5

Expressive Description Logics

Diego Calvanese

Giuseppe De Giacomo

Abstract

This chapter covers extensions of the basic description logics introduced in Chapter 2 by very expressive constructs that require advanced reasoning techniques. In particular, we study reasoning in description logics that include general inclusion axioms, inverse roles, number-restrictions, reflexive-transitive closure of roles, fixpoint constructs for recursive definitions, and relations of arbitrary arity. The chapter will also address reasoning w.r.t. knowledge bases including both a TBox and an ABox, and discuss more general ways to treat objects. Since the logics considered in the chapter lack the finite model property, finite model reasoning is of interest and will also be discussed. Finally, we mention several extensions to description logics that lead to undecidability, confirming that the expressive description logics considered in this chapter are close to the boundary between decidability and undecidability.

5.1 Introduction

Description logics have been introduced with the goal of providing a formal reconstruction of frame systems and semantic networks. Initially, the research has concentrated on subsumption of concept expressions. However, for certain applications, it turns out that it is necessary to represent knowledge by means of inclusion axioms without limitation on cycles in the TBox. Therefore, recently there has been a strong interest in the problem of reasoning over knowledge bases of a general form. See Chapters 2, 3, and 4 for more details.

When reasoning over general knowledge bases, it is not possible to gain tractability by limiting the expressive power of the description logic, because the power of arbitrary inclusion axioms in the TBox alone leads to high complexity in the inference mechanisms. Indeed, logical implication is ExpTime-hard even for the very simple language $\mathcal{AL}$ (see Chapter 3). This has lead to investigating very powerful languages for expressing concepts and roles, for which the property of interest is

no longer tractability of reasoning, but rather decidability. Such logics, called here *expressive description logics*, have the following characteristics:

(i) The language used for building concepts and roles comprises all classical concept forming constructs, plus several role forming constructs such as inverse roles, and reflexive-transitive closure.
(ii) No restriction is posed on the axioms in the TBox.

The goal of this chapter is to provide an overview on the results and techniques for reasoning in expressive description logics. The chapter is organized as follows. In Section 5.2, we outline the correspondence between expressive description logics and Propositional Dynamic Logics, which has given the basic tools to study reasoning in expressive description logics. In Section 5.3, we exploit automata-theoretic techniques developed for variants of Propositional Dynamic Logics to address reasoning in expressive description logics with functionality restrictions on roles. In Section 5.4 we illustrate the basic technique of *reification* for reasoning with expressive variants of number restrictions. In Section 5.5, we show how to reason with knowledge bases composed of a TBox and an ABox, and discuss extensions to deal with *names* (one-of construct). In Section 5.6, we introduce description logics with explicit fixpoint constructs, that are used to express in a natural way inductively and coinductively defined concepts. In Section 5.7, we study description logics that include relations of arbitrary arity, which overcome the limitations of traditional description logics of modeling only binary links between objects. This extension is particularly relevant for the application of description logics to databases. In Section 5.8, the problem of finite model reasoning in description logics is addressed. Indeed, for expressive description logics, reasoning w.r.t. finite models differs from reasoning w.r.t. unrestricted models, and requires specific methods. Finally, in Section 5.9, we discuss several extensions to description logics that lead in general to undecidability of the basic reasoning tasks. This shows that the expressive description logics considered in this chapter are close to the boundary to undecidability, and are carefully designed in order to retain decidability.

5.2 Correspondence between Description Logics and Propositional Dynamic Logics

In this section, we focus on expressive description logics that, besides the standard $\mathcal{ALC}$ constructs, include regular expression over roles and possibly inverse roles [Baader, 1991; Schild, 1991]. It turns out that such description logics correspond directly to Propositional Dynamic Logics, which are modal logics used to express properties of programs. We first introduce syntax and semantics of the description

logics we consider, then introduce Propositional Dynamic Logics, and finally discuss the correspondence between the two formalisms.

5.2.1 Description Logics

We consider the description logic $\mathcal{ALCI}_{reg}$, in which concepts and roles are formed according to the following syntax:

$$\begin{array}{rcl} C, C' & \longrightarrow & A \mid \neg C \mid C \sqcap C' \mid C \sqcup C' \mid \forall R.C \mid \exists R.C \\ R, R' & \longrightarrow & P \mid R \sqcup R' \mid R \circ R' \mid R^* \mid id(C) \mid R^- \end{array}$$

where A and P denote respectively atomic concepts and atomic roles, and C and R denote respectively arbitrary concepts and roles.

In addition to the usual concept forming constructs, $\mathcal{ALCI}_{reg}$ provides constructs to form regular expressions over roles. Such constructs include *role union*, *role composition*, *reflexive-transitive closure*, and *role identity*. Their meaning is straightforward, except for role identity $id(C)$ which, given a concept C, allows one to build a role which connects each instance of C to itself. As we shall see in the next section, there is a tight correspondence between these constructs and the operators on programs in Propositional Dynamic Logics. The presence in the language of the constructs for regular expressions is specified by the subscript "*reg*" in the name.

$\mathcal{ALCI}_{reg}$ includes also the *inverse role* construct, which allows one to denote the inverse of a given relation. One can, for example, state with $\exists \mathsf{child}^-.\mathsf{Doctor}$ that someone has a parent who is a doctor, by making use of the inverse of role child. It is worth noticing that, in a language without inverse of roles, in order to express such a constraint one must use two distinct roles (e.g., child and parent) that cannot be put in the proper relation to each other. We use the letter $\mathcal{I}$ in the name to specify the presence of inverse roles in a description logic; by dropping inverse roles from $\mathcal{ALC}_{reg}$, we obtain the description logic $\mathcal{ALC}_{reg}$.

From the semantic point of view, given an interpretation $\mathcal{I}$, concepts are interpreted as subsets of the domain $\Delta^{\mathcal{I}}$, and roles as binary relations over $\Delta^{\mathcal{I}}$, as follows[1]:

$$\begin{array}{rcl} A^{\mathcal{I}} & \subseteq & \Delta^{\mathcal{I}} \\ (\neg C)^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\ (C \sqcap C')^{\mathcal{I}} & = & C^{\mathcal{I}} \cap C'^{\mathcal{I}} \\ (C_1 \sqcup C_2)^{\mathcal{I}} & = & C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}} \\ (\forall R.C)^{\mathcal{I}} & = & \{o \in \Delta^{\mathcal{I}} \mid \forall o'.\, (o, o') \in R^{\mathcal{I}} \supset o' \in C^{\mathcal{I}}\} \end{array}$$

[1] We use $\mathcal{R}^*$ to denote the reflexive-transitive closure of the binary relation $\mathcal{R}$, and $\mathcal{R}_1 \circ \mathcal{R}_2$ to denote the chaining of the binary relations $\mathcal{R}_1$ and $\mathcal{R}_2$.

$$
\begin{array}{rcl}
(\exists R.C)^{\mathcal{I}} & = & \{o \in \Delta^{\mathcal{I}} \mid \exists o'.\, (o,o') \in R^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\} \\
P^{\mathcal{I}} & \subseteq & \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\
(R \sqcup R')^{\mathcal{I}} & = & R^{\mathcal{I}} \cup R'^{\mathcal{I}} \\
(R \circ R')^{\mathcal{I}} & = & R^{\mathcal{I}} \circ R'^{\mathcal{I}} \\
(R^*)^{\mathcal{I}} & = & (R^{\mathcal{I}})^* \\
id(C)^{\mathcal{I}} & = & \{(o,o) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid o \in C^{\mathcal{I}}\} \\
(R^-)^{\mathcal{I}} & = & \{(o,o') \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (o',o) \in R^{\mathcal{I}}\}
\end{array}
$$

We consider the most general form of TBoxes constituted by general inclusion axioms of the form $C \sqsubseteq C'$, without any restriction on cycles. We use $C \equiv C'$ as an abbreviation for the pair of axioms $C \sqsubseteq C'$ and $C' \sqsubseteq C$. We adopt the usual descriptive semantics for TBoxes (cf. Chapter 2).

Example 5.1 The following $\mathcal{ALCI}_{reg}$ TBox $\mathcal{T}_{file}$ models a file-system constituted by file-system elements (FSelem), each of which is either a Directory or a File. Each FSelem has a name, a Directory may have children while a File may not, and Root is a special directory which has no parent. The parent relationship is modeled through the inverse of role child.

$$
\begin{array}{rcl}
\mathsf{FSelem} & \sqsubseteq & \exists \mathsf{name}.\mathsf{String} \\
\mathsf{FSelem} & \equiv & \mathsf{Directory} \sqcup \mathsf{File} \\
\mathsf{Directory} & \sqsubseteq & \neg \mathsf{File} \\
\mathsf{Directory} & \sqsubseteq & \forall \mathsf{child}.\mathsf{FSelem} \\
\mathsf{File} & \sqsubseteq & \forall \mathsf{child}.\bot \\
\mathsf{Root} & \sqsubseteq & \mathsf{Directory} \\
\mathsf{Root} & \sqsubseteq & \forall \mathsf{child}^-.\bot
\end{array}
$$

The axioms in $\mathcal{T}_{file}$ imply that in a model every object connected by a chain of role child to an instance of Root is an instance of FSelem. Formally, $\mathcal{T}_{file} \models \exists(\mathsf{child}^-)^*.\mathsf{Root} \sqsubseteq \mathsf{FSelem}$. To verify that the implication holds, suppose that there exists a model in which an instance o of $\exists(\mathsf{child}^-)^*.\mathsf{Root}$ is not an instance of FSelem. Then, reasoning by induction on the length of the chain from the instance of Root to o, one can derive a contradiction. Observe that induction is required, and hence such reasoning is not first-order. ∎

In the following, when convenient, we assume, without loss of generality, that $\sqcup$ and $\forall R.C$ are expressed by means of $\neg$, $\sqcap$, and $\exists R.C$. We also assume that the inverse operator is applied to atomic roles only. This can be done again without

loss of generality, since the following equivalences hold: $(R_1; R_2)^- = R_1^- \circ R_2^-$, $(R - 1 \sqcup R_2)^- = R_1^- \sqcup R_2^-$, $(R^*)^- = (R^-)^*$, and $(id(C))^- = id(C)$.

5.2.2 Propositional Dynamic Logics

Propositional Dynamic Logics (PDLs) are modal logics specifically developed for reasoning about computer programs [Fischer and Ladner, 1979; Kozen and Tiuryn, 1990; Harel *et al.*, 2000]. In this section, we provide a brief overview of PDLs, and illustrate the correspondence between description logics and PDLs.

Syntactically, a PDL is constituted by expressions of two sorts: . *programs* and *formulae*. Programs and formulae are built by starting from *atomic programs* and *propositional letters*, and applying suitable operators. We denote propositional letters with A, arbitrary formulae with ϕ, atomic programs with P, and arbitrary programs with r, all possibly with subscripts. We focus on *converse*-PDL [Fischer and Ladner, 1979] which, as it turns out, corresponds to $\mathcal{ALCI}_{reg}$. The abstract syntax of *converse*-PDL is as follows:

$$\begin{array}{rcl} \phi, \phi' & \longrightarrow & \top \mid \bot \mid A \mid \phi \wedge \phi' \mid \phi \vee \phi' \mid \neg\phi \mid \langle r \rangle \phi \mid [r]\phi \\ r, r' & \longrightarrow & P \mid r \cup r' \mid r; r' \mid r^* \mid \phi? \mid r^- \end{array}$$

The basic Propositional Dynamic Logic PDL [Fischer and Ladner, 1979] is obtained from *converse*-PDL by dropping converse programs r^-.

The semantics of PDLs is based on the notion of (Kripke) structure, defined as a triple $\mathcal{M} = (\mathcal{S}, \{\mathcal{R}_P\}, \Pi)$, where $\mathcal{S}$ denotes a non-empty set of states, $\{\mathcal{R}_P\}$ is a family of binary relations over $\mathcal{S}$, each of which denotes the state transitions caused by an atomic program P, and Π is a mapping from $\mathcal{S}$ to propositional letters such that $\Pi(s)$ determines the letters that are true in state s. The basic semantical relation is "a formula ϕ holds at a state s of a structure $\mathcal{M}$", written $\mathcal{M}, s \models \phi$, and is defined by induction on the formation of ϕ:

$$\begin{array}{ll} \mathcal{M}, s \models A & \text{iff} \quad A \in \Pi(s) \\ \mathcal{M}, s \models \top & \text{always} \\ \mathcal{M}, s \models \bot & \text{never} \\ \mathcal{M}, s \models \phi \wedge \phi' & \text{iff} \quad \mathcal{M}, s \models \phi \text{ and } \mathcal{M}, s \models \phi' \\ \mathcal{M}, s \models \phi \vee \phi' & \text{iff} \quad \mathcal{M}, s \models \phi \text{ or } \mathcal{M}, s \models \phi' \\ \mathcal{M}, s \models \neg\phi & \text{iff} \quad \mathcal{M}, s \not\models \phi \\ \mathcal{M}, s \models \langle r \rangle \phi & \text{iff} \quad \text{there is } s' \text{ such that } (s, s') \in \mathcal{R}_r \text{ and } \mathcal{M}, s' \models \phi \\ \mathcal{M}, s \models [r]\phi & \text{iff} \quad \text{for all } s', (s, s') \in \mathcal{R}_r \text{ implies } \mathcal{M}, s' \models \phi \end{array}$$

where the family $\{\mathcal{R}_P\}$ is systematically extended so as to include, for every program

r, the corresponding relation $\mathcal{R}_r$ defined by induction on the formation of r:

$$\begin{array}{rcl}
\mathcal{R}_P & \subseteq & \mathcal{S} \times \mathcal{S} \\
\mathcal{R}_{r \cup r'} & = & \mathcal{R}_r \cup \mathcal{R}_{r'} \\
\mathcal{R}_{r;r'} & = & \mathcal{R}_r \circ \mathcal{R}_{r'} \\
\mathcal{R}_{r^*} & = & (\mathcal{R}_r)^* \\
\mathcal{R}_{\phi?} & = & \{(s,s) \in \mathcal{S} \times \mathcal{S} \mid \mathcal{M}, s \models \phi\} \\
\mathcal{R}_{r^-} & = & \{(s_1,s_2) \in \mathcal{S} \times \mathcal{S} \mid (s_2,s_1) \in \mathcal{R}_r\}.
\end{array}$$

If, for each atomic program P, the transition relation $\mathcal{R}_P$ is required to be a function that assigns to each state a unique successor state, then we are dealing with the *deterministic* variants of PDLs, namely DPDL and *converse*-DPDL [Ben-Ari *et al.*, 1982; Vardi and Wolper, 1986].

It is important to understand, given a formula ϕ, which are the formulae that play some role in establishing the truth-value of ϕ. In simpler modal logics, these formulae are simply all the subformulae of ϕ, but due to the presence of reflexive-transitive closure this is not the case for PDLs. Such a set of formula is given by the *Fischer-Ladner closure* of ϕ [Fischer and Ladner, 1979].

To be concrete we now illustrate the Fischer-Ladner closure for *converse*-PDL. However, the notion of Fischer-Ladner closure can be easily extended to other PDLs. Let us assume, without loss of generality, that $\vee$ and $[\cdot]$ are expressed by means of $\neg$, $\wedge$, and $\langle\cdot\rangle$. We also assume that the converse operator is applied to atomic programs only. This can again be done without loss of generality, since the following equivalences hold: $(r \cup r')^- = r^- \cup r'^-$, $(r;r')^- = r'^-;r^-$, $(r^*)^- = (r^-)^*$, and $(\phi?)^- = \phi?$.

The Fischer-Ladner closure of a *converse*-PDL formula ψ, denoted $CL(\psi)$, is the least set F such that $\psi \in F$ and such that:

$$\begin{array}{llll}
\text{if} & \phi \in F & \text{then} & \neg\phi \in F \quad (\text{if } \phi \text{ is not of the form } \neg\phi') \\
\text{if} & \neg\phi \in F & \text{then} & \phi \in F \\
\text{if} & \phi \wedge \phi' \in F & \text{then} & \phi, \phi' \in F \\
\text{if} & \langle r \rangle \phi \in F & \text{then} & \phi \in F \\
\text{if} & \langle r \cup r' \rangle \phi \in F & \text{then} & \langle r \rangle \phi, \langle r' \rangle \phi \in F \\
\text{if} & \langle r;r' \rangle \phi \in F & \text{then} & \langle r \rangle \langle r' \rangle \phi \in F \\
\text{if} & \langle r^* \rangle \phi \in F & \text{then} & \langle r \rangle \langle r^* \rangle \phi \in F \\
\text{if} & \langle \phi'? \rangle \phi \in F & \text{then} & \phi' \in F.
\end{array}$$

Note that $CL(\psi)$ includes all the subformulae of ψ, but also formulae of the form $\langle r \rangle \langle r^* \rangle \phi$ derived from $\langle r^* \rangle \phi$, which are in fact bigger than the formula they derive from. On the other hand, both the number and the size of the formulae in $CL(\psi)$ are linearly bounded by the size of ψ [Fischer and Ladner, 1979], exactly as the set of subformulae. Note also that, by definition, if $\phi \in CL(\psi)$, then $CL(\phi) \subseteq CL(\psi)$.

A structure $\mathcal{M} = (\mathcal{S}, \{\mathcal{R}_P\}, \Pi)$ is called a *model* of a formula ϕ if there exists a state $s \in \mathcal{S}$ such that $\mathcal{M}, s \models \phi$. A formula ϕ is *satisfiable* if there exists a model of ϕ, otherwise the formula is *unsatisfiable*. A formula ϕ is *valid* in structure $\mathcal{M}$ if for all $s \in \mathcal{S}$, $\mathcal{M}, s \models \phi$. We call *axioms* formulae that are used to select the interpretations of interest. Formally, a structure $\mathcal{M}$ is a model of an axiom ϕ, if ϕ is valid in $\mathcal{M}$. A structure $\mathcal{M}$ is a model of a finite set of axioms Γ if $\mathcal{M}$ is a model of all axioms in Γ. An axiom is satisfiable if it has a model and a finite set of axioms is satisfiable if it has a model. We say that a finite set Γ of axioms *logically implies* a formula ϕ, written $\Gamma \models \phi$, if ϕ is valid in every model of Γ.

It is easy to see that satisfiability of a formula ϕ as well as satisfiability of a finite set of axioms Γ can be reformulated by means of logical implication, as $\emptyset \not\models \neg\phi$ and $\Gamma \not\models \bot$ respectively.

Interestingly, logical implication can, in turn, be reformulated in terms of satisfiability, by making use of the following theorem (cf. [Kozen and Tiuryn, 1990]).

Theorem 5.2 (Internalization of axioms) *Let Γ be a finite set of* converse-PDL *axioms, and ϕ a* converse-PDL *formula. Then $\Gamma \models \phi$ if and only if the formula*

$$\neg\phi \wedge [(P_1 \cup \cdots \cup P_m \cup P_1^- \cup \cdots \cup P_m^-)^*]\Gamma'$$

is unsatisfiable, where $P_1, \ldots, P_m$ are all atomic programs occurring in $\Gamma \cup \{\phi\}$ and Γ' is the conjunction of all axioms in Γ.

Such a result exploits the power of program constructs (union, reflexive-transitive closure) and the *connected model property* (i.e., if a formula has a model, it has a model which is connected) of PDLs in order to represent axioms. The connected model property is typical of modal logics and it is enjoyed by all PDLs. As a consequence, a result analogous to Theorem 5.2 holds for virtually all PDLs.

Reasoning in PDLs has been thoroughly studied from the computational point of view, and the results for the PDLs considered here are summarized in the following theorem [Fischer and Ladner, 1979; Pratt, 1979; Ben-Ari *et al.*, 1982; Vardi and Wolper, 1986]:

Theorem 5.3 *Satisfiability in* PDL *is* ExpTime*-hard. Satisfiability in* PDL*, in* converse-PDL*, and in* converse-DPDL *can be decided in deterministic exponential time.*

5.2.3 The correspondence

The correspondence between description logics and PDLs was first published by Schild [1991].[1] In the work by Schild, it was shown that $\mathcal{ALCI}_{reg}$ can be considered a notational variant of *converse*-PDL. This observation allowed for exploiting the results on *converse*-PDL for instantly closing long standing issues regarding the decidability and complexity of both satisfiability and logical implication in $\mathcal{ALC}_{reg}$ and $\mathcal{ALCI}_{reg}$.[2] The paper was very influential for the research in expressive description logics in the following decade, since thanks to the correspondence between PDLs and description logics, first results but especially formal techniques and insights could be shared by the two communities. The correspondence between PDLs and description logics has been extensively used to study reasoning methods for expressive description logics. It has also lead to a number of interesting extensions of PDLs in terms of those constructs that are typical of description logics and have never been considered in PDLs. In particular, there is a tight relation between qualified number restrictions and graded modalities in modal logics [Van der Hoek, 1992; Van der Hoek and de Rijke, 1995; Fattorosi-Barnaba and De Caro, 1985; Fine, 1972].

The correspondence is based on the similarity between the interpretation structures of the two logics: at the extensional level, individuals (members of $\Delta^{\mathcal{I}}$) in description logics correspond to states in PDLs, whereas links between two individuals correspond to state transitions. At the intensional level, concepts correspond to propositions, and roles correspond to programs. Formally, the correspondence is realized through a one-to-one and onto mapping τ from $\mathcal{ALCI}_{reg}$ concepts to *converse*-PDL formulae, and from $\mathcal{ALCI}_{reg}$ roles to *converse*-PDL programs. The mapping τ is defined inductively as follows:

$$
\begin{array}{rclcrcl}
\tau(A) & = & A & \qquad & \tau(P) & = & P \\
\tau(\neg C) & = & \neg\tau(C) & & \tau(R^-) & = & \tau(R)^- \\
\tau(C \sqcap C') & = & \tau(C) \wedge \tau(C') & & \tau(R \sqcup R') & = & \tau(R) \cup \tau(R') \\
\tau(C \sqcup C') & = & \tau(C) \vee \tau(C') & & \tau(R \circ R') & = & \tau(R); \tau(R') \\
\tau(\forall R.C) & = & [\tau(R)]\tau(C) & & \tau(R^*) & = & \tau(R)^* \\
\tau(\exists R.C) & = & \langle\tau(R)\rangle\tau(C) & & \tau(id(C)) & = & \tau(C)?
\end{array}
$$

Axioms in description logics' TBoxes correspond in the obvious way to axioms in PDLs. Moreover all forms of reasoning (satisfiability, logical implication, etc.) have their natural counterpart.

One of the most important contributions of the correspondence is obtained by

[1] In fact, the correspondence was first noticed by Levesque and Rosenschein at the beginning of the '80s, but never published. In those days Levesque just used it in seminars to show intractability of certain description logics.

[2] In fact, the decidability of $\mathcal{ALC}_{reg}$ without the $id(C)$ construct was independently established by Baader [1991].

rephrasing Theorem 5.2 in terms of description logics. It says that every TBox can be "internalized" into a single concept, i.e., it is possible to build a concept that expresses all the axioms of the TBox. In doing so we rely on the ability to build a "universal" role, i.e., a role linking all individuals in a (connected) model. Indeed, a universal role can be expressed by using regular expressions over roles, and in particular the union of roles and the reflexive-transitive closure. The possibility of internalizing the TBox when dealing with expressive description logics tells us that for such description logics reasoning with TBoxes, i.e., logical implication, is no harder that reasoning with a single concept.

Theorem 5.4 *Concept satisfiability and logical implication in* $\mathcal{ALC}_{reg}$ *are* ExpTime*-hard. Concept satisfiability and logical implication in* $\mathcal{ALC}_{reg}$ *and* $\mathcal{ALCI}_{reg}$ *can be decided in deterministic exponential time.*

Observe that for description logics that do not allow for expressing a universal role, there is a sharp difference between reasoning techniques used in the presence of TBoxes, and techniques used to reason on concept expressions. The profound difference is reflected by the computational properties of the associated decision problems. For example, the logic $\mathcal{AL}$ admits simple structural algorithms for deciding reasoning tasks not involving axioms, and these algorithms are sound and complete and work in polynomial time. However, if general inclusion axioms are considered, then reasoning becomes ExpTime-complete (cf. Chapter 3), and the decision procedures that have been developed include suitable termination strategies [Buchheit *et al.*, 1993a]. Similarly, for the more expressive logic $\mathcal{ALC}$, reasoning tasks not involving a TBox are PSpace-complete [Schmidt-Schauß and Smolka, 1991], while those that do involve it are ExpTime-complete.

5.3 Functional restrictions

We have seen that the logics $\mathcal{ALC}_{reg}$ and $\mathcal{ALCI}_{reg}$ correspond to standard PDL and *converse*-PDL respectively, which are both well studied. In this section we show how the correspondence can be used to deal also with constructs that are typical of description logics, namely functional restrictions, by exploiting techniques developed for reasoning in PDLs. In particular, we will adopt automata-based techniques, which have been very successful in studying reasoning for expressive variants of PDL and characterizing their complexity.

Functional restrictions are the simplest form of number restrictions considered in description logics, and allow for specifying local functionality of roles, i.e., that instances of certain concepts have unique role-fillers for a given role. By adding functional restrictions on atomic roles and their inverse to $\mathcal{ALCI}_{reg}$, we obtain the description logic $\mathcal{ALCFI}_{reg}$. The PDL corresponding to $\mathcal{ALCFI}_{reg}$ is a PDL

that extends *converse*-DPDL [Vardi and Wolper, 1986] with determinism of both atomic programs and their inverse, and such that determinism is no longer a global property, but one that can be imposed locally.

Formally, $\mathcal{ALCFI}_{reg}$ is obtained from $\mathcal{ALCI}_{reg}$ by adding *functional restrictions* of the form $\leqslant 1\, Q$, where Q is a *basic role*, i.e., either an atomic role or the inverse of an atomic role. Such a functional restriction is interpreted as follows:

$$(\leqslant 1\, Q)^{\mathcal{I}} \quad = \quad \{o \in \Delta^{\mathcal{I}} \mid |\{o' \in \Delta^{\mathcal{I}} \mid (o, o') \in Q^{\mathcal{I}}\}| \leq 1\}$$

We show that reasoning in $\mathcal{ALCFI}_{reg}$ is in ExpTime, and, since reasoning in $\mathcal{ALC}_{reg}$ is already ExpTime-hard, is in fact ExpTime-complete. Without loss of generality we concentrate on concept satisfiability. We exploit the fact that $\mathcal{ALCFI}_{reg}$ has the *tree model property*, which states that if a $\mathcal{ALCFI}_{reg}$ concept C is satisfiable then it is satisfied in an interpretation which has the structure of a (possibly infinite) tree with bounded branching degree (see later). This allows us to make use of techniques based on automata on infinite trees. In particular, we make use of *two-way alternating automata on infinite trees* (2ATAs) introduced by Vardi [1998]. 2ATAs were used by Vardi [1998] to derive a decision procedure for modal μ-calculus with backward modalities. We first introduce 2ATAs and then show how they can be used to reason in $\mathcal{ALCFI}_{reg}$.

5.3.1 Automata on infinite trees

Infinite trees are represented as prefix closed (infinite) sets of words over $\mathbb{N}$ (the set of positive natural numbers). Formally, an *infinite tree* is a set of words $T \subseteq \mathbb{N}^*$, such that if $x{\cdot}c \in T$, where $x \in \mathbb{N}^*$ and $c \in \mathbb{N}$, then also $x \in T$. The elements of T are called *nodes*, the empty word ε is the *root* of T, and for every $x \in T$, the nodes $x{\cdot}c$, with $c \in \mathbb{N}$, are the *successors* of x. By convention we take $x{\cdot}0 = x$, and $x{\cdot}i{\cdot}{-}1 = x$. The *branching degree* $d(x)$ of a node x denotes the number of successors of x. If the branching degree of all nodes of a tree is bounded by k, we say that the tree has branching degree k. An *infinite path* P of T is a prefix-closed set $P \subseteq T$ such that for every $i \geq 0$ there exists a unique node $x \in P$ with $|x| = i$. A *labeled tree* over an alphabet Σ is a pair (T, V), where T is a tree and $V : T \to \Sigma$ maps each node of T to an element of Σ.

Alternating automata on infinite trees are a generalization of nondeterministic automata on infinite trees, introduced by Muller and Schupp [1987]. They allow for an elegant reduction of decision problems for temporal and program logics [Emerson and Jutla, 1991; Bernholtz *et al.*, 1994]. Let $\mathcal{B}(I)$ be the set of positive Boolean formulae over I, built inductively by applying $\wedge$ and $\vee$ starting from **true**, **false**, and elements of I. For a set $J \subseteq I$ and a formula $\varphi \in \mathcal{B}(I)$, we say that J *satisfies* φ if and only if, assigning **true** to the elements in J and **false** to those in $I \setminus J$, makes

φ true. For a positive integer k, let $[k] = \{-1, 0, 1, \ldots, k\}$. A *two-way alternating automaton* over infinite trees with branching degree k, is a tuple $\mathbf{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$, where Σ is the input alphabet, Q is a finite set of states, $\delta : Q \times \Sigma \to \mathcal{B}([k] \times Q)$ is the transition function, $q_0 \in Q$ is the initial state, and F specifies the acceptance condition.

The transition function maps a state $q \in Q$ and an input letter $\sigma \in \Sigma$ to a positive Boolean formula over $[k] \times Q$. Intuitively, if $\delta(q, \sigma) = \varphi$, then each pair (c, q') appearing in φ corresponds to a new copy of the automaton going to the direction suggested by c and starting in state q'. For example, if $k = 2$ and $\delta(q_1, \sigma) = (1, q_2) \wedge (1, q_3) \vee (-1, q_1) \wedge (0, q_3)$, when the automaton is in the state q_1 and is reading the node x labeled by the letter σ, it proceeds either by sending off two copies, in the states q_2 and q_3 respectively, to the first successor of x (i.e., $x{\cdot}1$), or by sending off one copy in the state q_1 to the predecessor of x (i.e., $x{\cdot}{-1}$) and one copy in the state q_3 to x itself (i.e., $x{\cdot}0$).

A run of a 2ATA $\mathbf{A}$ over a labeled tree (T, V) is a labeled tree (T_r, r) in which every node is labeled by an element of $T \times Q$. A node in T_r labeled by (x, q) describes a copy of $\mathbf{A}$ that is in the state q and reads the node x of T. The labels of adjacent nodes have to satisfy the transition function of $\mathbf{A}$. Formally, a run (T_r, r) is a $T \times Q$-labeled tree satisfying:

(i) $\varepsilon \in T_r$ and $r(\varepsilon) = (\varepsilon, q_0)$.

(ii) Let $y \in T_r$, with $r(y) = (x, q)$ and $\delta(q, V(x)) = \varphi$. Then there is a (possibly empty) set $S = \{(c_1, q_1), \ldots, (c_n, q_n)\} \subseteq [k] \times Q$ such that:

- S satisfies φ and
- for all $1 \leq i \leq n$, we have that $y{\cdot}i \in T_r$, $x{\cdot}c_i$ is defined, and $r(y{\cdot}i) = (x{\cdot}c_i, q_i)$.

A run (T_r, r) is *accepting* if all its infinite paths satisfy the acceptance condition[1]. Given an infinite path $P \subseteq T_r$, let $inf(P) \subseteq Q$ be the set of states that appear infinitely often in P (as second components of node labels). We consider here *Büchi* acceptance conditions. A Büchi condition over a state set Q is a subset F of Q, and an infinite path P satisfies F if $inf(P) \cap F \neq \emptyset$.

The non-emptiness problem for 2ATAs consists in determining, for a given a, whether the set of trees it accepts is nonempty. The results by Vardi [1998] provide the following complexity characterization of non-emptiness of 2ATAs.

Theorem 5.5 ([Vardi, 1998]) *Given a 2ATA* $\mathbf{A}$ *with* n *states and an input alphabet with* m *elements, deciding non-emptiness of* $\mathbf{A}$ *can be done in time exponential in* n *and polynomial in* m*.*

[1] No condition is imposed on the finite paths of the run.

5.3.2 Reasoning in $\mathcal{ALCFI}_{reg}$

The (Fischer-Ladner) *closure* for $\mathcal{ALCFI}_{reg}$ extends immediately the analogous notion for *converse*-PDL (see Section 5.2.2), treating functional restrictions as atomic concepts. In particular, the closure $CL(C_0)$ of an $\mathcal{ALCFI}_{reg}$ concept C_0 is defined as the smallest set of concepts such that $C_0 \in CL(C_0)$ and such that (assuming $\sqcup$ and $\forall$ to be expressed by means of $\sqcap$ and $\exists$, and the inverse operator applied only to atomic roles)[2]:

if	$C \in CL(C_0)$	then	$\neg C \in CL(C_0)$ (if C is not of the form $\neg C'$)
if	$\neg C \in CL(C_0)$	then	$C \in CL(C_0)$
if	$C \sqcap C' \in CL(C_0)$	then	$C, C' \in CL(C_0)$
if	$\exists R.C \in CL(C_0)$	then	$C \in CL(C_0)$
if	$\exists(R \sqcup R').C \in CL(C_0)$	then	$\exists R.C, \exists R'.C \in CL(C_0)$
if	$\exists(R \circ R').C \in CL(C_0)$	then	$\exists R.\exists R'.C \in CL(C_0)$
if	$\exists R^*.C \in CL(C_0)$	then	$\exists R.\exists R^*.C \in CL(C_0)$
if	$\exists id(C).C' \in CL(C_0)$	then	$C \in CL(C_0)$

The cardinality of $CL(C_0)$ is linear in the length of C_0.

It can be shown, following the lines of the proof in [Vardi and Wolper, 1986] for *converse*-DPDL, that $\mathcal{ALCFI}_{reg}$ enjoys the *tree model property*, i.e., every satisfiable concept has a model that has the structure of a (possibly infinite) tree with branching degree linearly bounded by the size of the concept. More precisely, we have the following result.

Theorem 5.6 *Every satisfiable $\mathcal{ALCFI}_{reg}$ concept C_0 has a tree model with branching degree k_{C_0} equal to twice the number of elements of $CL(C_0)$.*

This property allows us to check satisfiability of an $\mathcal{ALCFI}_{reg}$ concept C_0 by building a 2ATA that accepts the (labeled) trees that correspond to tree models of C_0. Let $\mathcal{A}$ be the set of atomic concepts appearing in C_0, and $\mathcal{B} = \{Q_1, \ldots, Q_n\}$ the set of atomic roles appearing in C_0 and their inverses. We construct from C_0 a 2ATA $\mathbf{A}_{C_0}$ that checks that C_0 is satisfied at the root of the input tree. We represent in each node of the tree the information about which atomic concepts are true in the node, and about the basic role that connects the predecessor of the node to the node itself (except for the root). More precisely, we label each node with a pair $\sigma = (\alpha, q)$, where α is the set of atomic concepts that are true in the node, and $q = Q$ if the node is reached from its predecessor through the basic role Q. That is, if Q stands for an atomic role P, then the node is reached from its predecessor through P, and if Q stands for P^-, then the predecessor is reached from the node

[2] We remind that C and C' stand for arbitrary concepts, and R and R' stand for arbitrary roles.

through P. In the root, $q = P_{dum}$, where P_{dum} is a new symbol representing a dummy role.

Given an $\mathcal{ALCFI}_{reg}$ concept C_0, we construct an automaton $\mathbf{A}_{C_0}$ that accepts trees that correspond to tree models of C_0. For technical reasons, it is convenient to consider concepts in *negation normal form* (i.e., negations are pushed inside as much as possible). It is easy to check that the transformation of a concept into negation normal form can be performed in linear time in the size of the concept. Below, we denote by $nnf(C)$ the negation normal form of C, and with $CL_{nnf}(C_0)$ the set $\{nnf(C) \mid C \in CL(C_0)\}$. The automaton $\mathbf{A}_{C_0} = (\Sigma, S, \delta, s_{ini}, F)$ is defined as follows.

- The alphabet is $\Sigma = 2^{\mathcal{A}} \times (\mathcal{B} \cup \{P_{dum}\})$, i.e., the set of pairs whose first component is a set of atomic concepts, and whose second component is a basic role or the dummy role P_{dum}. This corresponds to labeling each node of the tree with a truth assignment to the atomic concepts, and with the role used to reach the node from its predecessor.
- The set of states is $S = \{s_{ini}\} \cup CL_{nnf}(C_0) \cup \{Q, \neg Q \mid Q \in \mathcal{B}\}$, where s_{ini} is the initial state, $CL_{nnf}(C_0)$ is the set of concepts (in negation normal form) in the closure of C_0, and $\{Q, \neg Q \mid Q \in \mathcal{B}\}$ are states used to check whether a basic role labels a node. Intuitively, when the automaton in a state $C \in CL_{nnf}(C_0)$ visits a node x of the tree, this means that the automaton has to check that C holds in x.
- The transition function δ is defined as follows.

  1. For each $\alpha \in 2^{\mathcal{A}}$, there is a transition from the initial state

  $$\delta(s_{ini}, (\alpha, P_{dum})) \;=\; (0, nnf(C_0))$$

  Such a transition checks that the root of the tree is labeled with the dummy role P_{dum}, and moves to the state that verifies C_0 in the root itself.

  2. For each $(\alpha, q) \in \Sigma$ and each atomic concept $A \in \mathcal{A}$, there are transitions

  $$\begin{aligned} \delta(A, (\alpha, q)) &= \begin{cases} \textbf{true}, & \text{if } A \in \alpha \\ \textbf{false}, & \text{if } A \notin \alpha \end{cases} \\ \delta(\neg A, (\alpha, q)) &= \begin{cases} \textbf{true}, & \text{if } A \notin \alpha \\ \textbf{false}, & \text{if } A \in \alpha \end{cases} \end{aligned}$$

  Such transitions check the truth value of atomic concepts and their negations in the current node of the tree.

3. For each $(\alpha, q) \in \Sigma$ and each basic role $Q \in \mathcal{B}$, there are transitions

$$\begin{aligned} \delta(Q, (\alpha, q)) &= \begin{cases} \textbf{true}, & \text{if } q = Q \\ \textbf{false}, & \text{if } q \neq Q \end{cases} \\ \delta(\neg Q, (\alpha, q)) &= \begin{cases} \textbf{true}, & \text{if } q \neq Q \\ \textbf{false}, & \text{if } q = Q \end{cases} \end{aligned}$$

   Such transitions check through which role the current node is reached.

4. For the concepts in $CL_{nnf}(C_0)$ and each $\sigma \in \Sigma$, there are transitions

$$\begin{aligned} \delta(C \sqcap C', \sigma) &= (0, C) \ \wedge \ (0, C) \\ \delta(C \sqcup C', \sigma) &= (0, C) \ \vee \ (0, C') \\ \delta(\forall Q.C, \sigma) &= ((0, \neg Q^-) \vee (-1, C)) \ \wedge \ \textstyle\bigwedge_{1 \leq i \leq k_{C_0}} ((i, \neg Q) \vee (i, C)) \\ \delta(\forall (R \sqcup R').C, \sigma) &= (0, \forall R.C) \ \wedge \ (0, \forall R'.C) \\ \delta(\forall (R \circ R').C, \sigma) &= (0, \forall R.\forall R'.C) \\ \delta(\forall R^*.C, \sigma) &= (0, C) \ \wedge \ (0, \forall R.\forall R^*.C) \\ \delta(\forall id(C).C', \sigma) &= (0, nnf(\neg C)) \ \vee \ (0, C') \\ \delta(\exists Q.C, \sigma) &= ((0, Q^-) \wedge (-1, C)) \ \vee \ \textstyle\bigvee_{1 \leq i \leq k_{C_0}} ((i, Q) \wedge (i, C)) \\ \delta(\exists (R \sqcup R').C, \sigma) &= (0, \exists R.C) \ \vee \ (0, \exists R'.C) \\ \delta(\exists (R \circ R').C, \sigma) &= (0, \exists R.\exists R'.C) \\ \delta(\exists R^*.C, \sigma) &= (0, C) \ \vee \ (0, \exists R.\exists R^*.C) \\ \delta(\exists id(C).C', \sigma) &= (0, C) \ \wedge \ (0, C') \end{aligned}$$

   All such transitions, except for those involving $\forall R^*.C$ and $\exists R^*.C$, inductively decompose concepts and roles, and move to appropriate states of the automaton and nodes of the tree. The transitions involving $\forall R^*.C$ treat $\forall R^*.C$ as the equivalent concept $C \sqcap \forall R.\forall R^*.C$, and the transitions involving $\exists R^*.C$ treat $\exists R^*.C$ as the equivalent concept $C \sqcup \exists R.\exists R^*.C$.

5. For each concept of the form $\leqslant 1\, Q$ in $CL_{nnf}(C)$ and each $\sigma \in \Sigma$, there is a transition

$$\begin{aligned} \delta(\leqslant 1\, Q, \sigma) &= ((0, Q^-) \ \wedge \ \textstyle\bigwedge_{1 \leq i \leq k_{C_0}} (i, \neg Q)) \ \vee \\ &\quad ((0, \neg Q^-) \ \wedge \ \textstyle\bigwedge_{1 \leq i < j \leq k_{C_0}} ((i, \neg Q) \vee (j, \neg Q))) \end{aligned}$$

   Such transitions check that, for a node x labeled with $\leqslant 1\, Q$, there exists at most one node (among the predecessor and the successors of x) reachable from x through Q.

6. For each concept of the form $\neg \leqslant 1\, Q$ in $CL_{nnf}(C)$ and each $\sigma \in \Sigma$, there is a

transition

$$\delta(\neg \leqslant 1\, Q, \sigma) \;=\; ((0, Q^-) \;\wedge\; \bigvee_{1 \leq i \leq k_{C_0}} (i, Q)) \;\vee\; \bigvee_{1 \leq i < j \leq k_{C_0}} ((i, Q) \wedge (j, Q))$$

Such transitions check that, for a node x labeled with $\neg \leqslant 1\, Q$, there exist at least two nodes (among the predecessor and the successors of x) reachable from x through Q.

- The set F of final states is the set of concepts in $CL_{nnf}(C_0)$ of the form $\forall R^*.C$. Observe that concepts of the form $\exists R^*.C$ are not final states, and this is sufficient to guarantee that such concepts are satisfied in all accepting runs of the automaton.

A run of the automaton $\mathbf{A}_{C_0}$ on an infinite tree starts in the root checking that C_0 holds there (item 1 above). It does so by inductively decomposing $nnf(C_0)$ while appropriately navigating the tree (items 3 and 4) until it arrives to atomic concepts, functional restrictions, and their negations. These are checked locally (items 2, 5 and 6). Concepts of the form $\forall R^*.C$ and $\exists R^*.C$ are propagated using the equivalent concepts $C \sqcap \forall R.\forall R^*.C$ and $C \sqcup \exists R.\exists R^*.C$, respectively. It is only the propagation of such concepts that may generate infinite branches in a run. Now, a run of the automaton may contain an infinite branch in which $\exists R^*.C$ is always resolved by choosing the disjunct $\exists R.\exists R^*.C$, without ever choosing the disjunct C. This infinite branch in the run corresponds to an infinite path in the tree where R is iterated forever and in which C is never fulfilled. However, the semantics of $\exists R^*.C$ requires that C is fulfilled after a finite number of iterations of R. Hence such an infinite path cannot be used to satisfy $\exists R^*.C$. The acceptance condition of the automaton, which requires that each infinite branch in a run contains a state of the form $\forall R^*.C$, rules out such infinite branches in accepting runs. Indeed, a run always deferring the fulfillment of C will contain an infinite branch where all states have the form $\exists R_1. \cdots \exists R_n.\exists R^*.C$, with $n \geq 0$ and $R_1 \circ \cdots \circ R_n$ a postfix of R. Observe that the only remaining infinite branches in a run are those that arise by propagating concepts of the form $\forall R^*.C$ indefinitely often. The acceptance condition allows for such branches.

Given a labeled tree $\mathcal{T} = (T, V)$ accepted by $\mathbf{A}_{C_0}$, we define an interpretation $\mathcal{I}_\mathcal{T} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ as follows. First, we define for each atomic role P, a relation $\mathcal{R}_P$ as follows: $\mathcal{R}_P = \{\ (x, xi) \mid V(xi) = (\alpha, P) \text{ for some } \alpha \in 2^\mathcal{A}\ \} \cup \{\ (xi, x) \mid V(xi) = (\alpha, P^-) \text{ for some } \alpha \in 2^\mathcal{A}\ \}$. Then, using such relations, we define:

- $\Delta^\mathcal{I} = \{\ x \mid (\varepsilon, x) \in (\bigcup_P (\mathcal{R}_P \cup \mathcal{R}_P^-))^*\ \}$;
- $A^\mathcal{I} = \Delta^\mathcal{I} \cap \{\ x \mid V(x) = (\alpha, q) \text{ and } A \in \alpha, \text{ for some } \alpha \in 2^\mathcal{A} \text{ and } q \in \mathcal{B} \cup \{P_{dum}\}\ \}$, for each atomic concept A;

- $P^{\mathcal{I}} = (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \cap \mathcal{R}_P$, for each atomic role P.

Lemma 5.7 *If a labeled tree $\mathcal{T}$ is accepted by $\mathbf{A}_{C_0}$, then $\mathcal{I}_{\mathcal{T}}$ is a model of C_0.*

Conversely, given a tree model $\mathcal{I}$ of C_0 with branching degree k_{C_0}, we can obtain a labeled tree $\mathcal{T}_{\mathcal{I}} = (T, V)$ (with branching degree k_{C_0}) as follows:

- $T = \Delta^{\mathcal{I}}$;
- $V(\varepsilon) = (\alpha, P_{dum})$, where $\alpha = \{A \mid \varepsilon \in A^{\mathcal{I}}\}$;
- $V(xi) = (\alpha, Q)$, where $\alpha = \{A \mid xi \in A^{\mathcal{I}}\}$ and $(x, xi) \in Q^{\mathcal{I}}$.

Lemma 5.8 *If $\mathcal{I}$ is a tree model of C_0 with branching degree k_{C_0}, then $\mathcal{T}_{\mathcal{I}}$ is a labeled tree accepted by $\mathbf{A}_{C_0}$.*

From the lemmas above and the tree model property of $\mathcal{ALCFI}_{reg}$ (Theorem 5.6), we get the following result.

Theorem 5.9 *An $\mathcal{ALCFI}_{reg}$ concept C_0 is satisfiable if and only if the set of trees accepted by $\mathbf{A}_{C_0}$ is not empty.*

From this theorem, it follows that we can use algorithms for non-emptiness of 2ATAs to check satisfiability in $\mathcal{ALCFI}_{reg}$. It turns out that such a decision procedure is indeed optimal w.r.t. the computational complexity. The 2ATA $\mathbf{A}_{C_0}$ has a number of states that is linear in the size of C_0, while the alphabet is exponential in the number of atomic concepts occurring in C_0. By Theorem 5.5 we get an upper bound for reasoning in $\mathcal{ALCFI}_{reg}$ that matches the ExpTime lower bound.

Theorem 5.10 *Concept satisfiability (and hence logical implication) in $\mathcal{ALCFI}_{reg}$ is* ExpTime*-complete.*

Functional restrictions, in the context of expressive description logics that include inverse roles and TBox axioms, were originally studied in [De Giacomo and Lenzerini, 1994a; De Giacomo, 1995] using the so called *axiom schema instantiation* technique. The technique is based on the idea of devising an axiom schema corresponding to the property of interest (e.g., functional restrictions) and instantiating such a schema to a finite (polynomial) number of concepts. A nice illustration of this technique is the reduction of *converse*-PDL to PDL in [De Giacomo, 1996]. Axiom schema instantiation can be used to show that reasoning w.r.t. TBoxes is ExpTime-complete in significant sub-cases of $\mathcal{ALCFI}_{reg}$ (such as reasoning w.r.t. $\mathcal{ALCFI}$ TBoxes [Calvanese *et al.*, 2001b]). However, it is still open whether it can be applied to show ExpTime-completeness of $\mathcal{ALCFI}_{reg}$. The attempt in this direction presented in [De Giacomo and Lenzerini, 1994a; De Giacomo, 1995] turned out to be incomplete [Zakharyaschev, 2000].

5.4 Qualified number restrictions

Next we deal with *qualified number restrictions*, which are the most general form of number restrictions, and allow for specifying arbitrary cardinality constraints on roles with role-fillers belonging to a certain concept. In particular we will consider qualified number restrictions on basic roles, i.e., atomic roles and their inverse. By adding such constructs to $\mathcal{ALCI}_{reg}$ we obtain the description logic $\mathcal{ALCQI}_{reg}$. The PDL corresponding to $\mathcal{ALCQI}_{reg}$ is an extension of *converse*-PDL with "graded modalities" [Fattorosi-Barnaba and De Caro, 1985; Van der Hoek and de Rijke, 1995; Tobies, 1999c] on atomic programs and their converse.

Formally, $\mathcal{ALCQI}_{reg}$ is obtained from $\mathcal{ALCI}_{reg}$ by adding *qualified number restrictions* of the form $\leqslant n\, QC$ and $\geqslant n\, QC$, where n is a nonnegative integer, Q is a basic role, and C is an $\mathcal{ALCQI}_{reg}$ concept. Such constructs are interpreted as follows:

$$\begin{aligned}(\leqslant n\, QC)^{\mathcal{I}} &= \{o \in \Delta^{\mathcal{I}} \mid |\{o' \in \Delta^{\mathcal{I}} \mid (o,o') \in Q^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\}| \leq n\} \\ (\geqslant n\, QC)^{\mathcal{I}} &= \{o \in \Delta^{\mathcal{I}} \mid |\{o' \in \Delta^{\mathcal{I}} \mid (o,o') \in Q^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\}| \geq n\}\end{aligned}$$

Reasoning in $\mathcal{ALCQI}_{reg}$ is still ExpTime-complete under the standard assumption in description logics, that numbers in number restrictions are represented in unary[1]. This could be shown by extending the automata theoretic techniques introduced in Section 5.3 to deal also with qualified number restrictions. Here we take a different approach and study reasoning in $\mathcal{ALCQI}_{reg}$ by exhibiting a reduction from $\mathcal{ALCQI}_{reg}$ to $\mathcal{ALCFI}_{reg}$ [De Giacomo and Lenzerini, 1995; De Giacomo, 1995]. Since the reduction is polynomial, we get as a result ExpTime-completeness of $\mathcal{ALCQI}_{reg}$. The reduction is based on the notion of *reification*. Such a notion plays a major role in dealing with Boolean combinations of (atomic) roles [De Giacomo and Lenzerini, 1995; 1994c], as well as in extending expressive description logics with relation of arbitrary arity (see Section 5.7).

5.4.1 Reification of roles

Atomic roles are interpreted as binary relations. Reifying a binary relation means creating for each pair of individuals (o_1, o_2) in the relation an individual which is connected by means of two special roles V_1 and V_2 to o_1 and o_2, respectively. The set of such individuals represents the set of pairs forming the relation. However, the following problem arises: in general, there may be two or more individuals being all connected by means of V_1 and V_2 to o_1 and o_2 respectively, and thus all representing

[1] In [Tobies, 2001a] techniques for dealing with qualified number restrictions with numbers coded in binary are presented, and are used to show that even under this assumption reasoning over $\mathcal{ALCQI}$ knowledge bases can be done in ExpTime.

the same pair (o_1, o_2). Obviously, in order to have a correct representation of a relation, such a situation must be avoided.

Given an atomic role P, we call its *reified form* the following role

$$V_1^- \circ id(A_P) \circ V_2$$

where A_P is a new atomic concept denoting individuals representing the tuples of the relation associated with P, and V_1 and V_2 denote two functional roles that connect each individual in A_P to the first and the second component respectively of the tuple represented by the individual. Observe that there is a clear symmetry between the role $V_1^- \circ id(A_P) \circ V_2$ and its inverse $V_2^- \circ id(A_P) \circ V_1$.

Definition 5.11 Let C be an $\mathcal{ALCQI}_{reg}$ concept. The *reified counterpart* $\xi_1(C)$ of C is the conjunction of two concepts, $\xi_1(C) = \xi_0(C) \sqcap \Theta_1$, where:

- $\xi_0(C)$ is obtained from the original concept C by (i) replacing every atomic role P by the complex role $V_1^- \circ id(A_P) \circ V_2$, where V_1 and V_2 are new atomic roles (the only ones present after the transformation) and A_P is a new atomic concept; (ii) and then re-expressing every qualified number restriction

$$\begin{array}{lll}
\leqslant n\,(V_1^- \circ id(A_P) \circ V_2).D & \text{as} & \leqslant n\,V_1^-.(A_P \sqcap \exists V_2.D) \\
\geqslant n\,(V_1^- \circ id(A_P) \circ V_2).D & \text{as} & \geqslant n\,V_1^-.(A_P \sqcap \exists V_2.D) \\
\leqslant n\,(V_2^- \circ id(A_P) \circ V_1).D & \text{as} & \leqslant n\,V_2^-.(A_P \sqcap \exists V_1.D) \\
\geqslant n\,(V_2^- \circ id(A_P) \circ V_1).D & \text{as} & \geqslant n\,V_2^-.(A_P \sqcap \exists V_1.D)
\end{array}$$

- $\Theta_1 = \forall(V_1 \sqcup V_2 \sqcup V_1^- \sqcup V_2^-)^*.(\leqslant 1\,V_1 \sqcap \leqslant 1\,V_2)$. ∎

The next theorem guarantees that, without loss of generality, we can restrict our attention to models of $\xi_1(C)$ that correctly represent relations associated with atomic roles, i.e., models in which each tuple of such relations is represented by a single individual.

Theorem 5.12 *If the concept $\xi_1(C)$ has a model $\mathcal{I}$ then it has a model $\mathcal{I}'$ such that for each $(o, o') \in (V_1^- \circ id(A_{P_i}) \circ V_2)^{\mathcal{I}'}$ there is exactly one individual $o_{oo'}$ such that $(o_{oo'}, o) \in V_1^{\mathcal{I}'}$ and $(o_{oo'}, o') \in V_2^{\mathcal{I}'}$. That is, for all $o_1, o_2, o, o' \in \Delta^{\mathcal{I}'}$ such that $o_1 \neq o_2$ and $o \neq o'$, the following condition holds:*

$$o_1, o_2 \in A_{P_i}^{\mathcal{I}'} \quad \supset \quad \neg((o_1, o) \in V_1^{\mathcal{I}'} \wedge (o_2, o) \in V_1^{\mathcal{I}'} \wedge (o_1, o') \in V_2^{\mathcal{I}'} \wedge (o_2, o') \in V_2^{\mathcal{I}'}).$$

The proof of Theorem 5.12 exploits the *disjoint union model property*: let C be an $\mathcal{ALCQI}_{reg}$ concept and $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ and $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be two models of C, then also the interpretation $\mathcal{I} \uplus \mathcal{J} = (\Delta^{\mathcal{I}} \uplus \Delta^{\mathcal{J}}, \cdot^{\mathcal{I}} \uplus \cdot^{\mathcal{J}})$ which is the disjoint union of $\mathcal{I}$ and $\mathcal{J}$, is a model of C. We remark that most description logics have such a property, which is, in fact, typical of modal logics. Without going into details, we

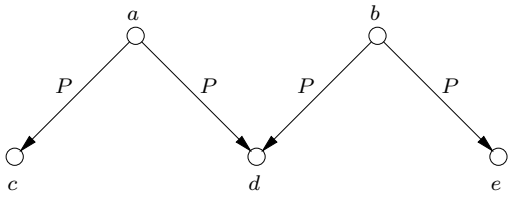

Fig. 5.1. A model of the $\mathcal{ALCQI}_{reg}$ concept $C_0 = \exists P.(=2\,P^-.(=2\,P.\top))$.

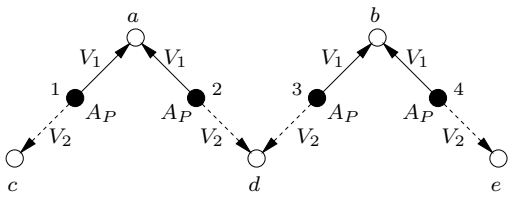

Fig. 5.2. A model of the reified counterpart $\xi_1(C_0)$ of C_0.

just mention that the model $\mathcal{I}'$ is constructed from $\mathcal{I}$ as the disjoint union of several copies of $\mathcal{I}$, in which the extension of role V_2 is modified by exchanging, in those instances that cause a wrong representation of a role, the second component with a corresponding individual in one of the copies of $\mathcal{I}$.

By using Theorem 5.12 we can prove the result below.

Theorem 5.13 *An $\mathcal{ALCQI}_{reg}$ concept C is satisfiable if and only if its reified counterpart $\xi_1(C)$ is satisfiable.*

5.4.2 Reducing $\mathcal{ALCQI}_{reg}$ to $\mathcal{ALCFI}_{reg}$

By Theorem 5.13, we can concentrate on the reified counterparts of $\mathcal{ALCQI}_{reg}$ concepts. Note that these are $\mathcal{ALCQI}_{reg}$ concepts themselves, but their special form allows us to convert them into $\mathcal{ALCFI}_{reg}$ concepts. Intuitively, we represent the role V_i^-, $i = 1, 2$ (recall that V_i is functional while V_i^- is not), by the role $F_{V_i} \circ {F'_{V_i}}^*$, where F_{V_i} and F'_{V_i} are new functional roles[1]. The main point of such transformation is that it is easy to express qualified number restrictions as constraints on the chain of $(F_{V_i} \circ {F'_{V_i}}^*)$-successor of an individual. Formally, we define the $\mathcal{ALCFI}_{reg}$-counterpart of an $\mathcal{ALCQI}_{reg}$ concept as follows.

Definition 5.14 Let C be an $\mathcal{ALCQI}_{reg}$ concept and $\xi_1(C) = \xi_0(C) \sqcap \Theta_1$ its reified counterpart. The *$\mathcal{ALCFI}_{reg}$-counterpart* $\xi_2(C)$ of C is the conjunction of two concepts, $\xi_2(C) = \xi'_0(C) \wedge \Theta_2$, where:

[1] The idea of expressing nonfunctional roles by means of chains of functional roles is due to Parikh [1981], who used it to reduce standard PDL to DPDL.

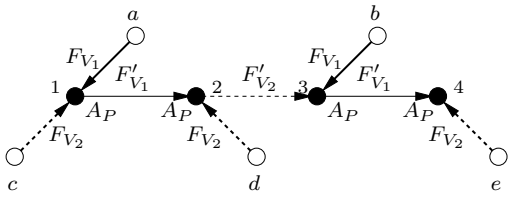

Fig. 5.3. A model of the $\mathcal{ALCFI}$-counterpart $\xi_2(C_0)$ of C_0.

- $\xi'_0(C)$ is obtained from $\xi_0(C)$ by simultaneously replacing:[2]
  - every occurrence of role V_i in constructs different from qualified number restrictions by $(F_{V_i} \circ {F'_{V_i}}^*)^-$, where F_{V_i} and F'_{V_i} are new atomic roles;
  - every $\leqslant n\, V_i^-.D$ by $\forall(F_{V_i} \circ {F'_{V_i}}^* \circ (id(D) \circ {F'_{V_i}}^+)^n).\neg D$;
  - every $\geqslant n\, V_i^-.D$ by $\exists(F_{V_i} \circ {F'_{V_i}}^* \circ (id(D) \circ {F'_{V_i}}^+)^{n-1}).D$.
- $\Theta_2 = \forall(\bigsqcup_{i=1,2}(F_{V_i} \sqcup F'_{V_i} \sqcup F_{V_i}^- \sqcup {F'_{V_i}}^-))^*.(\theta_1 \sqcap \theta_2)$, with θ_i of the form:

$$\leqslant 1\, F_{V_i} \sqcap \leqslant 1\, F'_{V_i} \sqcap \leqslant 1\, F_{V_i}^- \sqcap \leqslant 1\, {F'_{V_i}}^- \sqcap \neg(\exists F_{V_i}^-.\top \sqcap \exists {F'_{V_i}}^-.\top). \qquad \blacksquare$$

Observe that Θ_2 constrains each model $\mathcal{I}$ of $\xi_2(C)$ so that the relations $F_{V_i}^{\mathcal{I}}$, ${F'_{V_i}}^{\mathcal{I}}$, $(F_{V_i}^-)^{\mathcal{I}}$, and $({F'_{V_i}}^-)^{\mathcal{I}}$ are partial functions, and each individual cannot be linked to other individuals by both $(F_{V_i}^-)^{\mathcal{I}}$ and $({F'_{V_i}}^-)^{\mathcal{I}}$. As a consequence, we get that $((F_{V_i} \circ {F'_{V_i}}^*)^-)^{\mathcal{I}}$ is a partial function. This allows us to reconstruct the extension of V_i, as required.

We illustrate the basic relationships between a model of an $\mathcal{ALCQI}_{reg}$ concept and the models of its reified counterpart and $\mathcal{ALCFI}_{reg}$-counterpart by means of an example.

Example 5.15 Consider the concept

$$C_0 = \exists P.(=2\, P^-.(=2\, P.\top))$$

and consider the model $\mathcal{I}$ of C_0 depicted in Figure 5.1, in which $a \in C_0^{\mathcal{I}}$. Such a model corresponds to a model $\mathcal{I}'$ of the reified counterpart $\xi_1(C_0)$ of C_0, shown in Figure 5.2. The model $\mathcal{I}'$ of $\xi_1(C_0)$ in turn, corresponds to a model $\mathcal{I}''$ of the $\mathcal{ALCFI}_{reg}$-counterpart $\xi_2(C_0)$ of C_0, shown in Figure 5.3. Notice that, from $\mathcal{I}''$ we can easily reconstruct $\mathcal{I}'$, and from $\mathcal{I}'$ the model $\mathcal{I}$ of the original concept. $\blacksquare$

It can be shown that $\xi_1(C)$ is satisfiable if and only if $\xi_2(C)$ is satisfiable. Since, as it is easy to see, the size of $\xi_2(C)$ is polynomial in the size of C, we get the following characterization of the computational complexity of reasoning in $\mathcal{ALCQI}_{reg}$.

[2] Here R^+ stands for $R \circ R^*$ and R^n stands for $R \circ \cdots \circ R$ (n times).

Theorem 5.16 *Concept satisfiability (and hence logical implication) in* $\mathcal{ALCQI}_{reg}$ *is* ExpTime*-complete.*

5.5 Objects

In this section, we review results involving knowledge on individuals expressed in terms of membership assertions. Given an alphabet $\mathcal{O}$ of symbols for individuals, a *(membership) assertion* has one of the following forms:

$$C(a) \qquad\qquad P(a_1, a_2)$$

where C is a concept, P is an atomic role, and a, a_1, a_2 belong to $\mathcal{O}$. An interpretation $\mathcal{I}$ is extended so as to assign to each $a \in \mathcal{O}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ in such a way that the *unique name assumption* is satisfied, i.e., different elements are assigned to different symbols in $\mathcal{O}$. $\mathcal{I}$ *satisfies* $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and $\mathcal{I}$ *satisfies* $P(a_1, a_2)$ if $(a_1^{\mathcal{I}}, a_2^{\mathcal{I}}) \in R^{\mathcal{I}}$. An *ABox* $\mathcal{A}$ is a finite set of membership assertions, and an interpretation $\mathcal{I}$ is called a *model of* $\mathcal{A}$ if $\mathcal{I}$ satisfies every assertion in $\mathcal{A}$.

A *knowledge base* is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a TBox, and $\mathcal{A}$ is an ABox. An interpretation $\mathcal{I}$ is called a *model of* $\mathcal{K}$ if it is a model of both $\mathcal{T}$ and $\mathcal{A}$. $\mathcal{K}$ is *satisfiable* if it has a model, and $\mathcal{K}$ *logically implies* an assertion β, denoted $\mathcal{K} \models \beta$, where β is either an inclusion or a membership assertion, if every model of $\mathcal{K}$ satisfies β. Logical implication can be reformulated in terms of unsatisfiability: e.g., $\mathcal{K} \models C(a)$ iff $\mathcal{K} \cup \{\neg C(a)\}$ is unsatisfiable; similarly $\mathcal{K} \models C_1 \sqsubseteq C_2$ iff $\mathcal{K} \cup \{(C_1 \sqcap \neg C_2)(a')\}$ is unsatisfiable, where a' does not occur in $\mathcal{K}$. Therefore, we only need a procedure for checking satisfiability of a knowledge base.

Next we illustrate the technique for reasoning on $\mathcal{ALCQI}_{reg}$ knowledge bases [De Giacomo and Lenzerini, 1996]. The basic idea is as follows: checking the satisfiability of an $\mathcal{ALCQI}_{reg}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is polynomially reduced to checking the satisfiability of an $\mathcal{ALCQI}_{reg}$ knowledge base $\mathcal{K}' = (\mathcal{T}', \mathcal{A}')$, whose ABox $\mathcal{A}'$ is made of a single membership assertion of the form $C(a)$. In other words, the satisfiability of $\mathcal{K}$ is reduced to the satisfiability of the concept C w.r.t. the TBox $\mathcal{T}'$ of the resulting knowledge base. The latter reasoning service can be realized by means of the method presented in Section 5.4, and, as we have seen, is ExpTime-complete. Thus, by means of the reduction, we get an ExpTime algorithm for satisfiability of $\mathcal{ALCQI}_{reg}$ knowledge bases, and hence for all standard reasoning services on $\mathcal{ALCQI}_{reg}$ knowledge bases.

Definition 5.17 Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be an $\mathcal{ALCQI}_{reg}$ knowledge base. We call *the reduced form* of $\mathcal{K}$ the $\mathcal{ALCQI}_{reg}$ knowledge base $\mathcal{K}' = (\mathcal{T}', \mathcal{A}')$ defined as follows. We introduce a new atomic role *create*, and for each individual a_i, $i = 1, \ldots, m$,

occurring in $\mathcal{A}$, a new atomic concept A_i. Then:

$$\mathcal{A}' = \{(\exists create.A_1 \sqcap \cdots \sqcap \exists create.A_m)(g)\},$$

where g is a new individual (the only one present in $\mathcal{A}'$), and $\mathcal{T}' = \mathcal{T} \cup \mathcal{T}_{\mathcal{A}} \cup \mathcal{T}_{aux}$, where:

- $\mathcal{T}_{\mathcal{A}}$ is constituted by the following inclusion axioms:
  - for each membership assertion $C(a_i) \in \mathcal{A}$, one inclusion axiom
    $$A_i \sqsubseteq C$$
  - for each membership assertion $P(a_i, a_j) \in \mathcal{A}$, two inclusion axioms
    $$\begin{aligned} A_i &\sqsubseteq \exists P.A_j \sqcap \leqslant 1\, P.A_j \\ A_j &\sqsubseteq \exists P^-.A_i \sqcap \leqslant 1\, P^-.A_i \end{aligned}$$
  - for each pair of distinct individuals a_i and a_j occurring in $\mathcal{A}$, one inclusion axiom
    $$A_i \sqsubseteq \neg A_j$$
- $\mathcal{T}_{aux}$ is constituted by one inclusion axiom (U stands for $(P_1 \sqcup \cdots \sqcup P_n \sqcup P_1^- \sqcup \cdots \sqcup P_n^-)^*$, where $P_1, \ldots, P_n$ are all atomic roles in $\mathcal{T} \cup \mathcal{T}_{\mathcal{A}}$):
  $$A_i \sqcap C \sqsubseteq \forall U.(\neg A_i \sqcup C)$$
  for each A_i occurring in $\mathcal{T} \cup \mathcal{T}_{\mathcal{A}}$ and each $C \in CL_{ext}(\mathcal{T} \cup \mathcal{T}_{\mathcal{A}})$, where $CL_{ext}(\mathcal{T} \cup \mathcal{T}_{\mathcal{A}})$ is a suitably extended syntactic closure of $\mathcal{T} \cup \mathcal{T}_{\mathcal{A}}$ [1] whose size is polynomially related to the size of $\mathcal{T} \cup \mathcal{T}_{\mathcal{A}}$ [De Giacomo and Lenzerini, 1996]. ∎

To understand how the reduced form $\mathcal{K}' = (\mathcal{T}', \mathcal{A}')$ relates to the original knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, first, observe that the ABox $\mathcal{A}'$ is used to force the existence of the only individual g, connected by the role *create* to one instance of each A_i. It can be shown that this allows us to restrict the attention to models of $\mathcal{K}'$ that represent a graph connected to g, i.e., models $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{K}'$ such that $\Delta^{\mathcal{I}} = \{g\} \cup \{s' \mid (g, s') \in create^{\mathcal{I}} \circ (\bigcup_P (P^{\mathcal{I}} \cup P^{\mathcal{I}-})^*)\}$.

The TBox $\mathcal{T}'$ consists of three parts $\mathcal{T}$, $\mathcal{T}_{\mathcal{A}}$, and $\mathcal{T}_{aux}$. $\mathcal{T}$ are the original inclusion axioms. $\mathcal{T}_{\mathcal{A}}$ is what we may call a "naive encoding" of the original ABox $\mathcal{A}$ as inclusion axioms. Indeed, each individual a_i is represented in $\mathcal{T}_{\mathcal{A}}$ as a new atomic concept A_i (disjoint from the other A_j's), and the membership assertions in the original ABox $\mathcal{A}$ are represented as inclusion axioms in $\mathcal{T}_{\mathcal{A}}$ involving such new atomic concepts. However $\mathcal{T} \cup \mathcal{T}_{\mathcal{A}}$ alone does not suffice to represent faithfully (w.r.t. the reasoning services we are interested in) the original knowledge base,

[1] The syntactic closure of a TBox is the syntactic closure of the concept obtained by internalizing the axioms of the TBox.

because an individual a_i in $\mathcal{K}$ is represented by the set of instances of A_i in $\mathcal{K}'$. In order to reduce the satisfiability of $\mathcal{K}'$ to the satisfiability of $\mathcal{K}$, we must be able to single out, for each A_i, one instance of A_i representative of a_i. For this purpose, we need to include in $\mathcal{T}'$ a new part, called $\mathcal{T}_{aux}$, which contains inclusion axioms of the form:

$$(A_i \sqcap C) \sqsubseteq \forall U.(\neg A_i \sqcup C)$$

Intuitively, such axioms say that, if an instance of A_i is also an instance of C, then every instance of A_i is an instance of C. Observe that, if we could add an infinite set of axioms of this form, one for each possible concept of the language (i.e., an axiom schema), we could safely restrict our attention to models of $\mathcal{K}'$ with just one instance for every concept A_i, since there would be no way in the logic to distinguish two instances of A_i one from the other. What is shown by De Giacomo and Lenzerini [1996] is that in fact we do need only a polynomial number of such inclusion axioms (as specified by $\mathcal{T}_{aux}$) in order to be able to identify, for each i, an instance of A_i as representative of a_i. This allows us to prove that the existence of a model of $\mathcal{K}'$ implies the existence of a model of $\mathcal{K}$.

Theorem 5.18 *Knowledge base satisfiability (and hence every standard reasoning service) in $\mathcal{ALCQI}_{reg}$ is* ExpTime*-complete.*

Using a similar approach, De Giacomo and Lenzerini [1994a] and De Giacomo [1995] extend $\mathcal{ALCQ}_{reg}$ and $\mathcal{ALCI}_{reg}$ by adding special atomic concepts A_a, called *nominals*, having exactly one single instance a, i.e., the individual they name. Nominals may occur in concepts exactly as atomic concepts, and hence they constitute one of the most flexible ways to express knowledge about single individuals.

By using nominals we can capture the "one-of" construct, having the form $\{a_1, \ldots, a_n\}$, denoting the concept made of exactly the enumerated individuals $a_1, \ldots, a_n$ [1]. We can also capture the "fills" construct, having the form $R : a$, denoting those individuals having the individual a as a role filler of R [2] (see [Schaerf, 1994b] and references therein for further discussion on these constructs).

Let us denote with $\mathcal{ALCQO}_{reg}$ and $\mathcal{ALCIO}_{reg}$ the description logics resulting by adding nominals to $\mathcal{ALCQ}_{reg}$ and $\mathcal{ALCI}_{reg}$ respectively. De Giacomo and Lenzerini [1994a] and De Giacomo [1995] polynomially reduce satisfiability in $\mathcal{ALCQO}_{reg}$ and $\mathcal{ALCIO}_{reg}$ knowledge bases to satisfiability of $\mathcal{ALCQ}_{reg}$ and $\mathcal{ALCI}_{reg}$ concepts respectively, hence showing decidability and ExpTime-completeness of reasoning in these logics. ExpTime-completeness does not hold for $\mathcal{ALCQIO}_{reg}$,

[1] Actually, nominals and the one-of construct are essentially equivalent, since a name A_a is equivalent to $\{a\}$ and $\{a_1, \ldots, a_n\}$ is equivalent to $A_{a_1} \sqcup \cdots \sqcup A_{a_n}$.

[2] The "fills" construct $R : a$ is captured by $\exists R.A_a$.

i.e., $\mathcal{ALCQI}_{reg}$ extended with nominals. Indeed, a result by Tobies [1999a; 1999b] shows that reasoning in such a logic is NExpTime-hard. Its decidability still remains an open problem.

The notion of nominal introduced above has a correspondent in modal logic [Prior, 1967; Bull, 1970; Blackburn and Spaan, 1993; Gargov and Goranko, 1993; Blackburn, 1993]. Nominals have also been studied within the setting of PDLs [Passy and Tinchev, 1985; Gargov and Passy, 1988; Passy and Tinchev, 1991]. The results for $\mathcal{ALCQO}_{reg}$ and $\mathcal{ALCIO}_{reg}$ are immediately applicable also in the setting of PDLs. In particular, the PDL corresponding to $\mathcal{ALCQO}_{reg}$ is standard PDL augmented with nominals and graded modalities (qualified number restrictions). It is an extension of *deterministic combinatory PDL*, DcPDL, which is essentially DPDL augmented with nominals. The decidability of DcPDL is established by Passy and Tinchev [1985], who also prove that satisfiability can be checked in nondeterministic double exponential time. This is tightened by the result above on ExpTime-completeness of $\mathcal{ALCQO}_{reg}$, which says that DcPDL is in fact ExpTime-complete, thus closing the previous gap between the upper bound and the lower bound. The PDL corresponding to $\mathcal{ALCIO}_{reg}$ is *converse*-PDL augmented with nominals, which is also called *converse combinatory PDL*, CcPDL [Passy and Tinchev, 1991]. Such logic was not known to be decidable [Passy and Tinchev, 1991]. Hence the results mentioned above allow us to establish the decidability of CcPDL and to precisely characterize the computational complexity of satisfiability (and hence of logical implication) as ExpTime-complete.

5.6 Fixpoint constructs

Decidable description logics equipped with explicit fixpoint constructs have been devised in order to model inductive and coinductive data structures such as lists, streams, trees, etc. [De Giacomo and Lenzerini, 1994d; Schild, 1994; De Giacomo and Lenzerini, 1997; Calvanese *et al.*, 1999c]. Such logics correspond to extensions of the *propositional μ-calculus* [Kozen, 1983; Streett and Emerson, 1989; Vardi, 1998], a variant of PDL with explicit fixpoints that is used to express temporal properties of reactive and concurrent processes [Stirling, 1996; Emerson, 1996]. Such logics can also be viewed as a well-behaved fragment of first-order logic with fixpoints [Park, 1970; 1976; Abiteboul *et al.*, 1995].

Here, we concentrate on the description logic $\mu\mathcal{ALCQI}$ studied by Calvanese *et al.* [1999c]. Such a description logic is derived from $\mathcal{ALCQI}$ by adding *least and greatest fixpoint constructs*. The availability of explicit fixpoint constructs allows for expressing *inductive* and *coinductive* concepts in a natural way.

Example 5.19 Consider the concept Tree, representing trees, inductively defined as follows:

(i) An individual that is an $\mathsf{EmptyTree}$ is a Tree.
(ii) If an individual is a Node, has at most one parent, has some children, and all children are Trees, then such an individual is a Tree.

In other words, Tree is the concept with the smallest extension among those satisfying the assertions (i) and (ii). Such a concept is naturally expressed in $\mu\mathcal{ALCQI}$ by making use of the least fixpoint construct $\mu X.C$:

$$\mathsf{Tree} \equiv \mu X.(\mathsf{EmptyTree} \sqcup (\mathsf{Node} \sqcap \leqslant 1\, \mathsf{child}^- \sqcap \exists \mathsf{child}.\top \sqcap \forall \mathsf{child}.X)) \qquad \blacksquare$$

Example 5.20 Consider the well-known linear data structure, called stream. Streams are similar to lists except that, while lists can be considered as finite sequences of nodes, streams are infinite sequences of nodes. Such a data structure is captured by the concept Stream, coinductively defined as follows:

(i) An individual that is a Stream, is a Node and has a single successor which is a Stream.

In other words, Stream is the concept with the largest extension among those satisfying condition (i). Such a concept is naturally expressed in $\mu\mathcal{ALCQI}$ by making use of the greatest fixpoint construct $\nu X.C$:

$$\mathsf{Stream} \equiv \nu X.(\mathsf{Node} \sqcap \leqslant 1\, \mathsf{succ} \sqcap \exists \mathsf{succ}.X) \qquad \blacksquare$$

Let us now introduce $\mu\mathcal{ALCQI}$ formally. We make use of the standard first-order notions of *scope*, *bound* and *free occurrences* of variables, *closed formulae*, etc., treating μ and ν as quantifiers.

The primitive symbols in $\mu\mathcal{ALCQI}$ are *atomic concepts*, *(concept) variables*, and *atomic roles*. Concepts and roles are formed according to the following syntax

$$\begin{array}{lcl} C & \longrightarrow & A \mid \neg C \mid C_1 \sqcap C_2 \mid \geqslant n\, R.C \mid \mu X.C \mid X \\ R & \longrightarrow & P \mid P^- \end{array}$$

where A denotes an atomic concept, P an atomic role, C an arbitrary $\mu\mathcal{ALCQI}$ concept, R an arbitrary $\mu\mathcal{ALCQI}$ role (i.e., either an atomic role or the inverse of an atomic role), n a natural number, and X a variable.

The concept C in μXC must be *syntactically monotone*, that is, every free occurrence of the variable X in C must be in the scope of an even number of negations [Kozen, 1983]. This restriction guarantees that the concept C denotes a monotonic operator and hence both the least and the greatest fixpoints exist and are unique (see later).

In addition to the usual abbreviations used in $\mathcal{ALCQI}$, we introduce the abbreviation $\nu X.C$ for $\neg\mu X.\neg C[X/\neg X]$, where $C[X/\neg X]$ is the concept obtained by substituting all free occurrences of X with $\neg X$.

The presence of free variables does not allow us to extend the interpretation function $\cdot^{\mathcal{I}}$ directly to every concept of the logic. For this reason we introduce valuations. A *valuation* ρ on an interpretation $\mathcal{I}$ is a mapping from variables to subsets of $\Delta^{\mathcal{I}}$. Given a valuation ρ, we denote by $\rho[X/\mathcal{E}]$ the valuation identical to ρ except for the fact that $\rho[X/\mathcal{E}](X) = \mathcal{E}$.

Let $\mathcal{I}$ be an interpretation and ρ a valuation on $\mathcal{I}$. We assign meaning to concepts of the logic by associating to $\mathcal{I}$ and ρ an *extension function* $\cdot^{\mathcal{I}}_{\rho}$, mapping concepts to subsets of $\Delta^{\mathcal{I}}$, as follows:

$$\begin{aligned}
X^{\mathcal{I}}_{\rho} &= \rho(X) \subseteq \Delta^{\mathcal{I}} \\
A^{\mathcal{I}}_{\rho} &= A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}}_{\rho} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}_{\rho} \\
(C_1 \sqcap C_2)^{\mathcal{I}}_{\rho} &= (C_1)^{\mathcal{I}}_{\rho} \cap (C_2)^{\mathcal{I}}_{\rho} \\
(\geqslant n\, R.C)^{\mathcal{I}}_{\rho} &= \{s \in \Delta^{\mathcal{I}} \mid |\{s' \mid (s,s') \in R^{\mathcal{I}} \text{ and } s' \in C^{\mathcal{I}}_{\rho}\}| \geq n\} \\
(\mu X.C)^{\mathcal{I}}_{\rho} &= \bigcap \{\mathcal{E} \subseteq \Delta^{\mathcal{I}} \mid\ C^{\mathcal{I}}_{\rho[X/\mathcal{E}]} \subseteq \mathcal{E}\ \}
\end{aligned}$$

Observe that $C^{\mathcal{I}}_{\rho[X/\mathcal{E}]}$ can be seen as an operator from subsets $\mathcal{E}$ of $\Delta^{\mathcal{I}}$ to subsets of $\Delta^{\mathcal{I}}$, and that, by the syntactic restriction enforced on variables, such an operator is guaranteed to be monotonic w.r.t. set inclusion. $\mu X.C$ denotes the *least fixpoint* of the operator. Observe also that the semantics assigned to $\nu X.C$ is

$$(\nu X.C)^{\mathcal{I}}_{\rho} = \bigcup \{\mathcal{E} \subseteq \Delta^{\mathcal{I}} \mid\ \mathcal{E} \subseteq\ C^{\mathcal{I}}_{\rho[X/\mathcal{E}]}\ \}$$

Hence $\nu X.C$ denotes the *greatest fixpoint* of the operator.

In fact, we are interested in closed concepts, whose extension is independent of the valuation. For closed concepts we do not need to consider the valuation explicitly, and hence the notion of concept satisfiability, logical implication, etc. extend straightforwardly.

Exploiting a recent result on ExpTime decidability of modal μ-calculus with converse [Vardi, 1998], and exploiting a reduction technique for qualified number restrictions similar to the one presented in Section 5.4, Calvanese *et al.* [1999c] have shown that the same complexity bound holds also for reasoning in $\mu\mathcal{ALCQI}$.

Theorem 5.21 *Concept satisfiability (and hence logical implication) in* $\mu\mathcal{ALCQI}$ *is* ExpTime*-complete.*

For certain applications, variants of $\mu\mathcal{ALCQI}$ that allow for *mutual fixpoints*, de-

noting least and greatest solutions of *mutually* recursive equations, are of interest [Schild, 1994; Calvanese *et al.*, 1998c; 1999b]. Mutual fixpoints can be re-expressed by suitably nesting the kind of fixpoints considered here (see, for example, [de Bakker, 1980; Schild, 1994]). It is interesting to notice that, although the resulting concept may be exponentially large in the size of the original concept with mutual fixpoints, the number of (distinct) subconcepts of the resulting concept is polynomially bounded by the size of the original one. By virtue of this observation, and using the reasoning procedure by Calvanese *et al.* [1999c], we can strengthen the above result.

Theorem 5.22 *Checking satisfiability of a closed $\mu\mathcal{ALCQI}$ concept C can be done in deterministic exponential time w.r.t. the number of (distinct) subconcepts of C.*

Although $\mu\mathcal{ALCQI}$ does not have the rich variety of role constructs of $\mathcal{ALCQI}_{reg}$, it is actually an extension of $\mathcal{ALCQI}_{reg}$, since any $\mathcal{ALCQI}_{reg}$ concept can be expressed in $\mu\mathcal{ALCQI}$ using the fixpoint constructs in a suitable way. To express concepts involving complex role expressions, it suffices to resort to the following equivalences:

$$\begin{aligned} \exists(R_1 \circ R_2).C &= \exists R_1.\exists R_2.C \\ \exists(R_1 \sqcup R_2).C &= \exists R_1.C \sqcup \exists R_2.C \\ \exists R^*.C &= \mu X.(C \sqcup \exists R.X) \\ \exists id(D).C &= C \sqcap D. \end{aligned}$$

Note that, according to such equivalences, we have also that

$$\forall R^*.C = \nu X.(C \sqcap \forall R.X)$$

Calvanese *et al.* [1995] advocate a further construct corresponding to an implicit form of fixpoint, the so called *well-founded* concept construct $wf(R)$. Such construct is used to impose well-foundedness of chains of roles, and thus allows one to correctly capture inductive structures. Using explicit fixpoints, $wf(R)$ is expressed as $\mu X.(\forall R.X)$.

We remark that, in order to gain the ability of expressing inductively and coinductively defined concepts, it has been proposed to adopt ad hoc semantics for interpreting knowledge bases, specifically the *least fixpoint semantics* for expressing inductive concepts and the *greatest fixpoint semantics* for expressing coinductive ones (see Chapter 2 and also [Nebel, 1991; Baader, 1990a; 1991; Dionne *et al.*, 1992; Küsters, 1998; Buchheit *et al.*, 1998]). Logics equipped with fixpoint constructs allow for mixing statements interpreted according to the least and greatest fixpoint semantics in the same knowledge base [Schild, 1994; De Giacomo and Lenzerini, 1997], and thus can be viewed as a generalization of these approaches.

Recently, using techniques based on alternating two-way automata, it has been shown that the propositional μ-calculus with converse programs remains ExpTime-decidable when extended with nominals [Sattler and Vardi, 2001]. Such a logic corresponds to a description logic which could be called $\mu\mathcal{ALCIO}$.

5.7 Relations of arbitrary arity

A limitation of traditional description logics is that only binary relationships between instances of concepts can be represented, while in some real world situations it is required to model relationships among more than two objects. Such relationships can be captured by making use of relations of arbitrary arity instead of (binary) roles. Various extensions of description logics with relations of arbitrary arity have been proposed [Schmolze, 1989; Catarci and Lenzerini, 1993; De Giacomo and Lenzerini, 1994c; Calvanese *et al.*, 1997; 1998a; Lutz *et al.*, 1999].

We concentrate on the description logic $\mathcal{DLR}$ [Calvanese *et al.*, 1997; 1998a], which represents a natural generalization of traditional description logics towards n-ary relations. The basic elements of $\mathcal{DLR}$ are *atomic relations* and *atomic concepts*, denoted by $\mathbf{P}$ and A respectively. Arbitrary *relations*, of given *arity* between 2 and n_{max}, and arbitrary *concepts* are formed according to the following syntax

$$\begin{array}{lcl} \mathbf{R} & \longrightarrow & \top_n \mid \mathbf{P} \mid (\$i/n{:}\,C) \mid \neg\mathbf{R} \mid \mathbf{R}_1 \sqcap \mathbf{R}_2 \\ C & \longrightarrow & \top_1 \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid \exists[\$i]\mathbf{R} \mid \leqslant k\,[\$i]\mathbf{R} \end{array}$$

where i and j denote components of relations, i.e., integers between 1 and n_{max}, n denotes the arity of a relation, i.e., an integer between 2 and n_{max}, and k denotes a nonnegative integer. Concepts and relations must be *well-typed*, which means that only relations of the same arity n can be combined to form expressions of type $\mathbf{R}_1 \sqcap \mathbf{R}_2$ (which inherit the arity n), and $i \leq n$ whenever i denotes a component of a relation of arity n.

The semantics of $\mathcal{DLR}$ is specified through the usual notion of *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the *interpretation function* $\cdot^{\mathcal{I}}$ assigns to each concept C a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, and to each relation $\mathbf{R}$ of arity n a subset $\mathbf{R}^{\mathcal{I}}$ of $(\Delta^{\mathcal{I}})^n$, such that

the following conditions are satisfied

$$\begin{array}{rcl}
\top_n^{\mathcal{I}} & \subseteq & (\Delta^{\mathcal{I}})^n \\
\mathbf{P}^{\mathcal{I}} & \subseteq & \top_n^{\mathcal{I}} \\
(\neg \mathbf{R})^{\mathcal{I}} & = & \top_n^{\mathcal{I}} \setminus \mathbf{R}^{\mathcal{I}} \\
(\mathbf{R}_1 \sqcap \mathbf{R}_2)^{\mathcal{I}} & = & \mathbf{R}_1^{\mathcal{I}} \cap \mathbf{R}_2^{\mathcal{I}} \\
(\$i/n\colon C)^{\mathcal{I}} & = & \{(d_1, \ldots, d_n) \in \top_n^{\mathcal{I}} \mid d_i \in C^{\mathcal{I}}\} \\
\top_1^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \\
A^{\mathcal{I}} & \subseteq & \Delta^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C_1 \sqcap C_2)^{\mathcal{I}} & = & C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} \\
(\exists[\$i]\mathbf{R})^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid \exists (d_1, \ldots, d_n) \in \mathbf{R}^{\mathcal{I}}.\, d_i = d\} \\
(\leqslant k\,[\$i]\mathbf{R})^{\mathcal{I}} & = & \{d \in \Delta^{\mathcal{I}} \mid |\{(d_1, \ldots, d_n) \in \mathbf{R}_1^{\mathcal{I}} \mid d_i = d\}| \leq k\}
\end{array}$$

where $\mathbf{P}$, $\mathbf{R}$, $\mathbf{R}_1$, and $\mathbf{R}_2$ have arity n. Observe that $\top_1$ denotes the interpretation domain, while $\top_n$, for $n > 1$, does *not* denote the n-cartesian product of the domain, but only a subset of it, that covers all relations of arity n that are introduced. As a consequence, the "$\neg$" construct on relations expresses *difference of relations* rather than complement.

The construct $(\$i/n\colon C)$ denotes all tuples in $\top_n$ that have an instance of concept C as their i-th component, and therefore represents a kind of selection. Existential quantification and number restrictions on relations are a natural generalization of the corresponding constructs using roles. This can be seen by observing that, while for roles the "direction of traversal" is implicit, for a relation one needs to explicitly say which component is used to "enter" a tuple and which component is used to "exit" it.

$\mathcal{DLR}$ is in fact a proper generalization of $\mathcal{ALCQI}$. The traditional description logic constructs can be reexpressed in $\mathcal{DLR}$ as follows:

$$\begin{array}{lcl}
\exists P.C & \text{as} & \exists[\$1](P \sqcap (\$2/2\colon C)) \\
\exists P^{-}.C & \text{as} & \exists[\$2](P \sqcap (\$1/2\colon C)) \\
\forall P.C & \text{as} & \neg\exists[\$1](P \sqcap (\$2/2\colon \neg C)) \\
\forall P^{-}.C & \text{as} & \neg\exists[\$2](P \sqcap (\$1/2\colon \neg C)) \\
\leqslant k\, P.C & \text{as} & \leqslant k\,[\$1](P \sqcap (\$2/2\colon C)) \\
\leqslant k\, P^{-}.C & \text{as} & \leqslant k\,[\$2](P \sqcap (\$1/2\colon C))
\end{array}$$

Observe that the constructs using direct and inverse roles are represented in $\mathcal{DLR}$ by using binary relations and explicitly specifying the direction of traversal.

A TBox in $\mathcal{DLR}$ is a finite set of inclusion axioms on both concepts and relations of the form

$$C \sqsubseteq C' \qquad\qquad \mathbf{R} \sqsubseteq \mathbf{R}'$$

where $\mathbf{R}$ and $\mathbf{R}'$ are two relations of the same arity. The notions of an interpretation *satisfying* an assertion, and of *model* of a TBox are defined as usual.

The basic technique used in $\mathcal{DLR}$ to reason on relations is *reification* (see Section 5.4.1), which allows one to reduce logical implication in $\mathcal{DLR}$ to logical implication in $\mathcal{ALCQI}$. Reification for n-ary relations is similar to reification of roles (see Definition 5.11): A relation of arity n is reified by means of a new concept and n functional roles $f_1, \ldots, f_n$. Let the $\mathcal{ALCQI}$ TBox $\mathcal{T}'$ be the *reified counterpart* of a $\mathcal{DLR}$ TBox $\mathcal{T}$. A tuple of a relation R in a model of $\mathcal{T}$ is represented in a model of $\mathcal{T}'$ by an instance of the concept corresponding to R, which is linked through $f_1, \ldots, f_n$ respectively to n individuals representing the components of the tuple. In this case reification is further used to encode Boolean constructs on relations into the corresponding constructs on the concepts representing relations.

As for reification of roles (cf. Section 5.4.1), performing the reification of relations requires some attention, since the semantics of a relation rules out that there may be two identical tuples in its extension, i.e., two tuples constituted by the same components in the same positions. In the reified counterpart, on the other hand, one cannot explicitly rule out (e.g., by using specific axioms) the existence of two individuals o_1 and o_2 "representing" the same tuple, i.e., that are connected through $f_1, \ldots, f_n$ to exactly the same individuals denoting the components of the tuple. A model of the reified counterpart $\mathcal{T}'$ of $\mathcal{T}$ in which this situation occurs may not correspond directly to a model of $\mathcal{T}$, since by collapsing the two equivalent individuals into a tuple, axioms may be violated (e.g., cardinality constraints). However, also in this case the analogue of Theorem 5.12 holds, ensuring that from any model of $\mathcal{T}'$ one can construct a new one in which no two individuals represent the same tuple. Therefore one does not need to take this constraint explicitly into account when reasoning on the reified counterpart of a knowledge base with relations. Since reification is polynomial, from ExpTime decidability of logical implication in $\mathcal{ALCQI}$ (and ExpTime-hardness of logical implication in $\mathcal{ALC}$) we get the following characterization of the computational complexity of reasoning in $\mathcal{DLR}$ [Calvanese *et al.*, 1997]

Theorem 5.23 *Logical implication in $\mathcal{DLR}$ is* ExpTime*-complete.*

$\mathcal{DLR}$ can be extended to include regular expressions built over projections of relations on two of their components, thus obtaining $\mathcal{DLR}_{reg}$. Such a logic, which represents a generalization of $\mathcal{ALCQI}_{reg}$, allows for the internalization of a TBox. ExpTime decidability (and hence completeness) of $\mathcal{DLR}_{reg}$ can again be shown by exploiting reification of relations and reducing logical implication to concept satisfiability in $\mathcal{ALCQI}_{reg}$ [Calvanese *et al.*, 1998a]. Recently, $\mathcal{DLR}_{reg}$ has been extended to $\mathcal{DLR}_{\mu}$, which includes explicit fixpoint constructs on concepts, as those

introduced in Section 5.6. The ExpTime-decidability result extends to $\mathcal{DLR}_\mu$ as well [Calvanese *et al.*, 1999c].

Recently it has been observed that guarded fragments of first order logic [Andréka *et al.*, 1996; Grädel, 1999] (see Section 4.2.1), which include n-ary relations, share with description logics the "locality" of quantification. This makes them of interest as extensions of description logics with n-ary relations [Grädel, 1998; Lutz *et al.*, 1999]. Such description logics are incomparable in expressive power with $\mathcal{DLR}$ and its extensions: On the one hand the description logics corresponding to guarded fragments allow one to refer, by the use of explicit variables, to components of relations in a more flexible way than what is possible in $\mathcal{DLR}$. On the other hand such description logics lack number restrictions, and extending them with number restrictions leads to undecidability of reasoning. Also, reasoning in the guarded fragments is in general NExpTime-hard [Grädel, 1998; 1999] and thus more difficult than in $\mathcal{DLR}$ and its extensions, although PSpace-complete fragments have been identified [Lutz *et al.*, 1999].

5.7.1 Boolean constructs on roles and role inclusion axioms

Observe also that $\mathcal{DLR}$ (and $\mathcal{DLR}_{reg}$) allows for Boolean constructs on relations (with negation interpreted as difference) as well as relation inclusion axioms $\mathbf{R} \sqsubseteq \mathbf{R}'$. In fact, $\mathcal{DLR}$ (resp. $\mathcal{DLR}_{reg}$) can be viewed as a generalization of $\mathcal{ALCQI}$ (resp. $\mathcal{ALCQI}_{reg}$) extended with Boolean constructs on atomic and inverse atomic roles. Such extensions of $\mathcal{ALCQI}$ were first studied in [De Giacomo and Lenzerini, 1994c; De Giacomo, 1995], where logical implication was shown to be ExpTime-complete by a reduction to $\mathcal{ALCQI}$ (resp. $\mathcal{ALCQI}_{reg}$). The logics above do not allow for combining atomic roles with inverse roles in Boolean combinations and role inclusion axioms. Tobies [2001a] shows that, for $\mathcal{ALCQI}$ extended with arbitrary Boolean combinations of atomic and inverse atomic roles, logical implication remains in ExpTime. Note that, in all logics above, negation on roles is interpreted as difference. For results on the impact of full negation on roles see [Lutz and Sattler, 2001; Tobies, 2001a].

Horrocks *et al.* [2000b] investigate reasoning in $\mathcal{SHIQ}$, which is $\mathcal{ALCQI}$ extended with roles that are transitive and with role inclusion axioms on arbitrary roles (direct, inverse, and transitive). $\mathcal{SHIQ}$ does not include reflexive-transitive closure. However, transitive roles and role inclusions allow for expressing a universal role (in a connected model), and hence allow for internalizing TBoxes. Satisfiability and logical implication in $\mathcal{SHIQ}$ are ExpTime-complete [Tobies, 2001a]. The importance of $\mathcal{SHIQ}$ lies in the fact that it is the logic implemented by the current state-of-the-art description logic-based systems (cf. Chapters 8 and 9).

5.7.2 Structured objects

An alternative way to overcome the limitations that result from the restriction to binary relationships between concepts, is to consider the interpretation domain as being constituted by objects with a complex structure, and extend the description logics with constructs that allow one to specify such structure [De Giacomo and Lenzerini, 1995]. This approach is in the spirit of object-oriented data models used in databases [Lecluse and Richard, 1989; Bancilhon and Khoshafian, 1989; Hull, 1988], and has the advantage, with respect to introducing relationships, that all aspects of the domain to be modeled can be represented in a uniform way, as concepts whose instances have certain structures. In particular, objects can either be unstructured or have the structure of a *set* or of a *tuple*. For objects having the structure of a set a particular role allows one to refer to the members of the set, and similarly each component of a tuple can be referred to by means of the (implicitly functional) role that labels it.

In general, reasoning over structured objects can have a very high computational complexity [Kuper and Vardi, 1993]. However, reasoning over a significant fragment of structuring properties can be polynomial reduced to reasoning in traditional description logics, by exploiting again reification to deal with tuples and sets. Thus, for such a fragment, reasoning can be done in ExpTime [De Giacomo and Lenzerini, 1995]. An important aspect in exploiting description logics for reasoning over structured objects, is being able to limit the depth of the structure of an object to avoid infinite nesting of tuples or sets. This requires the use of a well-founded construct, which is a restricted form of fixpoint (see Section 5.6).

5.8 Finite model reasoning

For expressive description logics, in particular for those containing inverse roles and functionality, a TBox may admit only models with an infinite domain [Cosmadakis *et al.*, 1990; Calvanese *et al.*, 1994]. Similarly, there may be TBoxes in which a certain concept can be satisfied only in an infinite model. This is illustrated in the following example by Calvanese [1996c].

Example 5.24 Consider the TBox

$$\begin{aligned} \mathsf{FirstGuard} &\sqsubseteq \mathsf{Guard} \sqcap \forall \mathsf{shields}^{-}.\bot \\ \mathsf{Guard} &\sqsubseteq \exists \mathsf{shields} \sqcap \forall \mathsf{shields}.\mathsf{Guard} \sqcap \leqslant 1\, \mathsf{shields}^{-} \end{aligned}$$

In a model of this TBox, an instance of FirstGuard can have no shields-predecessor, while each instance of Guard can have at most one. Therefore, the existence of an instance of FirstGuard implies the existence of an infinite sequence of instances of Guard, each one connected through the role shields to the following one. This means

that FirstGuard can be satisfied in an interpretation with a domain of arbitrary cardinality, but not in interpretations with a finite domain. ∎

Note that the TBox above is expressed in a very simple description logic, in particular $\mathcal{AL}$ (cf. Chapter 2) extended with inverse roles and functionality.

A logic is said to have the *finite model property* if every satisfiable formula of the logic admits a *finite model*, i.e., a model with a finite domain. The example above shows that virtually all description logics including functionality, inverse roles, and TBox axioms (or having the ability to internalize them) lack the finite model property. The example shows also that to lose the finite model property, functionality in only one direction is sufficient. In fact, it is well known that *converse*-DPDL, which corresponds to a fragment of $\mathcal{ALCFI}_{reg}$, lacks the finite model property [Kozen and Tiuryn, 1990; Vardi and Wolper, 1986].

For all logics that lack the finite model property, reasoning with respect to unrestricted and finite models are fundamentally different tasks, and this needs to be taken explicitly into account when devising reasoning procedures. Restricting reasoning to finite domains is not common in knowledge representation. However, it is typically of interest in databases, where one assumes that the data available are always finite [Calvanese *et al.*, 1994; 1999e].

When reasoning w.r.t. finite models, some properties that are essential for the techniques developed for unrestricted model reasoning in expressive description logics fail. In particular, all reductions exploiting the tree model property (or similar properties that are based on "unraveling" structures) [Vardi, 1997] cannot be applied since this property does not hold when only finite models are considered. An intuitive justification can be given by observing that, whenever a (finite) model contains a cycle, the unraveling of such a model into a tree generates an infinite structure. Therefore alternative techniques have been developed.

In this section, we study decidability and computational complexity of finite model reasoning over TBoxes expressed in various sublanguages of $\mathcal{ALCQI}$. Specifically, by using techniques based on reductions to linear programming problems, we show that finite concept satisfiability w.r.t. to $\mathcal{ALUNI}$ TBoxes[1] constituted by inclusion axioms only is EXPTIME-complete [Calvanese *et al.*, 1994], and that finite model reasoning in arbitrary $\mathcal{ALCQI}$ TBoxes can be done in deterministic double exponential time [Calvanese, 1996a].

5.8.1 Finite model reasoning using linear inequalities

A procedure for finite model reasoning must specifically address the presence of number restrictions, since it is only in their presence that the finite model property

[1] $\mathcal{ALUNI}$ is the description logic obtained by extending $\mathcal{ALUN}$ (cf. Chapter 2) with inverse roles.

fails. We discuss a method which is indeed based on an encoding of number restrictions into linear inequalities, and which generalizes the one developed by Lenzerini and Nobili [1990] for the Entity-Relationship model with disjoint classes and relationships (hence without IS-A). We first describe the idea underlying the reasoning technique in a simplified case. In the next section we show how to apply the technique to various expressive description logics [Calvanese and Lenzerini, 1994b; 1994a; Calvanese *et al.*, 1994; Calvanese, 1996a].

Consider an $\mathcal{ALNI}$ TBox[1] $\mathcal{T}$ containing the following axioms: for each pair of distinct atomic concepts A and A', an axiom $\mathcal{A} \sqsubseteq \neg A'$, and for each atomic role P, an axiom of the form $\top \sqsubseteq \forall P.A_2 \sqcap \forall P^{-}.A_1$, for some atomic concepts A_1 and A_2 (not necessarily distinct). Such axioms enforce that in all models of $\mathcal{T}$ the following hold:

- P_1: The atomic concepts have pairwise disjoint extensions.
- P_2: Each role is "typed", which means that its domain is included in the extension of an atomic concept A_1, and its codomain is included in the extension of an atomic concept A_2.

Assume further that the only additional axioms in $\mathcal{T}$ are used to impose cardinality constraints on roles and inverse roles, and are of the form

$$\begin{aligned} \top &\sqsubseteq \geqslant m_1\, P \sqcap \leqslant n_1\, P \\ \top &\sqsubseteq \geqslant m_2\, P^{-} \sqcap \leqslant n_2\, P^{-} \end{aligned}$$

where m_1, n_1, m_2, and n_2 are positive integers with $m_1 \leq n_1$ and $m_2 \leq n_2$.

Due to the fact that properties P_1 and P_2 hold, the local conditions imposed by number restrictions on the number of successors of each individual, are reflected into global conditions on the total number of instances of atomic concepts and roles. Specifically, it is not difficult to see that, for a model $\mathcal{I}$ of such a TBox, and for each P, A_1, A_2, m_1, m_2, n_1, and n_2 as above, the cardinalities of $P^{\mathcal{I}}$, $A_1^{\mathcal{I}}$, and $A_2^{\mathcal{I}}$ must satisfy the following inequalities:

$$\begin{array}{rcccl} m_1 \cdot |A_1^{\mathcal{I}}| & \leq & |P^{\mathcal{I}}| & \leq & n_1 \cdot |A_1^{\mathcal{I}}| \\ m_2 \cdot |A_2^{\mathcal{I}}| & \leq & |P^{\mathcal{I}}| & \leq & n_2 \cdot |A_2^{\mathcal{I}}| \end{array}$$

On the other hand, consider the system $\Psi_{\mathcal{T}}$ of linear inequalities containing for each atomic role P typed by A_1 and A_2 the inequalities

$$\begin{array}{rcccl} m_1 \cdot \mathrm{Var}(A_1) & \leq & \mathrm{Var}(P) & \leq & n_1 \cdot \mathrm{Var}(A_1) \\ m_2 \cdot \mathrm{Var}(A_2) & \leq & \mathrm{Var}(P) & \leq & n_2 \cdot \mathrm{Var}(A_2) \end{array} \tag{5.1}$$

[1] $\mathcal{ALNI}$ is the description logic obtained by extending $\mathcal{ALN}$ (cf. Chapter 2) with inverse roles.

where we denote by $\mathrm{Var}(A)$ and $\mathrm{Var}(P)$ the unknowns, ranging over the non-negative integers, corresponding to the atomic concept A and the atomic role P respectively.

It can be shown that, if the only axioms in $\mathcal{T}$ are those mentioned above, then certain non-negative integer solutions of $\Psi_{\mathcal{T}}$ (called *acceptable* solutions) can be put into correspondence with finite models of $\mathcal{T}$. More precisely, for each acceptable solution $\mathcal{S}$, one can construct a model of $\mathcal{T}$ in which the cardinality of each concept or role X is equal to the value assigned by $\mathcal{S}$ to $\mathrm{Var}(X)$ [Lenzerini and Nobili, 1990; Calvanese *et al.*, 1994; Calvanese, 1996c]. Moreover, given $\Psi_{\mathcal{T}}$, it is possible to verify in time polynomial in its size, whether it admits an acceptable solution.

This property can be exploited to check finite satisfiability of an atomic concept A w.r.t. a TBox $\mathcal{T}$ as follows:

(i) Construct the system $\Psi_{\mathcal{T}}$ of inequalities corresponding to $\mathcal{T}$.
(ii) Add to $\Psi_{\mathcal{T}}$ the inequality $\mathrm{Var}(A) > 0$, which enforces that the solutions correspond to models in which the cardinality of the extension of A is positive.
(iii) Check whether $\Psi_{\mathcal{T}}$ admits an acceptable solution.

Observe that for simple TBoxes of the form described above, this method works in polynomial time, since (i) $\Psi_{\mathcal{T}}$ is of size polynomial in the size of $\mathcal{T}$, and can also be constructed in polynomial time, and (ii) checking the existence of acceptable solutions of $\Psi_{\mathcal{T}}$ can be done in time polynomial in the its size. Notice also that the applicability of the technique heavily relies on conditions P_1 and P_2, which ensure that, from an acceptable solution of $\Psi_{\mathcal{T}}$, a model of $\mathcal{T}$ can be constructed.

5.8.2 Finite model reasoning in expressive description logics

The method we have presented above is not directly applicable to more complex languages or TBoxes not respecting the particular form above. In order to extend it to more general cases we make use of the following observation: Linear inequalities capture global constraints on the total number of instances of concepts and roles. So we have to represent local constraints expressed by number restrictions by means of global constraints. This can be done only if P_1 and the following generalization of P_2 hold:

P_2': For each atomic role P and each concept expression C appearing in $\mathcal{T}$, the domain of P is either included in the extension of C or disjoint from it. Similarly for the codomain of P.

This condition guarantees that, in a model, all instances of a concept "behave" in the same way, and thus the local constraints represented by number restrictions are

indeed correctly captured by the global constraints represented by the system of inequalities.

It is possible to enforce conditions P_1 and P'_2 for expressive description logics, by first transforming the TBox, and then deriving the system of inequalities from the transformed version. We briefly sketch the technique to decide finite concept satisfiability in $\mathcal{ALUNI}$ TBoxes consisting of *specializations*, i.e., inclusion axioms in which the concept on the left hand side is atomic. A detailed account of the technique and an analysis of its computational complexity has been presented by Calvanese [1996c].

First of all, it is easy to see that, by introducing at most a linear number of new atomic concepts and TBox axioms, we can transform the TBox into an equivalent one in which the nesting of constructs is eliminated. Specifically, in such a TBox the concept on the right hand side of an inclusion axiom is of the form L, $L_1 \sqcup L_2$, $\forall R.L$, $\geqslant n\, R$, or $\leqslant n\, R$, where L is an atomic or negated atomic concept. For example, given the axiom

$$A \sqsubseteq C_1 \sqcup C_2$$

where C_1 and C_2 do not have the form above, we introduce two new atomic concepts A_{C_1} and A_{C_2}, and replace the axiom above by the following ones

$$\begin{array}{lcl} A & \sqsubseteq & A_{C_1} \sqcup A_{C_2} \\ A_{C_1} & \sqsubseteq & C_1 \\ A_{C_2} & \sqsubseteq & C_2 \end{array}$$

Then, to ensure that conditions P_1 and P'_2 are satisfied, we use instead of atomic concepts, sets of atomic concepts, called *compound concepts*[1] and instead of atomic roles, so called *compound roles*. Each compound role is a triple $(P, \widehat{C}_1, \widehat{C}_2)$ consisting of an atomic role P and two compound concepts $\widehat{C}_1$ and $\widehat{C}_2$. Intuitively, the instances of a compound concept $\widehat{C}$ are all those individuals of the domain that are instances of all concepts in $\widehat{C}$ and are not instances of any concept not in $\widehat{C}$. A compound role $(P, \widehat{C}_1, \widehat{C}_2)$ is interpreted as the restriction of role P to the pairs whose first component is an instance of $\widehat{C}_1$ and whose second component is an instance of $\widehat{C}_2$.

This ensures that two different compound concepts have necessarily disjoint extensions, and hence that the property corresponding to P_1 holds. The same observation holds for two different compound roles $(P, \widehat{C}_1, \widehat{C}_2)$ and $(P, \widehat{C}'_1, \widehat{C}'_2)$ that correspond to the same role P. Moreover, for compound roles, the property corresponding to property P_2 holds by definition, and, considering that the TBox contains only specializations and that nesting of constructs has been eliminated, also P'_2 holds.

[1] A similar technique, called *atomic decomposition* there, was used by Ohlbach and Koehler [1999].

We first consider the set $\mathcal{T}'$ of axioms in the TBox that do not involve number restrictions. Such axioms force certain compound concepts and compound roles to be *inconsistent*, i.e., have an empty extension in all interpretations that satisfy $\mathcal{T}'$. For example, the axiom $A_1 \sqsubseteq \neg A_2$ makes all compound concepts that contain both A_1 and A_2 inconsistent. Similarly, the axiom $A_1 \sqsubseteq \forall P.A_2$ makes all compound roles $(P, \widehat{C}_1, \widehat{C}_2)$ such that $\widehat{C}_1$ contains A_1 and $\widehat{C}_2$ does not contain A_2 inconsistent. Checking whether a given compound concept is inconsistent essentially amounts to evaluating a propositional formula in a given propositional model (the one corresponding to the compound concept), and hence can be done in time polynomial in the size of the TBox. Similarly, one can check in time polynomial in the size of the TBox whether a given compound role is inconsistent. Observe however, that since the total number of compound concepts and roles is exponential in the number of atomic concepts in the TBox, doing the check for all compound concepts and roles takes in general exponential time.

Once the consistent compound concepts and roles have been determined, we can introduce for each of them an unknown in the system of inequalities (the inconsistent compound concepts and roles are discarded). The axioms in the TBox involving number restrictions are taken into account by encoding them into suitable linear inequalities. Such inequalities are derived in a way similar to inequalities 5.1, except that now each inequality involves one unknown corresponding to a compound concept and a sum of unknowns corresponding to compound roles.

Then, to check finite satisfiability of an atomic concept A, we can add to the system the inequality

$$\sum_{\widehat{C} \subseteq 2^{\mathcal{A}} \mid A \in \widehat{C}} \mathrm{Var}(\widehat{C}) \geq 1$$

which forces the extension of A to be nonempty. Again, if the system admits an acceptable solution, then we can construct from such a solution a finite model of the TBox in which A is satisfied; if no such solution exists, then A is not finitely satisfiable. To check finite satisfiability of an arbitrary concept C, we can introduce a new concept name A, add to the TBox the axiom $A \sqsubseteq C$, and then check the satisfiability of A. Indeed, if A is finitely satisfiable, then so is C. Conversely, if the original TBox admits a finite model $\mathcal{I}$ in which C has a nonempty extension, then we can simply extend $\mathcal{I}$ to A by interpreting A as $C^{\mathcal{I}}$, thus obtaining a finite model of the TBox plus the additional axiom in which A is satisfied.

The system of inequalities can be effectively constructed in time exponential in the size of the TBox, and checking for the existence of acceptable solutions is polynomial in the size of the system [Calvanese *et al.*, 1994]. Moreover, since verifying concept satisfiability is already ExpTime-hard for TBoxes consisting of specializations only

and expressed in the much simpler language $\mathcal{ALU}$ [Calvanese, 1996b], the above method provides a computationally optimal reasoning procedure.

Theorem 5.25 *Finite concept satisfiability in $\mathcal{ALUNI}$ TBoxes consisting of specializations only is* ExpTime*-complete.*

The method can be extended to decide finite concept satisfiability also for a wider class of TBoxes, in which a negated atomic concept and, more in general, an arbitrary Boolean combination of atomic concepts may appear on the left hand side of axioms. In particular, this makes it possible to deal also with knowledge bases containing definitions of concepts that are Boolean combinations of atomic concepts, and reason on such knowledge bases in deterministic exponential time. Since $\mathcal{ALUNI}$ is not closed under negation, we cannot immediately reduce logical implication to concept satisfiability. However, the technique presented above can be adapted to decide in deterministic exponential time also finite logical implication in specific cases [Calvanese, 1996c].

A further extension of the above method can be used to decide logical implication in $\mathcal{ALCQI}$. The technique uses two successive transformations on the TBox, each of which introduces a worst case exponential blow up, and a final polynomial encoding into a system of linear inequalities [Calvanese, 1996c; 1996a].

Theorem 5.26 *Logical implication w.r.t. finite models in $\mathcal{ALCQI}$ can be decided in worst case deterministic double exponential time.*

For more expressive description logics, and in particular for all those description logics containing the construct for reflexive-transitive closure of roles, the decidability of finite model reasoning is still an open problem. Decidability of finite model reasoning for $\mathcal{C}^2$, i.e., first order logic with two variables and counting quantifiers (see also Chapter 4, Section 4.2) was shown recently [Grädel *et al.*, 1997b]. $\mathcal{C}^2$ is a logic that is strictly more expressive than $\mathcal{ALCQI}$ TBoxes, since it allows, for example, to impose cardinality restrictions on concepts [Baader *et al.*, 1996] or to use the full negation of a role. However, apart from decidability, no complexity bound is known for finite model reasoning in $\mathcal{C}^2$.

Techniques for finite model reasoning have also been studied in databases. In the relational model, the interaction between inclusion dependencies and functional dependencies causes the loss of the finite model property, and finite implication of dependencies under various assumptions has been investigated by Cosmadakis *et al.* [1990]. A method for finite model reasoning has been presented by Calvanese and Lenzerini [1994b; 1994a] in the context of a semantic and an object-oriented database model, respectively. The reasoning procedure, which represents a direct generalization of the one discussed above to relations of arbitrary arity, does not

exploit reification to handle relations (see Section 5.7) but encodes directly the constraints on them into a system of linear inequalities.

5.9 Undecidability results

Several additional description logic constructs besides those discussed in the previous sections have been proposed in the literature. In this section we present the most important of these extensions, discussing how they influence decidability, and what modifications to the reasoning procedures are needed to take them into account. In particular, we discuss Boolean constructs on roles, variants of role-value-maps or role agreements, and number restrictions on complex roles. Most of these constructs lead to undecidability of reasoning, if used in an unrestricted way. Roughly speaking, this is mainly due to the fact that the tree model property is lost [Vardi, 1997].

5.9.1 Boolean constructs on complex roles

In those description logics that include regular expressions over roles, such as $\mathcal{ALCQI}_{reg}$, since regular languages are closed under intersection and complementation, the intersection of roles and the complement of a role are already expressible, if we consider them applied to the set of role expressions. Here we consider the more common approach in PDLs, namely to regard Boolean operators as applied to the binary relations denoted by complex roles. The logics thus obtained are more expressive than traditional PDL [Harel, 1984] and reasoning is usually harder. We notice that the semantics immediately implies that intersection of roles can be expressed by means of union and complementation.

Satisfiability in PDL augmented with intersection of arbitrary programs is decidable in deterministic double exponential time [Danecki, 1984], and thus is satisfiability in $\mathcal{ALC}_{reg}$ augmented with intersection of complex roles, even though these logics have neither the tree nor the finite model property. On the other hand, satisfiability in PDL augmented with complementation of programs is undecidable [Harel, 1984], and so is reasoning in $\mathcal{ALC}_{reg}$ augmented with complementation of complex roles. Also, DPDL augmented with intersection of complex roles is highly undecidable [Harel, 1985; 1986], and since global functionality of roles (which corresponds to determinism of programs) can be expressed by means of local functionality, the undecidability carries over to $\mathcal{ALCF}_{reg}$ augmented with intersection of roles.

These proofs of undecidability make use of a general technique based on the reduction from the unbounded *tiling* (or *domino*) *problem* [Berger, 1966; Robinson, 1971], which is the problem of checking whether a quadrant of the integer plane can be tiled using a finite set of tile types—i.e., square tiles with a color on each

side—in such a way that adjacent tiles have the same color on the sides that touch[1]. We sketch the idea of the proof using the terminology of description logics, instead of that of PDLs. The reduction uses two roles right and up which are globally functional (i.e., $\leqslant 1\,\mathsf{right}$, $\leqslant 1\,\mathsf{up}$) and denote pairs of tiles that are adjacent in the x and y directions, respectively. By means of intersection of roles, right and up are constrained to effectively define a two-dimensional grid. This is achieved by imposing for each point of the grid (i.e., reachable through right and up) that by following $\mathsf{right} \circ \mathsf{up}$ one reaches a point reached also by following $\mathsf{up} \circ \mathsf{right}$:

$$\forall(\mathsf{right} \sqcup \mathsf{up})^*.\exists((\mathsf{right} \circ \mathsf{up}) \sqcap (\mathsf{up} \circ \mathsf{right}))$$

To enforce this condition, the use of intersection of compositions of atomic roles is essential. Reflexive-transitive closure (i.e., $\forall(\mathsf{right} \sqcup \mathsf{up})^*.C$) is then also exploited to impose the required constraints on all tiles of the grid. Observe that, in the above reduction, one can use TBox axioms instead of reflexive-transitive closure to enforce the necessary conditions in every point of the grid.

The question arises if decidability can be preserved if one restricts Boolean operations to basic roles, i.e., atomic roles and their inverse. This is indeed the case if complementation of basic roles is used only to express difference of roles, as demonstrated by the ExpTime decidability of $\mathcal{DLR}$ and its extensions, in which intersection and difference of relations are allowed (see Section 5.7).

5.9.2 Role-value-maps

Another construct, which stems from frame-systems, and which provides additional useful means to specify structural properties of concepts, is the so called *role-value-map* [Brachman and Schmolze, 1985], which comes in two forms: An *equality role-value-map*, denoted $R_1 = R_2$, represents the individuals o such that the set of individuals that are connected to o via role R_1 equals the set of individuals connected to o via role R_2. The second form of role-value-map is *containment role-value-map*, denoted $R_1 \subseteq R_2$, whose semantics is defined analogously, using set inclusion instead of set equality. Using these constructs, one can denote, for example, by means of $\mathsf{owns} \circ \mathsf{made\_in} \subseteq \mathsf{lives\_in}$ the set of all persons that own only products manufactured in the country they live in.

When role-value-maps are added, the logic loses the tree model property, and this construct leads immediately to undecidability of reasoning when applied to *role chains* (i.e., compositions of atomic roles). For $\mathcal{ALC}_{reg}$, this can be shown by a reduction from the tiling problem in a similar way as to what is done in [Harel, 1985] for DPDL with intersection of roles. In this case, the concept $\mathsf{right} \circ \mathsf{up} =$

[1] In fact the reduction is from the Π^1_1-complete—and thus highly undecidable—recurring tiling problem [Harel, 1986], where one additionally requires that a certain tile occurs infinitely often on the x-axis.

$\mathsf{up} \circ \mathsf{right}$ involving role-value-map can be used instead of role intersection to define the constraints on the grid. The proof is slightly more involved than that for DPDL, since one needs to take into account that the roles right and up are not functional (while in DPDL all programs/roles are functional). However, undecidability holds already for concept subsumption (with respect to an empty TBox) in $\mathcal{AL}$ (in fact $\mathcal{FL}^-$) augmented with role-value-maps, where the involved roles are compositions of atomic roles [Schmidt-Schauß, 1989]—see Chapter 3 for the details of the proof.

As for role intersection, in order to show undecidability, it is necessary to apply role-value-maps to compositions of roles. Indeed, if the application of role-value-maps is restricted to Boolean combinations of basic roles, it can be added to $\mathcal{ALCQI}_{reg}$ without influencing decidability and worst case complexity of reasoning. This follows directly from the decidability results for the extension with Boolean constructs on atomic and inverse atomic roles (captured by $\mathcal{DLR}$). Indeed, $R_1 \subseteq R_2$ is equivalent to $\forall(R_1 \sqcap \neg R_2).\bot$, and thus can be expressed using difference of roles. We observe also that *universal* and *existential role agreements* introduced in [Hanschke, 1992], which allow one to define concepts by posing various types of constraints that relate the sets of fillers of two roles, can be expressed by means of intersection and difference of roles. Thus reasoning in the presence of role agreements is decidable, provided these constructs are applied only to basic roles.

5.9.3 Number restrictions on complex roles

In $\mathcal{ALCFI}_{reg}$, the use of (qualified) number restrictions is restricted to atomic and inverse atomic roles, which guarantees that the logic has the tree model property. This property is lost, together with decidability, if functional restrictions may be imposed on arbitrary roles. The reduction to show undecidability is analogous to the one used for intersection of roles, except that now functionality of a complex role (i.e., $\leqslant 1\,(\mathsf{right} \circ \mathsf{up}) \sqcup (\mathsf{up} \circ \mathsf{right})$) is used instead of role intersection to define the grid.

An example of decidable logic that does not have the tree model property is obtained by allowing the use of role composition (but not transitive closure) inside number restrictions. Let us denote with $\mathcal{N}(X)$, where X is a subset of $\{\sqcup, \sqcap, \circ, ^-\}$, unqualified number restrictions on roles that are obtained by applying the role constructs in X to atomic roles. Let us denote with $\mathcal{ALCN}(X)$ the description logic obtained by extending $\mathcal{ALC}$ (cf. Chapter 2) with number restrictions in $\mathcal{N}(X)$. As shown by Baader and Sattler [1999], concept satisfiability is decidable for the logic $\mathcal{ALCN}(\circ)$, even when extended with number restrictions on union and intersection of role chains of the same length. Notice that, decidability for $\mathcal{ALCN}(\circ)$ holds only for reasoning on concept expressions and is lost if one considers reasoning with respect to a TBox (or alternatively adds transitive closure of roles) [Baader

and Sattler, 1999]. Reasoning even with respect to the empty TBox is undecidable if one adds to $\mathcal{ALCN}$ number restrictions on more complex roles. In particular, this holds for $\mathcal{ALCN}(\sqcap, \circ)$ (if no constraints on the lengths of the role chains are imposed) and for $\mathcal{ALCN}(\sqcup, \circ, ^-)$ [Baader and Sattler, 1999]. The reductions exploit again the tiling problem, and make use of number restrictions on complex roles to simulate a universal role that is used for imposing local conditions on all points of the grid.

Summing up we can state that the borderline between decidability and undecidability of reasoning in the presence of number restrictions on complex roles has been traced quite precisely, although there are still some open problems. E.g., it is not known whether concept satisfiability in $\mathcal{ALCN}(\sqcup, \circ)$ is decidable (although logical implication is undecidable) [Baader and Sattler, 1999].