma 4.6 it follows that

$$\mathcal{I} = {\mathcal{I}_0}^{\mathsf{L}_\theta \setminus \neg \mathsf{L}_{\boldsymbol{\alpha}} \cup\, \mathsf{L}_{\boldsymbol{\alpha}}}.$$

Hence, $\mathfrak{d}_{\mathcal{I}} = (\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \mathsf{L}_\theta \setminus \neg \mathsf{L}_{\boldsymbol{\alpha}} \cup \mathsf{L}_{\boldsymbol{\alpha}})$. Consequently, $\mathfrak{d}_{\mathcal{I}}$ satisfies (P1) required by the definition of "$\Longrightarrow^{\boldsymbol{\alpha}}_{\widehat{\mathcal{C}}}$".

Next, we show that $\mathfrak{K}_{\mathcal{W}}$ satisfies (P2). Let $(\mathfrak{t}, \mathsf{L}) \in \mathfrak{K}_{\mathcal{W}}$. We have to show that a pair $(\mathfrak{t}, \mathsf{E}) \in \mathfrak{K}_{\mathcal{N}}$ exists such that

$$\left(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}_{\mathcal{Y}}), \boldsymbol{\alpha}, \text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{E})\right) \in \sim^{\widehat{\mathcal{C}}}_{\mathsf{s}}$$

and

$$\mathsf{L} = \mathsf{E} \setminus \neg \mathsf{eff}(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{E}), \boldsymbol{\alpha}) \cup \mathsf{eff}(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}, \mathsf{E}), \boldsymbol{\alpha}).$$

By definition of $\mathfrak{K}_{\mathcal{W}}$ we have that $(\mathfrak{t}, \mathsf{L}) \in \mathfrak{K}_{\mathcal{W}}$ implies that there exists some $\mathcal{J}_0 \in \mathcal{M}(\mathcal{K})$ with

$$\mathfrak{t} = \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \mathsf{L} = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \theta \cdot \boldsymbol{\alpha}) \text{ and } \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\theta \cdot \boldsymbol{\alpha}} \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0).$$

It follows that $\mathsf{E} = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \theta)$ satisfies the property as required, i.e.

$$(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \mathsf{Eff}_{\widehat{\mathcal{C}}}(\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \theta)) \in \mathfrak{K}_{\mathcal{N}}.$$

For the proof of the other direction of (P2) consider $(\mathfrak{t}', \mathsf{E}_\theta) \in \mathfrak{K}_{\mathcal{N}}$ such that

$$(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}_{\mathcal{Y}}), \boldsymbol{\alpha}, \text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{E}_\theta)) \in \sim^{\widehat{\mathcal{C}}}_{\mathsf{s}}.$$

We have to show that $(\mathfrak{t}', \mathsf{E}_{\theta \cdot \boldsymbol{\alpha}}) \in \mathfrak{K}_{\mathcal{W}}$ with

$$\mathsf{E}_{\theta \cdot \boldsymbol{\alpha}} = \mathsf{E}_\theta \setminus \neg \mathsf{eff}_{\widehat{\mathcal{C}}}(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{E}_\theta), \boldsymbol{\alpha}) \cup \mathsf{eff}_{\widehat{\mathcal{C}}}(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{E}_\theta), \boldsymbol{\alpha}).$$

Since $(\mathfrak{t}', \mathsf{E}_\theta) \in \mathfrak{K}_{\mathcal{N}}$, we have that $\mathfrak{t}' = \text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0)$ for some $\mathcal{J}_0 \in \mathcal{M}(\mathcal{K})$ such that $\mathsf{E}_\theta = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \theta)$ and $\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\theta} \mathfrak{t}'$. It follows that

$$\mathsf{E}_{\theta \cdot \boldsymbol{\alpha}} = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \theta \cdot \boldsymbol{\alpha}).$$

With $(\text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{d}_{\mathcal{Y}}), \boldsymbol{\alpha}, \text{s-type}_{\widehat{\mathcal{C}}}(\mathfrak{t}', \mathsf{E}_\theta)) \in \sim^{\widehat{\mathcal{C}}}_{\mathsf{s}}$ and $\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\theta} \mathfrak{t}'$ it follows that

$$\text{d-type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\theta \cdot \boldsymbol{\alpha}} \mathfrak{t}'$$

Therefore, $(\mathfrak{t}', \mathsf{E}_{\theta \cdot \boldsymbol{\alpha}}) \in \mathfrak{K}_{\mathcal{W}}$.

□

We are now ready to define the abstraction of the transition system induced by $\mathcal{P}$. We consider the termination and failure extension $\Sigma_\mathsf{s} \uplus \{\epsilon, \mathfrak{f}\}$ of Σ_s but still use Σ_s in the definition for better readability.

Definition 7.38. Let $\mathcal{P} = (\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}), \delta)$ be a knowledge-based $\mathcal{ALCOK}$-ConGolog program over ground actions with $\Sigma_\mathsf{s} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$, where $\boldsymbol{A}$ is a finite set of ground actions, and let $\mathcal{C} = \mathcal{C}_\mathsf{o} \uplus \mathcal{C}_\mathbf{K}$ be a proper $\mathcal{ALCOK}$-context for $\mathcal{P}$, $\widehat{\mathcal{C}}$ the knowledge closure of $\mathcal{C}$, $\mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_\mathsf{s}))$ the set of all dynamic types w.r.t. $\widehat{\mathcal{C}}$ and $\mathsf{Lit}(\Sigma_\mathsf{s})$ and AP a finite set of atomic propositions with a bijection $\iota_\mathcal{C} : \mathcal{C} \to \mathsf{AP}$.

Let $\mathfrak{I}_\mathcal{P}$ be the transition system induced by $\mathcal{P}$. The *abstraction of* $\mathfrak{I}_\mathcal{P}$ *w.r.t.* $\mathcal{C}$ is a propositional transition system

$$\mathfrak{T}_\mathcal{P} = (Q_{\mathfrak{T}_\mathcal{P}}, I_{\mathfrak{T}_\mathcal{P}}, \hookrightarrow_{\mathfrak{T}_\mathcal{P}}, \lambda_{\mathfrak{T}_\mathcal{P}})$$

over AP, where

- $Q_{\mathfrak{T}_\mathcal{P}} := \big\{(\mathfrak{d}, \mathfrak{K}, \rho) \,\big|\, \mathfrak{K} \subseteq \mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_\mathsf{s})) \times 2^{\mathsf{Lit}(\Sigma_\mathsf{s})}, \mathfrak{d} \in \mathfrak{K}, \rho \in \mathsf{sub}(\delta)\big\}$;
- $I_{\mathfrak{T}_\mathcal{P}} := \big\{(\mathfrak{d}, \mathfrak{K}, \rho) \in Q_{\mathfrak{T}_\mathcal{P}} \,\big|\, \mathfrak{K} = \{(\mathfrak{t}, \emptyset) \mid (\varphi, \emptyset) \in \mathfrak{t} \text{ for all } \varphi \text{ occurring in } \mathcal{K}\}, \rho = \delta\big\}$;
- for each $(\mathfrak{d}, \mathfrak{K}, \rho) \in Q_{\mathfrak{T}_\mathcal{P}}$ we have

$$\lambda_{\mathfrak{T}_\mathcal{P}}(\mathfrak{d}, \mathfrak{K}, \rho) := \{\iota_\mathcal{C}(\psi) \mid (\mathfrak{s}, \mathcal{S}) \models_\mathcal{C} \psi, \psi \in \mathcal{C}\}, \text{ where}$$

  $\mathfrak{s} = \mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{d})$ and $\mathcal{S} = \{\mathsf{s\text{-}type}_{\widehat{\mathcal{C}}}(\mathfrak{d}') \mid \mathfrak{d}' \in \mathfrak{K}\}$.
- $\hookrightarrow_{\mathfrak{T}_\mathcal{P}} := \big\{\big((\mathfrak{d}, \mathfrak{K}, \rho), (\mathfrak{d}', \mathfrak{K}', \rho')\big) \in Q_{\mathfrak{T}_\mathcal{P}} \times Q_{\mathfrak{T}_\mathcal{P}} \text{ (i) or (ii) }\big\}$ with
  (i) there exists a guarded action $\mathfrak{a} = \psi_1?; \cdots; \psi_n?; \boldsymbol{\alpha} \in \mathsf{head}(\rho)$ for some $n \geq 0$ such that
    - $\mathfrak{a}$ is executable in $(\mathfrak{d}, \mathfrak{K})$, and
    - $(\mathfrak{d}, \mathfrak{K}) \overset{\alpha}{\Longrightarrow}_{\widehat{\mathcal{C}}} (\mathfrak{d}', \mathfrak{K}')$ and $\rho' \in \mathsf{tail}(\mathfrak{a}, \rho)$.

  (ii) there is *no* guarded action contained in $\mathsf{head}(\rho)$ that is executable in $(\mathfrak{d}, \mathfrak{K})$ and we have $(\mathfrak{d}, \mathfrak{K}) \overset{\mathfrak{f}}{\Longrightarrow}_{\widehat{\mathcal{C}}} (\mathfrak{d}', \mathfrak{K}')$ and $\rho = \rho'$.

▲

We show that $\mathfrak{T}_\mathcal{P} = (Q_{\mathfrak{T}_\mathcal{P}}, I_{\mathfrak{T}_\mathcal{P}}, \hookrightarrow_{\mathfrak{T}_\mathcal{P}}, \lambda_{\mathfrak{T}_\mathcal{P}})$ is a bisimilar propositional abstraction of $\mathcal{P}$. Let $\mathfrak{I}_\mathcal{P} = (Q_\mathcal{P}, I_\mathcal{P}, \hookrightarrow_\mathcal{P}, \lambda_\mathcal{P})$ be the transition system induced by $\mathcal{P}$. We define a relation

$$\simeq_\mathcal{C} \subseteq Q_\mathcal{P} \times Q_{\mathfrak{T}_\mathcal{P}}$$

such that

$$\langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle \simeq_\mathcal{C} (\mathfrak{d}, \mathfrak{K}, \rho) \text{ iff the following conditions are satisfied}$$

- there exists $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \in I_\mathcal{P}$ such that $\langle(\mathcal{I}_0, \mathcal{M}(\mathcal{K})), \langle\rangle, \delta\rangle \hookrightarrow_\mathcal{P}^* \langle(\mathcal{I}, \mathcal{W}), \sigma, \rho\rangle$ and
  - $\mathfrak{d} = (\mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \mathsf{L})$ and $\mathsf{L} = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0), \sigma)$;

– $$\mathfrak{K} = \left\{ (\mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \mathsf{E}) \;\middle|\; \begin{array}{l} \mathcal{J}_0 \in \mathcal{M}(\mathcal{K}), \mathsf{E} = \mathsf{Eff}_{\widehat{\mathcal{C}}}(\mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \sigma), \\ \mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{I}_0) \asymp^{\widehat{\mathcal{C}}}_{\sigma} \mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0) \end{array} \right\}.$$

- $\rho = \theta$.

It is easy to prove that $\simeq_{\mathcal{C}}$ is a $\mathcal{C}$-bisimulation using Lemma 7.35 and Lemma 7.37.

Lemma 7.39. *$\simeq_{\mathcal{C}}$ is a $\mathcal{C}$-bisimulation.*

For each abstract initial state $(\mathfrak{d}, \mathfrak{K}, \delta) \in I_{\mathfrak{T}_{\mathcal{P}}}$ it holds that

$$\mathfrak{K} = \left\{ (\mathsf{d\text{-}type}^{\mathsf{loc}}_{\widehat{\mathcal{C}}}(\mathcal{J}_0), \emptyset) \;\middle|\; \mathcal{J}_0 \in \mathcal{M}(\mathcal{K}) \right\}.$$

It follows that each abstract initial state is $\mathcal{C}$-bisimilar to an initial state in $I_{\mathcal{P}}$ and vice versa. Therefore, $\mathfrak{T}_{\mathcal{P}}$ is a finite $\mathcal{C}$-bsimilar propositional abstraction of $\mathfrak{I}_{\mathcal{P}}$.

Next, we count the number of states in $\mathfrak{T}_{\mathcal{P}}$ and determine an upper bound of the complexity of computing $\mathfrak{T}_{\mathcal{P}}$.

First, we observe that the set $\mathfrak{F}_{\mathcal{C}}$ of all relevant instance functions w.r.t. $\mathcal{C}$ is exponentially large in the size of $\mathcal{C}$ and Σ_{s}. Second, the knowledge closure $\widehat{\mathcal{C}}$ is therefore also at most exponentially large in the size of Σ_{s} and $\mathcal{C}$. Thus, the cardinality of

$$\widehat{\mathcal{C}} \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$$

is exponentially bounded as well. Consequently, there are at most double-exponentially many dynamic types in $\mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$. Realizability of a complete subset of $\widehat{\mathcal{C}} \times 2^{\mathsf{Lit}(\Sigma_{\mathsf{s}})}$ can be checked in 2EXPTIME using the method given in Chapter 4, Section 4.2. We obtain that $\mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$ is computable in 2EXPTIME. The knowledge state component of the abstract states in $\mathfrak{T}_{\mathcal{P}}$ consists of subsets of $\mathsf{D\text{-}Types}(\widehat{\mathcal{C}}, \mathsf{Lit}(\Sigma_{\mathsf{s}}))$. Therefore, there are at most triple-exponentially many states in $\mathfrak{T}_{\mathcal{P}}$. We obtain a 3EXPTIME upper bound for the verification problem.

Theorem 7.40. *The verification problem for $\mathcal{ALCOK}$-CTL* properties and knowledge-based $\mathcal{ALCOK}$-ConGolog programs over ground actions is decidable in* 3EXPTIME.

7.4 A Deciable Pick-Operator with Epistemic Guards

So far, we have only obtained decidability results for programs over ground actions. We now add a restricted version of the pick-operator. Consider the guarded pick expression used in Example 7.1:

$$\mathsf{pick}(x) \to \mathbf{K}((\mathsf{dev}, x) \sqsubseteq \mathit{HasFault})?; \mathtt{repair}(\mathsf{dev}, x);$$

$$\mathsf{pick}(x) \to \mathbf{K}(x \sqsubseteq \mathit{Fault}) \land \neg \mathbf{Kw}((\mathsf{dev}, x) \sqsubseteq \mathit{HasFault})?; \mathtt{sense\text{-}fault}(\mathsf{dev}, x)$$

In this example, only a known instance of a concept and an object that is related via an epistemic role to some other named object is chosen. In such simple cases, we will show that only named object mentioned in the action theory can be chosen. As a consequence we can remove the pick-operator by means of grounding and obtain decidability.

To make this work we impose several restrictions on the action theory and on the guards of the pick-operator we allow in the program.

Definition 7.41. Let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be an epistemic $\mathcal{ALCO}$-action theory with $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. We say that Σ_{s} is an action theory with only *local sensing* iff the following restrictions are satisfied:

- $\mathcal{T}$ is an $\mathcal{ALC}$-TBox, i.e. nominals are disallowed;
- for each $\alpha(\bar{t}) \in \mathsf{Act}$ the preconditions in $\mathsf{pre}(\alpha(\bar{t}))$ the effect conditions of the effects in $\mathsf{eff}(\alpha(\bar{t}))$ and $\mathsf{sense}(\alpha(\bar{t}))$ are restricted to be Boolean $\mathcal{ALCO}$-ABoxes, i.e. concept inclusions are disallowed.

▲

In the next section, we show that disallowing sensing of concept inclusions and using an initial KB with an $\mathcal{ALC}$-TBox leads to simple knowledge states, where certain epistemic concepts and epistemic roles have a finite extensions of only named objects.

7.4.1 Knowledge States after Local Sensing

In this section, we consider an epistemic $\mathcal{ALCO}$-action theory with only local sensing of the form

$$\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense}),$$

where $\boldsymbol{A}$ is a finite set of ground action terms. $\mathsf{Obj}(\Sigma_{\mathsf{s}})$ is the set of all object names mentioned in Σ_{s} and $\mathcal{C}$ an $\mathcal{ALCO}$-context that consists exactly of the axioms in $\mathcal{K}$, all preconditions, all effect conditions, all sensing properties, and the respective negated formulas. $\mathsf{Lit}(\Sigma_{\mathsf{s}})$ is the set of all relevant local effects.

Definition 7.41 implies that $\mathcal{C}$ consists only of Boolean $\mathcal{ALCO}$-ABoxes and $\mathcal{ALC}$-concept inclusions and their negations contained in the initial TBox.

We consider the following property of the knowledge states that evolve from the epistemic model $\mathcal{M}(\mathcal{K})$ by executing a sequence of ground actions.

Lemma 7.42. *Let* $\Sigma_{\mathsf{s}} = (\mathcal{K}, \boldsymbol{A}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ *be as described above and let*

$$\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}) = (\mathcal{M}(\mathcal{K}), \mathcal{F}, \boldsymbol{A}, \mathcal{E}, \mathsf{Pre}, \sim_{\mathsf{s}})$$

be the induced epistemic FO-DS. Furthermore, let $\sigma \in \boldsymbol{A}^*$, $P \in \mathsf{N_R}$ *a role name,* C *an* $\mathcal{ALC}$*-concept,* $\mathcal{I} \in \mathcal{M}(\mathcal{K})$ *and* $(\mathcal{J}, \mathcal{M})$ *the epistemic interpretation with*

$$(\mathcal{I}, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}})} (\mathcal{J}, \mathcal{M}).$$

It holds that

1. *if* $\mathcal{K} \not\models \top \sqsubseteq C$*, then* $(\mathbf{K}C)^{\mathcal{M}} \subseteq \mathsf{Obj}(\Sigma_{\mathsf{s}})$ *and*
2. $(\mathbf{K}P)^{\mathcal{M}} \subseteq \mathsf{Obj}(\Sigma_{\mathsf{s}}) \times \mathsf{Obj}(\Sigma_{\mathsf{s}})$.

Before we prove this lemma we consider the static case where the action sequence is empty. A proof for this case was given by Mehdi [Meh14] (page 76, Lemma 4) also using the swapping technique for unnamed elements from [Don+98]. The proof can be outlined as follows. Assume to the contrary that there exists an unnamed element $d \in \Delta \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that $d \in (\mathbf{K}C)^{\mathcal{M}(\mathcal{K})}$. Since $\mathcal{K} \not\models \top \sqsubseteq C$, there exists a model $\mathcal{J} \in \mathcal{M}(\mathcal{K})$ and an anonymous element $e \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ such that $e \notin C^{\mathcal{J}}$. Now consider a bijection $\iota : \mathsf{N_O} \to \mathsf{N_O}$ with $\iota(e) = d$ and $\iota(d) = e$ and $\iota(o) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$. The renaming ι just swaps the two unnamed elements d and e. The renaming of the model $\mathcal{J}$ using ι yields an interpretation, denoted by $\iota(\mathcal{J})$, that is isomorphic to $\mathcal{J}$ and is also a model of $\mathcal{K}$. Thus, from $\mathcal{J} \in \mathcal{M}(\mathcal{K})$ and $e \notin C^{\mathcal{J}}$ it follows that $\iota(\mathcal{J}) \in \mathcal{M}(\mathcal{K})$ and $\iota(e) \notin C^{\iota(\mathcal{J})}$. With $\iota(e) = d$ we get a contradiction to the assumption $d \in (\mathbf{K}C)^{\mathcal{M}(\mathcal{K})}$.

The proof we found for the dynamic case is a bit more involved. To reuse the idea described above we need to show that if initially $\mathcal{K} \not\models \top \sqsubseteq C$ holds, then we can still find in the resulting knowledge state $\mathcal{M}$ an interpretation $\mathcal{Y}$ such that there exists an unnamed domain element *not* contained in the extension of C under $\mathcal{Y}$. This domain element will serve as the “swap partner” for the unnamed known instance d of $\mathbf{K}D$ under $\mathcal{M}$ and will lead to the contradiction as in the static case.

In the following we introduce several auxiliary notions we need for the construction of such an interpretation $\mathcal{Y} \in \mathcal{M}$. First, an operation that merges two interpretations together in one is defined.

Definition 7.43. Let $\mathcal{I}_0$ and $\mathcal{I}_1$ be two interpretations satisfying the SNA. The *sum of* $\mathcal{I}_0$ *and* $\mathcal{I}_1$ is an interpretation, denoted by $\mathcal{I}_0 \oplus \mathcal{I}_1$, that is defined as follows:

$$\begin{aligned}
\Delta^{\mathcal{I}_0 \oplus \mathcal{I}_1} &:= \mathsf{N_O} \times \{0,1\};\\
A^{\mathcal{I}_0 \oplus \mathcal{I}_1} &:= \{\langle d,0\rangle \mid d \in A^{\mathcal{I}_0}\} \cup \{\langle d,1\rangle \mid d \in A^{\mathcal{I}_1}\} \text{ for all } A \in \mathsf{N_C}\\
P^{\mathcal{I}_0 \oplus \mathcal{I}_1} &:= \{(\langle d,0\rangle, \langle e,0\rangle) \mid (d,e) \in P^{\mathcal{I}_0}\} \cup \{(\langle d,1\rangle, \langle e,1\rangle) \mid (d,e) \in P^{\mathcal{I}_1}\} \text{ for all } P \in \mathsf{N_R}\\
o^{\mathcal{I}_0 \oplus \mathcal{I}_1} &:= \langle o,0\rangle \text{ for all } o \in \mathsf{N_O}.
\end{aligned}$$

▲

Note that the operation is non-commutative due to the interpretation of object names. We now consider a renaming of the sum $\mathcal{I}_0 \oplus \mathcal{I}_1$ that interprets the named part of the domain given by $\mathsf{Obj}(\Sigma_\mathsf{s})$ as in $\mathcal{I}_0$.

Lemma 7.44. *Let* $\mathcal{I}_0$, $\mathcal{I}_1$ *and* $\mathcal{I}_0 \oplus \mathcal{I}_1$ *be as above,* $\iota : \mathsf{N_O} \times \{0,1\} \to \mathsf{N_O}$ *a bijection such that* $\iota(\langle o,0\rangle) = o$ *for all* $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$, $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ *a set of local effects,* C *an* $\mathcal{ALCO}$*-concept,* ψ *a Boolean* $\mathcal{ALCO}$*-ABox, where all object names mentioned in* C *and* ψ *are contained in* $\mathsf{Obj}(\Sigma_\mathsf{s})$. *Furthermore, let* $\mathcal{J} := \iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$.

1. $d \in A^{\mathcal{I}_0^{\mathsf{L}}}$ *iff* $\iota(\langle d,0\rangle) \in A^{\mathcal{J}^{\mathsf{L}}}$ *for all* $d \in \mathsf{N_O}$ *and* $A \in \mathsf{N_C}$*;*

2. $(d,e) \in P^{\mathcal{I}_0^{\mathsf{L}}}$ *iff* $\big(\iota(\langle d,0\rangle), \iota(\langle e,0\rangle)\big) \in P^{\mathcal{J}^{\mathsf{L}}}$ *for all* $d,e \in \mathsf{N_O}$ *and* $P \in \mathsf{N_R}$*;*

3. $d \in C^{\mathcal{I}_0^{\mathsf{L}}}$ *iff* $\iota(\langle d,0\rangle) \in C^{\mathcal{J}^{\mathsf{L}}}$ *for all* $d \in \mathsf{N_O}$*;*

4. $\mathcal{I}_0^{\mathsf{L}} \models \psi$ *iff* $\mathcal{J}^{\mathsf{L}} \models \psi$*.*

Proof.

1. Let $d \in \mathsf{N_O}$ and $A \in \mathsf{N_O}$ and $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ a set of literals. First we show
$$d \in A^{\mathcal{I}_0} \text{ iff } \iota(\langle d,0\rangle) \in A^{\mathcal{J}}. \tag{7.10}$$
Using the definitions we get $d \in A^{\mathcal{I}_0}$ iff $\langle d,0\rangle \in A^{\mathcal{I}_0 \oplus \mathcal{I}_1}$ iff $\iota(\langle d,0\rangle) \in A^{\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)}$. Since L mentions only individuals from $\mathsf{Obj}(\Sigma_\mathsf{s})$ and by construction $\iota(\langle o,0\rangle) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$, it follows that
$$d \in \{o \mid \langle A,\{o\}\rangle^+ \in \mathsf{L}\} \text{ iff } \iota(\langle d,0\rangle) \in \{o \mid \langle A,\{o\}\rangle^+ \in \mathsf{L}\} \tag{7.11}$$
and
$$d \in \{o \mid \langle A,\{o\}\rangle^- \in \mathsf{L}\} \text{ iff } \iota(\langle d,0\rangle) \in \{o \mid \langle A,\{o\}\rangle^- \in \mathsf{L}\}. \tag{7.12}$$
By definition of interpretation updates and (7.10), (7.11) and (7.12) it follows that
$$d \in A^{\mathcal{I}_0{}^{\mathsf{L}}} \text{ iff } \iota(\langle d,0\rangle) \in A^{\mathcal{J}^{\mathsf{L}}}.$$

2. The proof is analogous to the proof of 1.

3. The proof is by induction on the structure of C.

   $C = A$: for some $A \in \mathsf{N_C}$, see 1.

   $C = \{o\}$: for some $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$. It holds that $d \in \{o\}^{\mathcal{I}_0{}^{\mathsf{L}}}$
   iff $d \in \{o^{\mathcal{I}_0{}^{\mathsf{L}}}\}$
   iff $d \in \{o\}$
   iff $d = o$
   iff $\iota(\langle d,0\rangle) = o$
   iff $\iota(\langle d,0\rangle) \in \{o\}$
   iff $\iota(\langle d,0\rangle) \in \{o^{\mathcal{J}^{\mathsf{L}}}\}$
   iff $\iota(\langle d,0\rangle) \in \{o\}^{\mathcal{J}^{\mathsf{L}}}$.

   $C = \neg D$: It holds that $d \in (\neg D)^{\mathcal{I}_0{}^{\mathsf{L}}}$
   iff $d \notin D^{\mathcal{I}_0{}^{\mathsf{L}}}$
   iff $\iota(\langle d,0\rangle) \notin D^{\mathcal{J}^{\mathsf{L}}}$ (by induction)
   iff $\iota(\langle d,0\rangle) \in (\neg D)^{\mathcal{J}^{\mathsf{L}}}$.

   $C = D_1 \sqcap D_2$: It holds that $d \in (D_1 \sqcap D_2)^{\mathcal{I}_0{}^{\mathsf{L}}}$
   iff $d \in D_1^{\mathcal{I}_0{}^{\mathsf{L}}}$ and $d \in D_2^{\mathcal{I}_0{}^{\mathsf{L}}}$
   iff $\iota(\langle d,0\rangle) \in D_1^{\mathcal{J}^{\mathsf{L}}}$ and $\iota(\langle d,0\rangle) \in D_2^{\mathcal{J}^{\mathsf{L}}}$ (by induction)
   iff $\iota(\langle d,0\rangle) \in (D_1 \sqcap D_2)^{\mathcal{J}^{\mathsf{L}}}$.

$C = \exists P.D$: It holds that $d \in (\exists P.D)^{\mathcal{I}_0{}^{\mathsf{L}}}$

iff there exists an $e \in \mathsf{N_O}$ s.t. $(d,e) \in P^{\mathcal{I}_0{}^{\mathsf{L}}}$ and $e \in D^{\mathcal{I}_0{}^{\mathsf{L}}}$

iff $\big(\iota(\langle d,0\rangle), \iota(\langle e,0\rangle)\big) \in P^{\mathcal{J}^{\mathsf{L}}}$ (by claim 2.) and $\iota(\langle e,0\rangle) \in D^{\mathcal{J}_0{}^{\mathsf{L}}}$ (by induction)

iff $\iota(\langle d,0\rangle) \in (\exists P.D)^{\mathcal{J}^{\mathsf{L}}}$.

The last equivalence holds because $(d,e) \in r^{\mathcal{J}^{\mathsf{L}}}$ and $\iota^{-}(d) = \langle d',0\rangle$ for some $d' \in \mathsf{N_O}$ implies $\iota^{-}(e) = \langle e',0\rangle$ for some $e' \in \mathsf{N_O}$.

4. It follows from claim 2 and 3 and the fact that $\iota(\langle o,0\rangle) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$.

□

Thus, the construction ensures that $\mathcal{I}_0$ and the renamed interpretation $\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$ with $\iota(\langle o,0\rangle) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$ have the same dynamic type w.r.t. $\mathsf{Lit}(\Sigma_\mathsf{s})$ and the context $\mathcal{C}$. However, the $\mathcal{I}_1$-part of $\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$ is not affected by updates as shown in the next lemma.

Lemma 7.45. *Let $\mathcal{I}_0$, $\mathcal{I}_1$ and $\mathcal{I}_0 \oplus \mathcal{I}_1$ be as above, $\iota : \mathsf{N_O} \times \{0,1\} \to \mathsf{N_O}$ a bijection such that $\iota(\langle o,0\rangle) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$, $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ a set of local effects, C an $\mathcal{ALC}$-concept and $\mathcal{J} := \iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$.*

1. *$d \in A^{\mathcal{I}_1}$ iff $\iota(\langle d,1\rangle) \in A^{\mathcal{J}^{\mathsf{L}}}$ for all $d \in \mathsf{N_O}$ and $A \in \mathsf{N_C}$;*

2. *$(d,e) \in P^{\mathcal{I}_1}$ iff $\big(\iota(\langle d,1\rangle), \iota(\langle e,1\rangle)\big) \in P^{\mathcal{J}^{\mathsf{L}}}$ for all $d,e \in \mathsf{N_O}$ and $P \in \mathsf{N_R}$;*

3. *$d \in C^{\mathcal{I}_1}$ iff $\iota(\langle d,1\rangle) \in C^{\mathcal{J}^{\mathsf{L}}}$ for all $d \in \mathsf{N_O}$.*

Proof.

1. Due to the construction of $\mathcal{J} := \iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$ it holds that $d \in A^{\mathcal{I}_1}$ iff $\langle d,1\rangle \in A^{\mathcal{I}_0 \oplus \mathcal{I}_1}$ iff $\iota(\langle d,1\rangle) \in A^{\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)}$. By definition of ι it holds that $\iota(\langle d,1\rangle) \notin \mathsf{Obj}(\Sigma_\mathsf{s})$ for all $d \in \mathsf{N_O}$. Since L contains only individuals from $\mathsf{Obj}(\Sigma_\mathsf{s})$, it follows that $\iota(\langle d,1\rangle) \in A^{\mathcal{J}}$ iff $\iota(\langle d,1\rangle) \in A^{\mathcal{J}^{\mathsf{L}}}$. Consequently, $d \in A^{\mathcal{I}_1}$ iff $\iota(\langle d,1\rangle) \in A^{\mathcal{J}^{\mathsf{L}}}$.

2. The proof is analogous to the proof of claim 1.

3. The proof is by induction on the structure of the $\mathcal{ALC}$-concept C using claim 1 and 2 and the property that for all $(d,e) \in \mathsf{N_O} \times \mathsf{N_O}$ and $P \in \mathsf{N_R}$ it holds that $(d,e) \in P^{\mathcal{J}^{\mathsf{L}}}$ and $\iota^{-}(d) = \langle d',1\rangle$ for some $d' \in \mathsf{N_O}$ implies $\iota^{-}(e) = \langle e',1\rangle$ for some $e' \in \mathsf{N_O}$.

□

We can now prove that $\mathcal{I}_0$ and $\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$ have the same dynamic type w.r.t. $\mathsf{Lit}(\Sigma_\mathsf{s})$ and $\mathcal{C}$.

Lemma 7.46. *Let $\mathcal{I}_0$, $\mathcal{I}_1$ and $\mathcal{I}_0 \oplus \mathcal{I}_1$ be as above, $\iota : \mathsf{N_O} \times \{0,1\} \to \mathsf{N_O}$ a bijection such that $\iota(\langle o,0\rangle) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$ and $\text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_0)$ and $\text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\iota(\mathcal{I}_0 \oplus \mathcal{I}_1))$ the dynamic type of $\mathcal{I}_0$ and $\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$, respectively, w.r.t. $\mathsf{Lit}(\Sigma_\mathsf{s})$ and $\mathcal{C}$. It holds that*

$$\text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_0) = \text{d-type}^{\mathsf{loc}}_{\mathcal{C}}(\iota(\mathcal{I}_0 \oplus \mathcal{I}_1)).$$

Proof. $\mathcal{C}$ consists of Boolean $\mathcal{ALCO}$-ABoxes and of Boolean $\mathcal{ALC}$-KBs of the form $C \sqsubseteq D$ and $\neg(C \sqsubseteq D)$ with $C \sqsubseteq D \in \mathcal{T}$, where $\mathcal{T}$ is the TBox of the initial KB $\mathcal{K}$. Let $\mathcal{J} := \iota(\mathcal{I}_0 \oplus \mathcal{I}_1)$. Due to Lemma 7.44.4 we have

$$(\psi, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_0) \text{ iff } (\psi, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$$

for all $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ and all Boolean $\mathcal{ALCO}$-ABoxes ψ in $\mathcal{C}$.

Let $(C \sqsubseteq D, \mathsf{L}) \in \mathcal{C} \times 2^{\mathsf{Lit}(\Sigma_\mathsf{s})}$ for some $C \sqsubseteq D \in \mathcal{T}$. First, assume $(C \sqsubseteq D, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_0)$. Let $d \in C^{\mathcal{J}^\mathsf{L}}$. First, assume $d = \iota(\langle e, 0\rangle)$ for some $e \in \mathsf{N_O}$. We have that $\iota(\langle e, 0\rangle) \in C^{\mathcal{J}^\mathsf{L}}$ implies $e \in C^{\mathcal{I}_0{}^\mathsf{L}}$ with Lemma 7.44. With $\mathcal{I}_0{}^\mathsf{L} \models C \sqsubseteq D$ it follows that $e \in D^{\mathcal{I}_0{}^\mathsf{L}}$. Lemma 7.44 implies $\iota(\langle e, 0\rangle) \in D^{\mathcal{J}^\mathsf{L}}$. Second, assume $d = \iota(\langle e, 1\rangle)$ for some $e \in \mathsf{N_O}$. With Lemma 7.45 we have that $\iota(\langle e, 1\rangle) \in C^{\mathcal{J}^\mathsf{L}}$ implies $e \in C^{\mathcal{I}_1}$. With $\mathcal{I}_1 \in \mathcal{M}(\mathcal{K})$ and $C \sqsubseteq D \in \mathcal{T}$ it follows that $\mathcal{I}_1 \models C \sqsubseteq D$. Hence, $e \in D^{\mathcal{I}_1}$ and due to Lemma 7.45 we also get $\iota(\langle e, 1\rangle) \in D^{\mathcal{J}^\mathsf{L}}$. It follows that $(C \sqsubseteq D, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$.

Next, assume $(C \sqsubseteq D, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{J})$. We show that $(C \sqsubseteq D, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_0)$. Let $e \in C^{\mathcal{I}_0{}^\mathsf{L}}$. Lemma 7.44 implies $\iota(\langle e, 0\rangle) \in C^{\mathcal{J}^\mathsf{L}}$. By assumption it follows that $\iota(\langle e, 0\rangle) \in D^{\mathcal{J}^\mathsf{L}}$. Again Lemma 7.44 yields $e \in D^{\mathcal{I}_0{}^\mathsf{L}}$. Hence, $C^{\mathcal{I}_0{}^\mathsf{L}} \subseteq D^{\mathcal{I}_0{}^\mathsf{L}}$. Consequently,

$$(C \sqsubseteq D, \mathsf{L}) \in \mathsf{d\text{-}type}^{\mathsf{loc}}_{\mathcal{C}}(\mathcal{I}_0).$$

□

Now we are ready to prove Lemma 7.42.

Proof of Lemma 7.42. Let $\mathcal{I}_0 \in \mathcal{M}(\mathcal{K})$, σ an action sequence, $(\mathcal{I}_\sigma, \mathcal{M})$ the epistemic interpretation with $(\mathcal{I}_0, \mathcal{M}(\mathcal{K})) \Longrightarrow^{\sigma}_{\mathfrak{D}_\mathsf{K}(\Sigma_\mathsf{s})} (\mathcal{I}_\sigma, \mathcal{M})$, C an $\mathcal{ALC}$-concept with $\mathcal{K} \not\models \top \sqsubseteq C$ and $P \in \mathsf{N_R}$.

1. We have to show that $(\mathbf{K}C)^{\mathcal{M}} \subseteq \mathsf{Obj}(\Sigma_\mathsf{s})$. Let $d \in (\mathbf{K}C)^{\mathcal{M}}$. Assume to the contrary that $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$. Since $\mathcal{K} \not\models \top \sqsubseteq C$, there exists $\mathcal{Y}_0 \in \mathcal{M}(\mathcal{K})$ and $e \in \mathsf{N_O}$ such that $e \notin C^{\mathcal{Y}_0}$. Consider the sum $\mathcal{I}_0 \oplus \mathcal{Y}_0$. Obviously, there exists a bijection $\iota : \mathsf{N_O} \times \{0,1\} \to \mathsf{N_O}$ such that $\iota(\langle o, 0\rangle) = o$ for all $o \in \mathsf{Obj}(\Sigma_\mathsf{s})$ and $\iota(\langle e, 1\rangle) = d$. Let $\mathcal{J} := \iota(\mathcal{I}_0 \oplus \mathcal{Y}_0)$ be the corresponding renamed interpretation. From Lemma 7.44 and 7.45 it follows that $\mathcal{J} \in \mathcal{M}(\mathcal{K})$. Lemma 7.46 implies that $\mathcal{I}_0$ and $\mathcal{J}$ have the same dynamic type. Due to Lemma 6.44 there exists a set of local effects $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ such that $\mathcal{J}^\mathsf{L} \in \mathcal{M}$ and $\mathcal{I}_\sigma = \mathcal{I}_0{}^\mathsf{L}$. Using Lemma 7.45.3 it holds that $e \in (\neg C)^{\mathcal{Y}_0}$ implies $\iota(\langle e, 1\rangle) \in (\neg C)^{\mathcal{J}^\mathsf{L}}$. Since $\iota(\langle e, 1\rangle) = d$ and $\mathcal{J}^\mathsf{L} \in \mathcal{M}$ it follows that $d \notin (\mathbf{K}C)^{\mathcal{M}}$. This is a contradiction to $d \in (\mathbf{K}C)^{\mathcal{M}}$. Thus, there is no unnamed element in $(\mathbf{K}C)^{\mathcal{M}}$.

2. Let $(d, e) \in (\mathbf{K}P)^{\mathcal{M}}$. We show that $(d, e) \in \mathsf{Obj}(\Sigma_\mathsf{s}) \times \mathsf{Obj}(\Sigma_\mathsf{s})$. First, assume to the contrary that $d \in \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$. Let $\mathcal{Y}_0 \in \mathcal{M}(\mathcal{K})$ be an arbitrary model and $\mathcal{I}_0 \oplus \mathcal{Y}_0$ the sum of $\mathcal{I}_0$ and $\mathcal{Y}_0$. Since $d \notin \mathsf{Obj}(\Sigma_\mathsf{s})$, there exists a bijection $\iota : \mathsf{N_O} \times \{0,1\} \to \mathsf{N_O}$ such that

$$\iota(\langle o, 0\rangle) = o \text{ for all } o \in \mathsf{Obj}(\Sigma_\mathsf{s}) \text{ and } \iota(\langle d, 1\rangle) = d \text{ and } \iota(\langle e, 0\rangle) = e.$$

Let $\mathcal{J} := \iota(\mathcal{I}_0 \oplus \mathcal{Y}_0)$ be the corresponding renamed interpretation. As in the first part of the lemma we can show that there exists a set of local effects $\mathsf{L} \subseteq \mathsf{Lit}(\Sigma_\mathsf{s})$ such that $\mathcal{J}^\mathsf{L} \in \mathcal{M}$. By definition of the sum it holds that $\big(\langle d, 1\rangle, \langle e, 0\rangle\big) \notin P^{\mathcal{I}_0 \oplus \mathcal{Y}_0}$ which implies also $\big(\iota(\langle d, 1\rangle), \iota(\langle e, 0\rangle)\big) \notin P^{\iota(\mathcal{I}_0 \oplus \mathcal{Y}_0)}$. The bijection ι is defined such that $\iota(\langle d, 1\rangle) \notin \mathsf{Obj}(\Sigma_\mathsf{s})$. Therefore,

$$\big(\iota(\langle d, 1\rangle), \iota(\langle e, 0\rangle)\big) \notin \Big\{(o, o') \,\Big|\, \big\langle P, \{(o, o')\}\big\rangle^+ \in \mathsf{L}\Big\} \subseteq \mathsf{Obj}(\Sigma_\mathsf{s}) \times \mathsf{Obj}(\Sigma_\mathsf{s}).$$

Consequently, we have $\big(\iota(\langle d, 1\rangle), \iota(\langle e, 0\rangle)\big) \notin P^{\mathcal{J}^\mathsf{L}}$ with $\mathcal{J} = \iota(\mathcal{I}_0 \oplus \mathcal{Y}_0)$. Hence, $\big(d, e\big) \notin P^{\mathcal{J}^\mathsf{L}}$. Since $\mathcal{J}^\mathsf{L} \in \mathcal{M}$, this is a contradiction to the assumption $(d, e) \in (\mathbf{K}P)^\mathcal{M}$ and $d \notin \mathsf{Obj}(\Sigma_\mathsf{s})$. Using symmetric arguments it can be shown that also the assumption $e \notin \mathsf{N_O} \setminus \mathsf{Obj}(\Sigma_\mathsf{s})$ leads to a contradiction.

□

7.4.2 A Pick-Operator with Epistemic Guards

To achieve decidability in presence of the pick-operator we restrict sensing to local sensing and guard the pick with ABox assertions of the form $(x \sqsubseteq \mathbf{K}C)$ or $((x, y) \sqsubseteq \mathbf{K}P)$, where C satisfies $\mathcal{K} \not\models \top \sqsubseteq C$ for the initial KB $\mathcal{K}$ with an $\mathcal{ALC}$-TBox. Lemma 7.42 implies that this leads to finitely many possible choices among the set of named objects. Consequently, a pick-operator guarded like this can be removed by grounding using the non-deterministic choice constructor “|”.

We formally define the class of $\mathcal{ALCOK}$-ConGolog with restricted guarded pick-operators.

Definition 7.47. Let $\mathcal{P} = (\mathfrak{D}_\mathbf{K}(\Sigma_\mathsf{s}), \delta)$ be an $\mathcal{ALCOK}$-ConGolog program. We say that $\mathcal{P}$ has only *restricted guarded pick expressions* iff the following conditions are satisfied:

- $\Sigma_\mathsf{s} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ offers only local sensing;
- all tests in δ are Boolean $\mathcal{ALCOK}$-ABoxes;
- for each guarded pick expression $(\mathsf{pick}(\bar{x}) \rightarrow \psi?; \rho)$ occurring in δ it holds that ψ is of the form $\varphi \wedge \psi'$, where all variables in $\bar{x}$ occur in φ and φ is restricted to be a conjunction of ABox assertions of the form

$$(t \sqsubseteq \mathbf{K}C) \text{ or } ((t, t') \sqsubseteq \mathbf{K}P)$$

with $\mathcal{K} \not\models \top \sqsubseteq C$.

▲

The tests are restricted to Boolean $\mathcal{ALCOK}$-ABoxes. This ensures that we can choose a relevant context $\mathcal{C}_\mathcal{P}$ for $\mathcal{P}$ that only consists of Boolean $\mathcal{ALCOK}$-ABoxes except for the concept inclusions in the initial $\mathcal{ALC}$-TBox and their negation.

Next, the grounding of $\mathcal{P}$ is defined.

Definition 7.48. Let $\mathcal{P} = (\mathfrak{D}_{\mathbf{K}}(\Sigma_{\mathsf{s}}), \delta)$ be an $\mathcal{ALCOK}$-ConGolog program with only restricted guarded pick expressions. Furthermore, let $\Sigma_{\mathsf{s}} = (\mathcal{K}, \mathsf{Act}, \mathsf{pre}, \mathsf{eff}, \mathsf{sense})$ be the underlying action theory and $\mathsf{Obj}(\Sigma_{\mathsf{s}})$ the set of all object names mentioned in Σ_{s}.

The *grounding of* $\mathcal{P}$ is an $\mathcal{ALCOK}$-ConGolog program over ground actions of the form

$$\widehat{\mathcal{P}} = (\mathfrak{D}_{\mathbf{K}}(\widehat{\Sigma_{\mathsf{s}}}), \widehat{\delta}),$$

where $\widehat{\Sigma_{\mathsf{s}}}$ is obtained from Σ_{s} by ground instantiating all action terms in Act with object names from $\mathsf{Obj}(\Sigma_{\mathsf{s}})$ in all possible ways. Let $(\mathsf{pick}(\bar{x}) \rightarrow \psi?; \rho)$ be a guarded pick expression occurring in δ and let

$$\widehat{\psi_1}; \widehat{\rho}_1, \ldots, \widehat{\psi_n}; \widehat{\rho}_n$$

be all program expressions obtained from $(\mathsf{pick}(\bar{x}) \rightarrow \psi?; \rho)$ by instantiating the free occurrences of the variables $\bar{x}$ in $\psi; \rho$ with object names from $\mathsf{Obj}(\Sigma_{\mathsf{s}})$. $\widehat{\delta}$ is then obtained from δ by exhaustively replacing each $(\mathsf{pick}(\bar{x}) \rightarrow \psi?; \rho)$ in δ by the program expression

$$\widehat{\psi_1}; \widehat{\rho}_1 \mid \cdots \mid \widehat{\psi_n}; \widehat{\rho}_n.$$

▲

Using Lemma 7.42 it is straightforward to show that $\mathcal{P}$ and the grounding $\widehat{\mathcal{P}}$ satisfy the same $\mathcal{ALCOK}$-CTL* properties over Boolean $\mathcal{ALCOK}$-ABoxes and concept inclusions contained in the initial KB of $\mathcal{P}$. With Theorem 7.40 we obtain decidability of the verification problem.

Theorem 7.49. *Verifying $\mathcal{ALCOK}$-CTL* properties over Boolean $\mathcal{ALCOK}$-ABoxes and $\mathcal{ALC}$-concept inclusions from the initial KB of a knowledge-based $\mathcal{ALCOK}$-ConGolog program with only restricted guarded pick expressions is decidable.*

7.5 Summary and Discussion

Main Results

In this chapter, we have shown decidability and complexity results for the verification problem of knowledge-based $\mathcal{ALCOK}$-ConGolog programs and $\mathcal{ALCOK}$-CTL* properties.

The verification problem for programs over unconditional ground actions is 2ExpTime-complete (Theorem 7.23). Thus, the complexity is the same as for (non-epistemic) $\mathcal{ALCO}$-ConGolog programs over ground actions with only local effects (Theorem 4.26).

An ExpTime-complete fragment of knowledge-based $\mathcal{ALCOK}$-ConGolog over unconditional ground actions can be obtained by disallowing sensing and by resorting to only subjective temporal properties. Intuitively, with this restriction we completely decouple the verification problem from the actual state of the environment. No sensing is involved and the outcome of the actions is immediately observable due to the unconditional effects. Adding sensing and/or temporal properties with objective Boolean KBs to the fragment leads again to 2ExpTime-hardness (Corollary 7.13 and 7.14).

Another ExpTime-complete fragment is obtained by disallowing world-changing effects and considering a program over purely sensing actions.

In case of conditional effects the abstraction technique is much more involved. We have used a technique that abstracts the knowledge state as a set of dynamic types. Decidability is

effects **sensing**	unconditional no	unconditional yes	no effects yes	conditional yes/no
subjective $\mathcal{ALCOK}$-CTL*	Exp	2Exp	Exp	$\leq$ 3Exp
objective $\mathcal{ALCOK}$-CTL*	2Exp	2Exp	Exp	$\leq$ 3Exp

Table 7.1: Complexity of verifying pick-free knowledge-based programs

shown with a 3ExpTime upper bound. It is left as an open problem whether this bound is tight.

By disallowing sensing of concept inclusions and imposing some other restrictions we were able to add a restricted variant of the guarded pick operator to our decidable fragment (Theorem 7.49).

Related Work

The complexity of verifying postconditions of terminating knowledge-based programs in a propositional setting has been investigated in [LZ12].

Golog-like programs over DL-ontologies have been studied in [Cal+07a]. However, a distinction between world-changing and knowledge-changing effects of actions is not made. The actions only operate on the knowledge base.

Decidable verification of temporal properties of online executions with sensing in the framework of the bounded Situation Calculus has been studied in [De +16a].

Future Work

One direction for future work is to further explore the computational complexity of verification of knowledge-based $\mathcal{ALCOK}$-ConGolog programs. There are various fragments for which the complexity is still open.

Another direction is to further push the decidability border towards a more expressive language. One of the main limitations of our fragment is, that actions can be only instantiated with named objects. It is not possible to verify an agent that is able to discover previously anonymous objects during its execution. Also extensions to probabilistic beliefs and noisy sensing [BHL99] are relevant in order to improve the applicability of our methods.

Chapter 8

Conclusions

In this chapter, we first summarize the main results and then provide some directions for future research.

8.1 Main Results

The goal of this thesis was to explore the boundary between decidable and undecidable fragments of the verification problem for programs in the Golog family of action programming languages. One of the sources of undecidability is the use of first-order logic as the base logic for defining the underlying domain theory and the tests in the program. To overcome this problem we have defined a general class of action formalisms based on description logics. DLs are expressive and well-suited for efficient reasoning in presence of the open-world assumption. Building on previous results on decidable reasoning in DL-based action languages we have extended the applicability of the reasoning methods to considerably more expressive languages than in previous works.

We have introduced an abstraction technique that we have used to obtain decidability and complexity results for the verification problem for Golog programs over DL-definable actions and specifications formulated in temporalized Description Logics. In the fragments that we have considered the transition system of the program has infinitely many states. We have used our abstraction technique to show that a finite propositional abstraction of the infinite transition system is effectively computable. This finite transition system can be given as an input to a model checking tool.

In Chapter 4, we have considered the verification problem for $\mathcal{L}$-ConGolog programs over $\mathcal{L}$-definable actions with only local effects, and CTL* properties over $\mathcal{L}$-axioms, where $\mathcal{L}$ is a DL between $\mathcal{ALC}$ and $\mathcal{ALCQIO}$. We have shown that the verification problem is 2ExpTime-complete if $\mathcal{L} \in \{\mathcal{ALCO}, \mathcal{ALCIO}, \mathcal{ALCQO}\}$, and co-N2ExpTime-complete if $\mathcal{L} = \mathcal{ALCQIO}$.

In Chapter 5, the decidability boundary for programs over non-local effect actions is explored. We have identified two expressive classes of $\mathcal{DL}$-ConGolog programs over ground actions for which verifying $\mathcal{DL}$-CTL* properties is decidable.

In Chapter 6, we have defined a decidable fragment of the epistemic Situation Calculus based on the prototypical basic DL $\mathcal{ALCO}$. In this language we can distinguish between world-changing and knowledge-changing effects of an action. We have shown that the epistemic projection problem for queries formulated in the epistemic DL $\mathcal{ALCOK}$ is ExpTime-complete. Thus, the problem is not harder than standard reasoning in $\mathcal{ALCO}$.

In Chapter 7, we have studied the verification problem for knowledge-based ConGolog programs based on $\mathcal{ALCOK}$ and the action language defined in Chapter 6. We have investigated how the complexity of the verification problem is affected by the interactions between sensing and physical effects of actions. We have identified 2ExpTime-complete and ExpTime-complete fragments for the verification of subjective and objective CTL* properties over $\mathcal{ALCOK}$-axioms. Moreover, a restricted version of the pick-operator was added to the decidable fragment. In the previously considered fragments this construct was always excluded.

8.2 Future Work

The exact complexity of the verification problem is still open for many variants of the considered fragments. For example, for the decidable classes with non-local effect actions in Chapter 5 the complexity is unknown. Furthermore, it is interesting to explore whether the results in Chapter 6 can be used for an implementation of a Golog interpreter based on description logics.

Moreover, it is desirable to further extend the expressiveness of the obtained languages in several directions:

- The expressiveness of the temporal specification language used in this thesis is quite restricted because temporal operators can be applied only to axioms and not to concepts or roles. One could consider also the verification of properties with temporal properties within the scope of quantifiers.
- One could extend the verification method towards decision-theoretic [Bou+00] extensions of Golog.
- The notion of sensing we have adopted is quite abstract. To obtain a more realistic model it is important to take into account that sensors are noisy. A logical formalization for quantitative sensor models is considered, for instance, in [BHL99].

Bibliography

[Ahm+14] Shqiponja Ahmetaj et al. “Managing Change in Graph-Structured Data Using Description Logics”. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI 2014)*. Ed. by Carla E. Brodley and Peter Stone. AAAI Press, 2014, pp. 966–973.

[Baa+05a] Franz Baader et al. “Integrating Description Logics and Action Formalisms: First Results”. In: *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI 2005)*. Ed. by Manuela M. Veloso and Subbarao Kambhampati. AAAI Press, 2005, pp. 572–577.

[Baa+05b] F. Baader et al. *Integrating Description Logics and Action Formalisms for Reasoning about Web Services*. LTCS-Report LTCS-05-02. Germany: Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, 2005. URL: `http://lat.inf.tu-dresden.de/research/reports.html.`.

[Baa+10] Franz Baader et al. *The Description Logic Handbook: Theory, Implementation and Applications*. 2nd. New York, NY, USA: Cambridge University Press, 2010.

[Bac01] Fahiem Bacchus. “The AIPS ’00 Planning Competition”. In: *AI Magazine* 22.3 (2001), pp. 47–56. URL: `http://www.aaai.org/ojs/index.php/aimagazine/article/view/1571`.

[BBL05] Franz Baader, Sebastian Brandt, and Carsten Lutz. “Pushing the EL Envelope”. In: *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*. Ed. by Leslie Pack Kaelbling and Alessandro Saffiotti. Professional Book Center, 2005, pp. 364–369. URL: `http://ijcai.org/Proceedings/05/Papers/0372.pdf`.

[Ber+11] Julien Bertrane et al. “Static analysis by abstract interpretation of embedded critical software”. In: *ACM SIGSOFT Software Engineering Notes* 36.1 (2011), pp. 1–8. URL: `http://doi.acm.org/10.1145/1921532.1921553`.

[BGL12] Franz Baader, Silvio Ghilardi, and Carsten Lutz. “LTL over description logic axioms”. In: *ACM Trans. Comput. Log.* 13.3 (2012), 21:1–21:32. URL: `http://doi.acm.org/10.1145/2287718.2287721`.

[BH14] Bernhard Beckert and Reiner Hähnle. “Reasoning and Verification: State of the Art and Current Trends”. In: *IEEE Intelligent Systems* 29.1 (2014), pp. 20–29. URL: `https://doi.org/10.1109/MIS.2014.3`.

[BHL99] Fahiem Bacchus, Joseph Y. Halpern, and Hector J. Levesque. “Reasoning about Noisy Sensors and Effectors in the Situation Calculus”. In: *Artificial Intelligence* 111.1–2 (1999), pp. 171–208.

[BK08] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[BLL10] Franz Baader, Marcel Lippmann, and Hongkai Liu. "Using Causal Relationships to Deal with the Ramification Problem in Action Formalisms Based on Description Logics". In: *Logic for Programming, Artificial Intelligence, and Reasoning - 17th International Conference, LPAR-17, Yogyakarta, Indonesia, October 10-15, 2010. Proceedings*. 2010, pp. 82–96. URL: http://dx.doi.org/10.1007/978-3-642-16242-8_7.

[BLM10] Franz Baader, Hongkai Liu, and Anees ul Mehdi. "Verifying Properties of Infinite Sequences of Description Logic Actions". In: *Proceedings of the Nineteenth European Conference on Artificial Intelligence (ECAI 2010)*. Ed. by Helder Coelho, Rudi Studer, and Michael Wooldridge. Vol. 215. Frontiers in Artificial Intelligence and Applications. IOS Press, 2010, pp. 53–58.

[Bor96] Alexander Borgida. "On the Relative Expressiveness of Description Logics and Predicate Logics". In: *Artif. Intell.* 82.1-2 (1996), pp. 353–367. URL: http://dx.doi.org/10.1016/0004-3702(96)00004-5.

[Bou+00] Craig Boutilier et al. "Decision-Theoretic, High-Level Agent Programming in the Situation Calculus". In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI 2000)*. Ed. by Henry Kautz and Bruce Porter. AAAI Press, 2000, pp. 355–362.

[Bur+90] Jerry R. Burch et al. "Symbolic Model Checking: 10^20 States and Beyond". In: *Proceedings of the Fifth Annual Symposium on Logic in Computer Science (LICS '90), Philadelphia, Pennsylvania, USA, June 4-7, 1990*. IEEE Computer Society, 1990, pp. 428–439. URL: https://doi.org/10.1109/LICS.1990.113767.

[Bur+99] Wolfram Burgard et al. "Experiences with an Interactive Museum Tour-Guide Robot". In: *Artificial Intelligence* 114.1–2 (1999), pp. 3–55.

[BZ13] Franz Baader and Benjamin Zarrieß. "Verification of Golog Programs over Description Logic Actions". In: *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*. Ed. by Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt. Vol. 8152. Lecture Notes in Computer Science. Springer, 2013, pp. 181–196. URL: https://doi.org/10.1007/978-3-642-40885-4_12.

[Cal+07a] Diego Calvanese et al. "Actions and Programs over Description Logic Ontologies". In: *Proceedings of the 2007 International Workshop on Description Logics (DL2007), Brixen-Bressanone, near Bozen-Bolzano, Italy, 8-10 June, 2007*. Ed. by Diego Calvanese et al. Vol. 250. CEUR Workshop Proceedings. CEUR-WS.org, 2007. URL: http://ceur-ws.org/Vol-250/paper_74.pdf.

[Cal+07b] Diego Calvanese et al. "EQL-Lite: Effective First-Order Query Processing in Description Logics". In: *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*. 2007, pp. 274–279. URL: http://ijcai.org/Proceedings/07/Papers/042.pdf.

[CC77] Patrick Cousot and Radhia Cousot. “Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints”. In: *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977*. Ed. by Robert M. Graham, Michael A. Harrison, and Ravi Sethi. ACM, 1977, pp. 238–252. URL: `http://doi.acm.org/10.1145/512950.512973`.

[CE81] Edmund M. Clarke and E. Allen Emerson. “Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic”. In: *Logics of Programs, Workshop, Yorktown Heights, New York, May 1981*. 1981, pp. 52–71. URL: `http://dx.doi.org/10.1007/BFb0025774`.

[CGP01] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model checking*. MIT Press, 2001. URL: `http://books.google.de/books?id=Nmc4wEaLXFEC`.

[CHL07] Jens Claßen, Yuxiao Hu, and Gerhard Lakemeyer. “A Situation-Calculus Semantics for an Expressive Fragment of PDDL”. In: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI 2007)*. Ed. by Robert C. Holte and Adele E. Howe. AAAI Press, 2007, pp. 956–961.

[CL08] Jens Claßen and Gerhard Lakemeyer. “A Logic for Non-Terminating Golog Programs”. In: *Proceedings of the Eleventh International Conference on the Principles of Knowledge Representation and Reasoning (KR 2008)*. Ed. by Gerhard Brewka and Jérôme Lang. AAAI Press, 2008, pp. 589–599.

[CL10] Jens Claßen and Gerhard Lakemeyer. “On the Verification of Very Expressive Temporal Properties of Non-terminating Golog Programs”. In: *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*. Ed. by Helder Coelho, Rudi Studer, and Michael Wooldridge. Vol. 215. Frontiers in Artificial Intelligence and Applications. IOS Press, 2010, pp. 887–892. URL: `https://doi.org/10.3233/978-1-60750-606-5-887`.

[De +09] Giuseppe De Giacomo et al. “IndiGolog: A High-Level Programming Language for Embedded Reasoning Agents”. In: *Multi-Agent Programming: Languages, Platforms and Applications*. Ed. by Rafael H. Bordini et al. Springer, 2009. Chap. 2, pp. 31–72.

[De +16a] Giuseppe De Giacomo et al. “Progression and Verification of Situation Calculus Agents with Bounded Beliefs”. In: *Studia Logica* 104.4 (2016), pp. 705–739. URL: `https://doi.org/10.1007/s11225-015-9626-z`.

[De +16b] Giuseppe De Giacomo et al. “Verifying ConGolog Programs on Bounded Situation Calculus Theories”. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.* Ed. by Dale Schuurmans and Michael P. Wellman. AAAI Press, 2016, pp. 950–956. URL: `http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12470`.

[De +97] Giuseppe De Giacomo et al. “Planning with Sensing for a Mobile Robot”. In: *Recent Advances in AI Planning, 4th European Conference on Planning, ECP’97, Toulouse, France, September 24-26, 1997, Proceedings*. Ed. by Sam Steel and Rachid Alami. Vol. 1348. Lecture Notes in Computer Science. Springer, 1997, pp. 156–168. URL: `https://doi.org/10.1007/3-540-63912-8_83`.

[DLL00] Giuseppe De Giacomo, Yves Lespérance, and Hector J. Levesque. “ConGolog, a concurrent programming language based on the situation calculus”. In: *Artificial Intelligence* 121.1–2 (2000), pp. 109–169.

[DLP16] Giuseppe De Giacomo, Yves Lespérance, and Fabio Patrizi. “Bounded situation calculus action theories”. In: *Artif. Intell.* 237 (2016), pp. 172–203. URL: `https://doi.org/10.1016/j.artint.2016.04.006`.

[Don+98] Francesco M. Donini et al. “An Epistemic Operator for Description Logics”. In: *Artif. Intell.* 100.1-2 (1998), pp. 225–274. URL: `http://dx.doi.org/10.1016/S0004-3702(98)00009-5`.

[DTR97] Giuseppe De Giacomo, Evgenia Ternovska, and Raymond Reiter. “Non-terminating Processes in the Situation Calculus”. In: *Working Notes of “Robots, Softbots, Immobots: Theories of Action, Planning and Control”, AAAI’97 Workshop*. 1997.

[FL03] Maria Fox and Derek Long. “PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains”. In: *Journal of Artificial Intelligence Research* 20 (2003), pp. 61–124.

[FN71] Richard Fikes and Nils J. Nilsson. “STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving”. In: *Artificial Intelligence* 2.3/4 (1971), pp. 189–208.

[GL93] Michael Gelfond and Vladimir Lifschitz. “Representing Action and Change by Logic Programs”. In: *Journal of Logic Programming* 17.2 (1993), pp. 301–321.

[GOR97] Erich Grädel, Martin Otto, and Eric Rosen. “Two-Variable Logic with Counting is Decidable”. In: *Proceedings, 12th Annual IEEE Symposium on Logic in Computer Science, Warsaw, Poland, June 29 - July 2, 1997*. IEEE Computer Society, 1997, pp. 306–317. URL: `https://doi.org/10.1109/LICS.1997.614957`.

[GS10] Yilan Gu and Mikhail Soutchanski. “A description logic based situation calculus”. In: *Annals of Mathematics and Artificial Intelligence* 58.1–2 (2010), pp. 3–83.

[Har+13] Babak Bagheri Hariri et al. “Description Logic Knowledge and Action Bases”. In: *J. Artif. Intell. Res.* 46 (2013), pp. 651–686. URL: `https://doi.org/10.1613/jair.3826`.

[Hoa69] C. A. R. Hoare. “An Axiomatic Basis for Computer Programming”. In: *Commun. ACM* 12.10 (1969), pp. 576–580. URL: `http://doi.acm.org/10.1145/363235.363259`.

[HTK00] David Harel, Jerzy Tiuryn, and Dexter Kozen. *Dynamic Logic*. Cambridge, MA, USA: MIT Press, 2000.

[KR14] Markus Krötzsch and Sebastian Rudolph. “Nominal Schemas in Description Logics: Complexities Clarified”. In: *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014*. 2014. URL: `http://www.aaai.org/ocs/index.php/KR/KR14/paper/view/8027`.

[Krö+11] Markus Krötzsch et al. “A better uncle for OWL: nominal schemas for integrating rules and ontologies”. In: *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*. 2011, pp. 645–654. URL: `http://doi.acm.org/10.1145/1963405.1963496`.

[KS86] Robert A. Kowalski and Marek J. Sergot. “A Logic-based Calculus of Events”. In: *New Generation Computing* 4.1 (1986), pp. 67–95.

[Lam77] Leslie Lamport. “Proving the Correctness of Multiprocess Programs”. In: *IEEE Trans. Software Eng.* 3.2 (1977), pp. 125–143. URL: `https://doi.org/10.1109/TSE.1977.229904`.

[Lev+97] Hector J. Levesque et al. “GOLOG: A Logic Programming Language for Dynamic Domains”. In: *Journal of Logic Programming* 31.1–3 (1997), pp. 59–83.

[Lev84] Hector J. Levesque. “Foundations of a Functional Approach to Knowledge Representation”. In: *Artificial Intelligence* 23.2 (1984), pp. 155–212.

[Lip14] Marcel Lippmann. “Temporalised description logics for monitoring partially observable events”. PhD thesis. Dresden University of Technology, 2014. URL: `http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-147977`.

[Liu+06] Hongkai Liu et al. “Reasoning About Actions Using Description Logics with General TBoxes”. In: *Logics in Artificial Intelligence, 10th European Conference, JELIA 2006, Liverpool, UK, September 13-15, 2006, Proceedings*. 2006, pp. 266–279. URL: `http://dx.doi.org/10.1007/11853886_23`.

[Liu02] Yongmei Liu. “A Hoare-Style Proof System for Robot Programs”. In: *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI 2002)*. Ed. by Rina Dechter, Michael Kearns, and Rich Sutton. AAAI Press, 2002, pp. 74–79.

[LL01] Hector J. Levesque and Gerhard Lakemeyer. *The Logic of Knowledge Bases*. MIT Press, 2001.

[LL04] Gerhard Lakemeyer and Hector J. Levesque. “Situations, Si! Situation Terms, No!” In: *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004)*. Ed. by Didier Dubois, Christopher A. Welty, and Mary-Anne Williams. AAAI Press, 2004, pp. 516–526.

[LL05] Gerhard Lakemeyer and Hector J. Levesque. “Semantics for a useful fragment of the situation calculus”. In: *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI 2005)*. Ed. by Leslie Pack Kaelbling and Alessandro Saffiotti. Professional Book Center, 2005, pp. 490–496.

[LL08] Hector J. Levesque and Gerhard Lakemeyer. “Cognitive Robotics”. In: *Foundations of Artificial Intelligence* 3 (2008), pp. 869–886.

[LL11] Gerhard Lakemeyer and Hector J. Levesque. “A semantic characterization of a useful fragment of the situation calculus with knowledge”. In: *Artif. Intell.* 175.1 (2011), pp. 142–164. URL: http://dx.doi.org/10.1016/j.artint.2010.04.005.

[LL14] Gerhard Lakemeyer and Hector J. Levesque. “Decidable Reasoning in a Fragment of the Epistemic Situation Calculus”. In: *Proceedings of the Fourtenth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2014)*. Ed. by Thomas Eiter, Chitta Baral, and Giuseppe De Giacomo. AAAI Press, 2014, pp. 468–477.

[LR94] Fangzhen Lin and Raymond Reiter. “State Constraints Revisited”. In: *Journal of Logic and Computation* 4.5 (1994), pp. 655–678.

[LR98] Hector J. Levesque and Raymond Reiter. “High-level robotic control: Beyond planning. A position paper”. In: *AAAI 1998 Spring Symposium: Integrating Robotics Research: Taking the Next Big Leap*. 1998.

[LZ12] Jérôme Lang and Bruno Zanuttini. “Knowledge-Based Programs as Plans - The Complexity of Plan Verification”. In: *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*. 2012, pp. 504–509. URL: https://doi.org/10.3233/978-1-61499-098-7-504.

[McM93] Kenneth L. McMillan. *Symbolic model checking*. Kluwer, 1993.

[Meh14] Anees ul Mehdi. “Epistemic Reasoning in OWL 2 DL”. PhD thesis. Karlsruhe Institute of Technology, 2014. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000039963.

[MH69] John McCarthy and Patrick Hayes. “Some philosophical problems from the standpoint of artificial intelligence”. In: *Machine Intelligence 4*. Ed. by B. Meltzer and D. Michie. New York: American Elsevier, 1969, pp. 463–502.

[Mil71] Robin Milner. “An Algebraic Definition of Simulation Between Programs”. In: *Proceedings of the 2nd International Joint Conference on Artificial Intelligence. London, UK, September 1-3, 1971.* Ed. by D. C. Cooper. William Kaufmann, 1971, pp. 481–489. URL: http://ijcai.org/Proceedings/71/Papers/044.pdf.

[Min67] Marvin L. Minsky. *Computation: Finite and Infinite Machines*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1967.

[MRG11] Anees Mehdi, Sebastian Rudolph, and Stephan Grimm. “Epistemic Querying of OWL Knowledge Bases”. In: *The Semantic Web: Research and Applications - 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29-June 2, 2011, Proceedings, Part I*. 2011, pp. 397–409. URL: http://dx.doi.org/10.1007/978-3-642-21034-1_27.

[MSH09] Boris Motik, Rob Shearer, and Ian Horrocks. “Hypertableau Reasoning for Description Logics”. In: *J. Artif. Intell. Res.* 36 (2009), pp. 165–228. URL: https://doi.org/10.1613/jair.2811.

[NPW02] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*. Vol. 2283. Lecture Notes in Computer Science. Springer, 2002. URL: `https://doi.org/10.1007/3-540-45949-9`.

[Owr+96] Sam Owre et al. “PVS: Combining Specification, Proof Checking, and Model Checking”. In: *Computer Aided Verification, 8th International Conference, CAV ’96, New Brunswick, NJ, USA, July 31 - August 3, 1996, Proceedings*. Ed. by Rajeev Alur and Thomas A. Henzinger. Vol. 1102. Lecture Notes in Computer Science. Springer, 1996, pp. 411–414. URL: `https://doi.org/10.1007/3-540-61474-5_91`.

[Ped94] Edwin P. D. Pednault. “ADL and the State-Transition Model of Action”. In: *Journal of Logic and Computation* 4.5 (1994), pp. 467–512.

[Pnu77] Amir Pnueli. “The Temporal Logic of Programs”. In: *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*. 1977, pp. 46–57. URL: `http://dx.doi.org/10.1109/SFCS.1977.32`.

[PR99] Fiora Pirri and Raymond Reiter. “Some Contributions to the Metatheory of the Situation Calculus”. In: *Journal of the ACM* 46.3 (1999), pp. 325–361.

[Rei01a] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.

[Rei01b] Raymond Reiter. “On knowledge-based programming with sensing in the situation calculus”. In: *ACM Transactions on Computational Logic* 2.4 (2001), pp. 433–457.

[Rei91] Raymond Reiter. “The Frame Problem in the Situation Calculus: A simple Solution (sometimes) and a Completeness Result for Goal Regression”. In: *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy* (1991), pp. 359–380.

[Roe14] Gabriele Roeger. “Planning techniques and the action language Golog”. PhD thesis. University of Freiburg, 2014. URL: `http://nbn-resolving.de/urn:nbn:de:bsz:25-opus-98599`.

[Ros07] Riccardo Rosati. “The Limits of Querying Ontologies”. In: *Database Theory - ICDT 2007, 11th International Conference, Barcelona, Spain, January 10-12, 2007, Proceedings*. 2007, pp. 164–178. URL: `http://dx.doi.org/10.1007/11965893_12`.

[Sar+04] Sebastian Sardiña et al. “On the Semantics of Deliberation in Indigolog - from Theory to Implementation”. In: *Ann. Math. Artif. Intell.* 41.2-4 (2004), pp. 259–299. URL: `https://doi.org/10.1023/B:AMAI.0000031197.13122.aa`.

[SD09] Sebastian Sardiña and Giuseppe De Giacomo. “Composition of ConGolog Programs”. In: *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*. Ed. by Craig Boutilier. 2009, pp. 904–910. URL: `http://ijcai.org/Proceedings/09/Papers/154.pdf`.

[SL03] Richard B. Scherl and Hector J. Levesque. “Knowledge, action, and the frame problem”. In: *Artificial Intelligence* 144.1–2 (2003), pp. 1–39.

[SLG14] Andreas Steigmiller, Thorsten Liebig, and Birte Glimm. “Konclude: System description”. In: *J. Web Sem.* 27 (2014), pp. 78–85. URL: `https://doi.org/10.1016/j.websem.2014.06.003`.

[SLL02] Steven Shapiro, Yves Lespérance, and Hector J. Levesque. “The cognitive agents specification language and verification environment for multiagent systems”. In: *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*. Ed. by Maria Gini et al. ACM Press, 2002, pp. 19–26.

[SS91] Manfred Schmidt-Schauß and Gert Smolka. “Attributive Concept Descriptions with Complements”. In: *Artif. Intell.* 48.1 (1991), pp. 1–26. URL: `https://doi.org/10.1016/0004-3702(91)90078-X`.

[Ter99] Eugenia Ternovskaia. “Automata Theory for Reasoning About Actions”. In: *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI 1999)*. Ed. by Thomas Dean. Morgan Kaufmann Publishers Inc., 1999, pp. 153–159.

[Thi11] Michael Thielscher. “A unifying action calculus”. In: *Artificial Intelligence* 175.1 (2011), pp. 120–41.

[Thi98] Michael Thielscher. “Introduction to the Fluent Calculus”. In: *Electron. Trans. Artif. Intell.* 2 (1998), pp. 179–192. URL: `http://www.ep.liu.se/ej/etai/1998/006/`.

[uR11] Anees ul Mehdi and Sebastian Rudolph. “Revisiting Semantics for Epistemic Extensions of Description Logics”. In: *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2011)*. Ed. by Wolfram Burgard and Dan Roth. AAAI Press, 2011.

[VLL08] Stavros Vassos, Gerhard Lakemeyer, and Hector J. Levesque. “First-Order Strong Progression for Local-Effect Basic Action Theories”. In: *Proceedings of the Eleventh International Conference on the Principles of Knowledge Representation and Reasoning (KR 2008)*. Ed. by Gerhard Brewka and Jérôme Lang. AAAI Press, 2008, pp. 662–672.

[VW86] Moshe Y. Vardi and Pierre Wolper. “Automata-Theoretic Techniques for Modal Logics of Programs”. In: *J. Comput. Syst. Sci.* 32.2 (1986), pp. 183–221. URL: `https://doi.org/10.1016/0022-0000(86)90026-7`.

[Yeh+12] Wael Yehia et al. “Experimental Results on Solving the Projection Problem in Action Formalisms Based on Description Logics”. In: *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012*. 2012. URL: `http://ceur-ws.org/Vol-846/paper_15.pdf`.

[ZC14] Benjamin Zarrieß and Jens Claßen. “Verifying CTL* Properties of GOLOG Programs over Local-Effect Actions”. In: *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*. Ed. by Torsten Schaub, Gerhard Friedrich, and Barry O’Sullivan. Vol. 263. Frontiers in Artificial Intelligence and Applications. IOS Press, 2014, pp. 939–944. URL: `https://doi.org/10.3233/978-1-61499-419-0-939`.

[ZC15a] Benjamin Zarrieß and Jens Claßen. “Decidable Verification of Knowledge-Based Programs over Description Logic Actions with Sensing”. In: *Proceedings of the Twenty-Eighth International Workshop on Description Logics (DL 2015)*. Ed. by Diego Calvanese and Boris Konev. Vol. 1350. CEUR Workshop Proceedings. CEUR-WS.org, 2015.

[ZC15b] Benjamin Zarrieß and Jens Claßen. “Verification of Knowledge-Based Programs over Description Logic Actions”. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*. Ed. by Qiang Yang and Michael Wooldridge. AAAI Press, 2015, pp. 3278–3284.

[ZC16] Benjamin Zarrieß and Jens Claßen. “Decidable Verification of Golog Programs over Non-Local Effect Actions”. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.* Ed. by Dale Schuurmans and Michael P. Wellman. AAAI Press, 2016, pp. 1109–1115. URL: `http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12283`.