CHAPTER 23

Reasoning in Expressive Description Logics

Diego Calvanese   Giuseppe De Giacomo

Daniele Nardi   Maurizio Lenzerini

SECOND READERS: Enrico Franconi, Ian Horrocks, Maarten de Rijke, and Ulrike Sattler.

Contents

HANDBOOK OF AUTOMATED REASONING
Edited by Alan Robinson and Andrei Voronkov

1. Introduction

Knowledge Representation is the field of Artificial Intelligence which focuses on the design of formalisms that are both epistemologically and computationally adequate for expressing the knowledge an agent has about a particular domain. One of the main research lines of the field has been concerned with the idea that the knowledge structure should be expressed in terms of the classes of objects that are of interest in the domain, as well as the relevant relationships holding among such classes. One of the most important relationships is the one holding between two classes when one class is a subset of the other. Based on such a relationship, the organization of the set of classes used to characterize a domain of interest is based on hierarchical structures which not only provides for an effective and compact representation of information, but also allows one to perform the basic reasoning tasks in a computationally effective way.

The above principle formed the basis for the development of the first frame systems and semantic networks. However, such systems were in general not formally defined and the associated reasoning tools were strongly dependent on the implementation strategies. A fundamental step towards a logic-based characterization of such systems has been accomplished through the work on the KL-ONE system [Brachman and Schmolze 1985], which collected many of the ideas stemming from earlier semantic networks and frame-based systems, and provided a logical basis for interpreting objects, classes (or concepts), and relationships (or links, or roles) between them [see for example Woods and Schmolze 1992]. One of the basic goals of such a logical reconstruction was the precise characterization of the set of constructs used to build class and link expressions.

Providing a formal meaning to the constructs of the representation language has been fundamental, but knowledge representation systems should also come with reasoning procedures that are sound and complete with respect to a specified formal semantics. This is required to give the user a clear understanding of the results of reasoning in terms of well-established notions such as logical consequence. In addition, one should also give a precise characterization of the computational behavior of the inference system; thus, this aspect also is to be addressed with formal tools. With the article "The tractability of subsumption in Frame-Based Description Languages" by Brachman and Levesque [1984], a research line addressing the tradeoff between the expressiveness of KL-ONE like languages and the computational complexity of reasoning was originated. In fact, it was shown that an apparently minor extension of the language could make the basic deduction problem in the language computationally hard (even undecidable).

There has been a number of changes in terminology used in the area; these reflect the predominant aspects on which the research has concentrated. The descendants of KL-ONE have first been grouped under the label *terminological systems*, to emphasize the fact that classes and relationships were used to establish the basic terminology adopted in the modeled domain. Later, the emphasis was on the sets of concept forming constructs admitted in the language, giving rise to the name *concept languages*. Recently, after attention has been further moved towards the properties of the underlying logical systems, the term *Description Logics* has become popular, and will be the one used in this chapter.

A *knowledge base* expressed in a Description Logic (DL) is constituted by two components, traditionally called *TBox* and *ABox* (originally from "Terminological Box" and "Assertional Box" respectively). The TBox stores a set of universally quantified assertions, stating general properties of concepts and roles. For example, an assertion of this kind is the one stating that a certain concept, say `Parent`, is defined as a given expression using other concepts and roles, say "`Person` with at least one `child`". The ABox comprises assertions on individual objects, called instance assertions. A typical assertion in the ABox is the one stating that an individual is an instance of a certain concept. For example, one can assert that `Bill` is an instance of "`Person` with at least one `child`".

Several reasoning tasks can be carried out on a knowledge base of the above kind. The simplest form of reasoning involves computing the subsumption relation between two concept expressions, i.e., verifying whether one expression always denotes a subset of the objects denoted by another expression. In the above example, one can easily derive that `Parent` is a specialization of `Person`, i.e., `Person` subsumes `Parent`. A more complex reasoning task consists in checking whether a certain assertion is logically implied by a knowledge base. For example, we can infer from the above assertions that `Bill` is an instance of `Parent`.

The above observations emphasize that a DL system is characterized by four aspects:

1. The set of constructs constituting the language for building the concepts and the roles used in the TBox and in the ABox.
2. The kind of assertions that may appear in the TBox.
3. The kind of assertions that may appear in the ABox.
4. The inference mechanisms provided for reasoning on the knowledge bases expressible in the system.

The expressive power and the deduction capabilities of a DL system depend on the various choices and assumptions that the system adopts with regard to the above aspects. As to the fourth aspect, we concentrate in this chapter on inference mechanisms that are sound and complete with respect to the standard Tarskian semantics (see Sect. 2), although other choices are possible [Patel-Schneider 1989, Baader and Hollunder 1995, Donini, Lenzerini, Nardi, Nutt and Schaerf 1992].

The first aspect has been the subject of a lot of research work in the last decade. Indeed, most of the results on the computational complexity of DLs have been devised in a simplified context where both the TBox and the ABox are empty [Nebel 1988, Schmidt-Schauß and Smolka 1991, Donini, Lenzerini, Nardi and Nutt 1991, Donini, Lenzerini, Nardi and Schaerf 1996, Donini, Lenzerini, Nardi and Nutt 1997]. This is not surprising, since these works aimed at studying the language constructs in isolation, with the goal of singling out their impact on the complexity of subsumption between concept expressions.

The third aspect has been addressed by a few papers dealing with logical implication of ABox assertions under the simplifying assumption of an empty

TBox, again with the goal of studying how the various language constructs influence the reasoning on individuals [Hollunder 1996, Donini, Lenzerini, Nardi and Schaerf 1994, Schaerf 1994]. The complete setting, i.e., reasoning with both the TBox and the ABox, has been the subject of some investigations only recently. For example, Buchheit, Donini and Schaerf [1993] and De Giacomo and Lenzerini [1996] study two DL systems with powerful languages for expressing both the TBox and the ABox.

The second aspect, which is the focus of the present chapter, has first been analyzed under several simplifying assumptions, such as:

- The assertions in the TBox are restricted to so-called definitions, where a definition is an assertion stating that the extension of a concept denoted by a name is equal to the extension of another concept (typically a complex concept).
- For every concept C, at most one definition for C appears in the TBox.
- Definitions are acyclic, i.e., if we build a graph whose nodes are atomic concepts and whose arcs connect pairs of concepts such that one appears in the definition of the other, then the graph is acyclic.

It is easy to see that, in principle, when the TBox does not contain cycles, the reasoning tasks can be turned into reasoning on the concept expressions obtained by unfolding the definitions, i.e., by substituting the concepts defined in the knowledge base with their definitions. Interestingly, Nebel [1990] has shown that reasoning on assertions is computationally hard, even under the assumption of acyclicity (coNP-complete in this case).

More recently, there has been a strong interest in the problem of reasoning with TBox assertions without the acyclicity assumption [Nebel 1991, Baader 1991, Schild 1994, De Giacomo and Lenzerini 1994*a*, Calvanese, De Giacomo and Lenzerini 1995, Horrocks 1998, Horrocks and Sattler 1999]. One important outcome of this line of research is that, limiting the expressive power of the language with the goal of gaining tractability is useless in this setting, because the power of TBox assertions alone generally leads to high complexity in the inference mechanisms even in simple languages (see Sect. 3.3. For this reason, these investigations often refer to very powerful languages for expressing concepts and roles, and the property of interest is no longer tractability of reasoning, but rather decidability [Buchheit et al. 1993, Calvanese 1996*c*, De Giacomo 1995]. In addition, in the presence of assertions with no restrictions on cycles, there are languages which happen to lack the finite model property and, consequently, one has to distinguish between reasoning on finite and infinite models.

The goal of the chapter is to provide a thorough introduction to the recent results on reasoning on TBoxes in expressive DLs, i.e., DLs where:

1. The language used for building concepts and roles comprise all classical concept forming constructs, inverse roles, and general forms of number restrictions.
2. No restriction is posed on the assertions in the TBox.

We observe that expressive DLs are important in the light of the renewed interest in DLs that we find in disparate application areas. Indeed, DL systems are now advocated as suitable knowledge representation systems in many contexts, such as Information Systems [Catarci and Lenzerini 1993, Calvanese, De Giacomo and Lenzerini 1998*b*], Databases [Borgida 1995, Calvanese, Lenzerini and Nardi 1994, Bergamaschi and Sartori 1992, Sheth, Gala and Navathe 1993, Ullman 1997, Calvanese, Lenzerini and Nardi 1999], Software Engineering [Devambu, Brachman, Selfridge and Ballard 1991, Calvanese, De Giacomo and Lenzerini 1999], Intelligent Access to the Network [Levy, Rajaraman and Ordille 1996, Blanco, Illarramendi and Goñi 1994], action representation [Artale and Franconi 1994], and Planning [Weida and Litman 1992]. Many of the above papers point out that the full capabilities of a DL system (expressive language, general TBox assertions) are often required in the corresponding application fields [see also Doyle and Patil 1991].

The chapter is organized as follows. Sect. 2 presents syntax and semantics of a DL admitting a very powerful set of constructs. Such a logic, called $\mathcal{ALCQI}$, will be used as a basis in the paper. Also, Sect. 2 introduces the basic reasoning tasks in DLs, and provides a general discussion on the various techniques proposed to carry out such tasks. Sect. 3 addresses the correspondence between Description Logics and Propositional Dynamic Logics, which is the basis for several technical results that are presented in Sect. 4, where the problem of reasoning in unrestricted models is addressed. Sect. 5 presents results and techniques for reasoning on finite models. Finally, Sect. 6 provides a more complete picture of the decidability/undecidability borderline in expressive DLs.

2. Description Logics

The basic elements of DLs are *concepts* and *roles*, which denote classes and binary relations, respectively. Arbitrary concept and role expressions (in the following simply called *concepts* and *roles*) are formed by starting from a set of *atomic concepts* and *atomic roles*, i.e., concepts and roles denoted simply by a name, and applying concept and role *constructs*. Each variant of DLs is characterized by the set of constructors that can be used. We name a DL using calligraphic letters, according to the convention of using a certain symbol (either a letter or a subscript) for a specific set of constructs. All DLs we deal with include a set of basic constructs (see later), which give the prefix $\mathcal{AL}$ to the name.

In the following, we present the syntax of all constructs that are considered in this paper, which correspond to a language called $\mathcal{ALCQI}$. For a comprehensive discussion on the constructs used in DLs, see [Woods and Schmolze 1992, De Giacomo 1995, Calvanese 1996*c*, Donini et al. 1996].

2.1. Syntax and Semantics of the Logic $\mathcal{ALCQI}$

We introduce now the DL $\mathcal{ALCQI}$, in which concepts and roles are formed according to the following syntax:

$$\begin{array}{rcl} C, C' & \longrightarrow & A \mid \neg C \mid C \sqcap C' \mid C \sqcup C' \mid \forall R.C \mid \exists R.C \mid \exists^{\geq n} R.C \mid \exists^{\leq n} R.C \\ R & \longrightarrow & P \mid P^- \end{array}$$

where, A and P denote atomic concepts and atomic roles respectively, C and R denote arbitrary concepts and roles (either direct or inverse roles), and n denotes a positive integer. We also use the following abbreviations to increase readability:

- $\bot$ for $A \sqcap \neg A$ (where A is any atomic concept),
- $\top$ for $A \sqcup \neg A$,
- $C \Rightarrow D$ for $\neg C \sqcup D$,
- $\exists^{=n} R.C$ for $\exists^{\geq n} R.C \sqcap \exists^{\leq n} R.C$.

Let us comment on the constructs of $\mathcal{ALCQI}$.

Among the constructs used in forming concept expressions we find the basic set operators, namely set complement, intersection, and union that are denoted as *negation* ($\neg$), *conjunction* ($\sqcap$), and *disjunction* ($\sqcup$), respectively. DLs admit a restricted form of quantification which is realized through so-called *quantified role restrictions*, that are composed by a quantifier (existential or universal), a role, and a concept expression. Quantified role restrictions allow one to represent the relationships existing between the objects in two concepts, and the forms considered in $\mathcal{ALCQI}$ are general enough to capture the most common ways of establishing such relationships. For example, one can characterize the set of objects all of whose children are male as $\forall$`child.Male`, as well as the set of objects that have at least one male child as $\exists$`child.Male`. The former construct is called *universal role restriction* while the latter is called *(qualified) existential role restriction*. The simplest existence condition on a role is $\exists R.\top$, which is often abbreviated by $\exists R$ and is called *unqualified existential*. We obtain the basic language $\mathcal{AL}$ from $\mathcal{ALCQI}$ by allowing only atomic roles, and by restricting the concept constructs to conjunction, negation of atomic concepts only, universal role restriction, and unqualified existential. Adding to $\mathcal{AL}$ general negation, denoted by the letter $\mathcal{C}$, gives the language $\mathcal{ALC}$ [Schmidt-Schauß and Smolka 1991], in which also disjunction and qualified existential role restriction can be expressed. The letter $\mathcal{U}$ indicates the presence of disjunction in a language where negation is restricted to atomic concepts (such as for example $\mathcal{AL}$).

Number restrictions are used to constrain the number of *fillers*, i.e., the objects that are in a certain relationship with a given object. For example, $\exists^{=2}$`child.Male` characterizes the set of parents with exactly two male children. The form used here, called *qualified number restriction* [Hollunder and Baader 1991], is a very general one. It allows one to pose restrictions on the number of objects connected through a certain role, counting only those objects that satisfy a certain condition. The most common form of number restriction does not place any restriction on the concept the role fillers belong to. This form is called *unqualified*, and is written $\exists^{\geq n} R$, which stands for $\exists^{\geq n} R.\top$ (similarly for $\exists^{\leq n} R$ and $\exists^{=n} R$). Observe that the special cases of number restrictions where the number involved is equal to "1", express functionality ($\exists^{\leq 1} R$) and existence constraints ($\exists^{\geq 1} R$, i.e., $\exists R$), respectively. The presence of qualified number restrictions is specified by the letter $\mathcal{Q}$ in the name of the language; unqualified number restrictions are indicated by the letter $\mathcal{N}$, and when the number can be only "1", the letter used is $\mathcal{F}$.

In addition to concept forming constructs, $\mathcal{ALCQI}$ provides the *inverse role* construct, which allows us to denote the inverse of a given relation. One can for example state with $\exists^{\leq 2}\texttt{child}^-$ that someone has at most two parents, by making use of the inverse of the role `child`. It is worth noticing, that in a language without the 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. The presence in the language of inverse roles is specified by the letter $\mathcal{I}$.

From the semantic point of view, concepts are interpreted as subsets of a domain, and roles as binary relations over that domain. An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ over a set $\mathcal{A}$ of atomic concepts and a set $\mathcal{P}$ of atomic roles consists of a nonempty set $\Delta^\mathcal{I}$ (the *domain* of $\mathcal{I}$) and a function $\cdot^\mathcal{I}$ (the *interpretation function* of $\mathcal{I}$) that maps every atomic concept $A \in \mathcal{A}$ to a subset $A^\mathcal{I}$ of $\Delta^\mathcal{I}$ (the set of *instances* of A) and every atomic role $P \in \mathcal{P}$ to a subset $P^\mathcal{I}$ of $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$ (the set of *instances* of P). The interpretation function can then be extended to arbitrary concepts and roles as follows[1]:

$$
\begin{array}{rcl}
(\neg C)^\mathcal{I} & = & \Delta^\mathcal{I} \setminus C^\mathcal{I} \\
(C \sqcap C')^\mathcal{I} & = & C^\mathcal{I} \cap C'^\mathcal{I} \\
(C_1 \sqcup C_2)^\mathcal{I} & = & C_1^\mathcal{I} \cup C_2^\mathcal{I} \\
(\forall R.C)^\mathcal{I} & = & \{o \in \Delta^\mathcal{I} \mid \forall o'.\, (o,o') \in R^\mathcal{I} \rightarrow o' \in C^\mathcal{I}\} \\
(\exists R.C)^\mathcal{I} & = & \{o \in \Delta^\mathcal{I} \mid \exists o'.\, (o,o') \in R^\mathcal{I} \wedge o' \in C^\mathcal{I}\} \\
(\exists^{\geq n} R.C)^\mathcal{I} & = & \{o \in \Delta^\mathcal{I} \mid \sharp\{o' \mid (o,o') \in Q^\mathcal{I} \wedge o' \in C^\mathcal{I}\} \geq n\} \\
(\exists^{\leq n} R.C)^\mathcal{I} & = & \{o \in \Delta^\mathcal{I} \mid \sharp\{o' \mid (o,o') \in Q^\mathcal{I} \wedge o' \in C^\mathcal{I}\} \leq n\} \\
(R^-)^\mathcal{I} & = & \{(o,o') \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid (o',o) \in R^\mathcal{I}\}
\end{array}
$$

The basic reasoning tasks on concept expressions are concept satisfiability and concept subsumption.

- *Concept satisfiability* is the problem of deciding whether a concept has a nonempty interpretation.
- *Concept subsumption* (between C_1 and C_2) is the problem of deciding whether $C_1^\mathcal{I} \subseteq C_2^\mathcal{I}$ holds in every interpretation.

In $\mathcal{ALCQI}$, and in any language closed under negation, concept satisfiability and concept subsumption are obviously related to each other. Namely, a concept C is satisfiable if and only if it is not subsumed by $\perp$, while C_1 is subsumed by C_2 if and only if $C_1 \sqcap \neg C_2$ is unsatisfiable.

2.2. Knowledge Bases in $\mathcal{ALCQI}$

Usually, in DLs, a *knowledge base* is formed by two components, a *TBox*, expressing intensional knowledge about classes and relations, and an *ABox*, expressing exten-

[1]We use $\sharp S$ to denote the cardinality of a set S.

sional knowledge about objects. Here we concentrate on intensional knowledge only, and therefore we identify a knowledge base with a TBox.

Formally, an $\mathcal{ALCQI}$ knowledge base is constituted by a finite set of *inclusion assertions* of the form

$$C_1 \sqsubseteq C_2$$

with C_1 and C_2 arbitrary concept expressions.

The semantics of a knowledge base is specified through the notion of satisfaction of assertions. An interpretation $\mathcal{I}$ *satisfies* the assertion $C_1 \sqsubseteq C_2$ if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$. An interpretation is a *model* of a knowledge base if it satisfies all assertions in it. A knowledge base is *satisfiable* if it admits a model.

Assertions of the form above are usually called *free* (or *general*) [Buchheit et al. 1993]. Special cases of assertions are also of interest. A *primitive inclusion assertion* is an inclusion assertion of the form $A \sqsubseteq C$, which specifies (by means of C) only *necessary* conditions for an object to be an instance of the atomic concept A. Thus, with only primitive inclusion assertions, an object cannot be inferred to be an instance of A, unless this is explicitly stated. Symmetrically, an assertion $C \sqsubseteq A$ specifies a *sufficient* condition for an object to be an instance of A. In contrast, an *equality assertion* $A \equiv C$, which corresponds to the pair of assertions $A \sqsubseteq C$ and $C \sqsubseteq A$, specifies both *necessary and sufficient* conditions for the instances of A. Observe that the inclusion assertion $A \sqsubseteq C$ is equivalent to the equality assertion $A \equiv A \sqcap C$, and that the inclusion assertion $C \sqsubseteq A$ is equivalent to $A \equiv A \sqcup C$. Equality assertions, are typical of the frame systems from which DLs originate, where assertions of this kind (without cycles, see later) are used to define a taxonomy of concepts.

For knowledge bases consisting only of primitive inclusion and/or equality assertions (and no free assertions), specific restrictions on the form of the assertions have been considered. For such knowledge bases it is usually assumed that each atomic concept may appear at most once on the left hand side of an assertion. Under this condition, allowing or not for the presence of so-called *(terminological) cycles*[2]becomes relevant. When cycles are not allowed, adding knowledge bases to a DL does not substantially change its properties. In particular, reasoning wrt a knowledge base can be straightforwardly reduced to concept subsumption. This is done by *unfolding*, i.e., by recursively replacing atomic concepts on the left hand side of a knowledge base assertion with the corresponding right hand side [Nebel 1991]. Instead, the presence of a cyclic knowledge base does have a strong impact on the DL. In this case, different types of semantics of a knowledge base may be defined, which differ in the interpretation of cycles (but coincide for acyclic knowledge bases). The semantics specified above is called *descriptive semantics* and is the only one that generalizes to free assertions. Alternatively, *fixpoint semantics* have been considered, in which the assertions are viewed as equations and only those interpretations that are (least or greatest) fixpoints of the equations are accepted as

[2]We remind the reader that a knowledge base contains a cycle if some concept in the right part of an assertion refers (either directly or indirectly through other assertions) to the atomic concept on the left part of the assertion.

models. For a detailed discussion on the different semantics, see [Nebel 1991, Buchheit et al. 1993, Buchheit, Donini, Nutt and Schaerf 1994, Schild 1994, De Giacomo and Lenzerini 1997]. Note that the presence of cycles increases also the computational complexity of reasoning [Baader 1996, Calvanese 1996*b*] and for this reason, until recently, it was ruled out in most knowledge representation systems based on DLs.

2.1. Example. The following $\mathcal{ALCQI}$ knowledge base $\mathcal{K}_{\mathit{file}}$ models a file-system constituted by file-system elements (`FSelem`), each of which is either a `Directory` or a `File`. Each `FSelem` has a unique name and a unique parent (`child`$^-$). A `Directory` may have children while a `File` may not, and `Root` is a special directory which has no parent.

$$
\begin{array}{rcll}
\texttt{FSelem} & \sqsubseteq & \exists^{\leq 1}\texttt{child}^-.\top & \\
\texttt{FSelem} & \sqsubseteq & \exists\texttt{name.String} \sqcap \exists^{\leq 1}\texttt{name}.\top & (2.1)\\
\texttt{FSelem} & \equiv & \texttt{Directory} \sqcup \texttt{File} & (2.2)\\
\texttt{Directory} & \sqsubseteq & \neg\texttt{File} & (2.3)\\
\texttt{Directory} & \sqsubseteq & \forall\texttt{child.FSelem} & (2.4)\\
\texttt{File} & \sqsubseteq & \forall\texttt{child}.\bot & \\
\texttt{Root} & \equiv & \texttt{Directory} \sqcap \forall\texttt{child}^-.\bot & (2.5)
\end{array}
$$

The assertions above are typical examples of data modeling constructs. In particular, assertions (2.2) and (2.3) state that `FSelem` is a *complete generalization* of `Directory` and `File`. Assertion (2.1) and assertion (2.4 represent a *has-a* constraint and a *has-many* constraint respectively, while assertion (2.5) provides a definition of the class `Root` in terms of a class expression, and implicitly contains an *is-a* constraint between the two classes `Root` and `Directory`.

In the following, we consider knowledge bases constituted by free assertions, hence without any restriction on their form. Knowledge bases of this form are in fact equivalent to knowledge bases constituted by primitive inclusion and equality assertions without restrictions. This follows easily by observing that a free assertion $C_1 \sqsubseteq C_2$ is equivalent to the pair of assertions $A \equiv C_1$, $A \sqsubseteq C_2$, where A is a newly introduced atomic concept. In Sect. 5 we deal also with so-called *primitive knowledge bases*, which are constituted only by primitive inclusion assertions. Such assertions correspond to the kind of constraints that can typically be expressed in conceptual data models [Calvanese et al. 1994], and reasoning on them is easier than with free assertions, if the underlying concept language is simple [Calvanese 1996*b*].

The constructs allowed in $\mathcal{ALCQI}$, in particular number restrictions and inverse roles, can interact in such a way that a knowledge base may admit no finite model, although it is satisfiable with an *infinite domain*. Similarly, a concept may be nonempty only in infinite interpretations [Cosmadakis, Kanellakis and Vardi 1990, Calvanese et al. 1994].

2.2. Example. Let $\mathcal{K}_{guard}$ be the following knowledge base

$$\begin{array}{rcl} \texttt{Guard} & \sqsubseteq & \exists\texttt{shields} \sqcap \forall\texttt{shields.Guard} \sqcap \exists^{\leq 1}\texttt{shields}^{-} \\ \texttt{FirstGuard} & \sqsubseteq & \texttt{Guard} \sqcap \forall\texttt{shields}^{-}.\bot \end{array}$$

Intuitively, the assertions in $\mathcal{K}_{guard}$ state that: a guard is someone who shields guards and is shielded by at most one individual; a first guard is a guard who is not shielded by anyone. It is easy to see that the existence of a first guard implies the existence of an infinite sequence of guards, each one shielding the following one. This is due to the fact that all guards (including the first one) must have a `shields` successor, the first guard has no `shields` predecessor while any other guard can have at most one `shields` predecessor. Hence no guard can be reused to form a cycle of guards that shield each other, and the only possibility is to have an infinite chain (or tree). Summing up we can say that `FirstGuard` is consistent if we allow interpretations with a domain of arbitrary cardinality, but becomes inconsistent if we consider only interpretations with a finite domain.

This example shows that $\mathcal{ALCQI}$ lacks the *finite model property* [Ebbinghaus and Flum 1999], and hence reasoning with respect to unrestricted and finite domains are different. This fact becomes important when different assumptions on the domain being modeled are made. Finite interpretations (and thus models) are typically of interest in Databases, while a finite domain assumption is usually not considered in Knowledge Representation, and needs to be taken explicitly into account when devising reasoning procedures [Calvanese et al. 1994, Calvanese 1996*a*]. Therefore, we distinguish between unrestricted and finite model reasoning.

The basic reasoning tasks with respect to a given knowledge base are the following:

- *Knowledge base satisfiability* is the problem of deciding whether a knowledge base $\mathcal{K}$ is *satisfiable*, i.e., whether $\mathcal{K}$ admits a model $\mathcal{I}$.
- *Concept consistency* is the problem of deciding whether a concept C is *consistent* in a knowledge base $\mathcal{K}$, i.e., whether $\mathcal{K}$ admits a model $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$.
- *Logical implication* is the problem of deciding whether a knowledge base $\mathcal{K}$ *implies* an inclusion assertion $C_1 \sqsubseteq C_2$ (written as $\mathcal{K} \models C_1 \sqsubseteq C_2$), i.e., whether $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ for each model $\mathcal{I}$ of $\mathcal{K}$.

Concept consistency and logical implication generalize concept satisfiability and concept subsumption, respectively, when we take into account a knowledge base.

For logics that do not have the finite model property, we distinguish between unrestricted and finite model reasoning. In particular, we talk about *unrestricted model reasoning*, when we consider arbitrary, and *finite model reasoning*, when we consider finite models only. When necessary, we distinguish between the two variants using the subscript "u" or "f".

2.3. Example (2.1 continued). The assertions in $\mathcal{K}_{file}$ imply that in a model every object connected by a chain of role `child` of length n (for some n) to an instance of Root is an instance of FSelem. Formally, $\mathcal{K}_{\mathit{file}} \models \exists(\mathtt{child}^-)^n.\mathtt{Root} \sqsubseteq \mathtt{FSelem}$.

The basic reasoning tasks above can be reduced to each other (provided the language over which the knowledge base is built is sufficiently expressive). We have that logical implication $\mathcal{K} \models C_1 \sqsubseteq C_2$ can be reformulated as inconsistency of $C_1 \sqcap \neg C_2$ in $\mathcal{K}$, while consistency of C in $\mathcal{K}$ can be reformulated as $\mathcal{K} \not\models C \sqsubseteq \bot$. In addition, consistency of C in $\mathcal{K}$ can be reformulated as satisfiability of the knowledge base $\mathcal{K} \cup \{\top \sqsubseteq \exists P_{new}.C\}$, where P_{new} is a newly introduced atomic role. Finally, satisfiability of a knowledge base K can be reformulated as consistency of $\top$ in $\mathcal{K}$. Since the basic reasoning services can be reduced to each other, we can talk generically about *knowledge base reasoning*.

2.3. Reasoning Techniques

The study of suitable techniques for solving the reasoning problems in Description Logics has been developed starting with severe restrictions on the expressiveness of the language and on the form of the knowledge base. Consequently, the reasoning techniques have evolved over time, from specialized, ad-hoc methods to fully general ones.

The first approaches were developed under the assumption that one can embody the knowledge represented in the terminology directly into concept expressions, rather than assertions. Therefore, subsumption on concept expressions was regarded as the basic reasoning task.

The basic idea of the first algorithms for subsumption between concept expressions was to transform two input concepts into labeled graphs and test whether one could be embedded into the other; the embedded graph would correspond to the more general concept (the subsumer) [Borgida and Patel-Schneider 1994]. This method is called *structural comparison*, and the relation between concepts it computes is called *structural subsumption*. However, a careful analysis of the algorithms for structural subsumption shows that they are sound, but not always complete with respect to the semantics, provided the language is sufficiently expressive.

The studies on the trade-off between the expressiveness of a representation language and the difficulty of reasoning on the representations built using that language [Levesque and Brachman 1987] lead to the idea of carefully analyzing the various constructs of DLs, with the goal of characterizing the computational complexity of the reasoning tasks. This kind of research pointed out the need of a general approach to reasoning in DLs. Schmidt-Schauß and Smolka [1991] propose the notion of *constraint system* as a general technique to meet this need. Subsequent investigations showed that constraint systems can be seen as specialized forms of tableaux. Many results on algorithms for reasoning on concept expressions, and their complexity were then derived using tableau-based techniques [Donini et al. 1991, Buchheit et al. 1993, Donini et al. 1996, Donini et al. 1997, Horrocks 1998, Horrocks and Sattler 1999]. Such techniques, besides being intuitively appealing, provided a useful framework for modularizing the problem of designing reasoning algorithms for languages formed by different sets on constructs. In fact, a tableau-based algorithm essentially amounts to providing an expansion rule for each of the constructs in the language, and then show the correctness of each rule and the termination of the expansion process. The algorithms for concept satisfiability and subsumption obtained in this way have also lead to actual implementations by application of clever control strategies and optimization techniques [Horrocks and Patel-Schneider 1999].

One of the problems of tableau-based techniques for reasoning on concept expressions is that they do not easily extend to reasoning with assertions. Buchheit et al. [1993] present a first attempt to extend the tableau-based approach to (cyclic) knowledge bases. While providing an interesting result, this work points out the difficulties that can arise in proving termination of tableau-based algorithms. Such difficulties, combined with the reports from the first implementations of these methods (see for example the comparison of implemented systems by Baader, Hollunder, Nebel, Profitlich and Franconi [1992]), have shifted the attention to other techniques for reasoning in expressive DLs. In particular, the correspondence between DLs and Propositional Dynamic Logics (described in Sect. 3) have motivated the research on reasoning techniques for expressive DLs that are based on the translation into reasoning problems in Propositional Dynamic Logics, and therefore rely on the associated automata-based methods [Vardi and Wolper 1984, Vardi 1985, Vardi and Wolper 1986]. Such an approach is exactly the one reported in this chapter, in particular in Sect. 4.

It is worth noticing that the tableau-based approach has recently led to interesting developments towards DLs of the expressive power considered in this paper [Baader and Sattler 2000]. In particular it has led to the implementation of systems for reasoning on (free) assertions [Horrocks and Sattler 1999, Horrocks, Sattler and Tobies 1999, Horrocks and Patel-Schneider 1999].

All the above observations apply basically to unrestricted reasoning. However, as we mentioned before, unrestricted reasoning and finite model reasoning differ in expressive DLs. In Sect. 5 we describe the basic ideas of the techniques for finite model reasoning in expressive DLs.

3. Description Logics and Propositional Dynamic Logics

Propositional Dynamic Logics (PDLs) have been introduced by Fischer and Ladner [1979] as a formal system for reasoning about computer programs and they have since been studied extensively and extended in several ways [Kozen and Tiuryn 1990]. In this section we provide a brief overview of PDLs, and of the correspondence between PDLs and DLs.

3.1. Syntax and Semantics of PDLs

Syntactically, a PDL is constituted by expressions of two sorts: *programs* and *formulas*. Programs and formulas are built by starting from a set *Prog* of *atomic* *programs* and a set *Prop* of *propositional letters* and applying suitable operators. We denote propositional letters with A, arbitrary formulas with ϕ, atomic programs with P, and arbitrary programs with r, all possibly with subscripts. We focus on Converse-PDL [Fischer and Ladner 1979] whose abstract syntax is as follows:

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

The basic propositional dynamic logic PDL [Fischer and Ladner 1979] is obtained from Converse-PDL by dropping converse programs r^-.

The semantics of Propositional Dynamic Logics [see for example Kozen and Tiuryn 1990] is based on the notion of a (Kripke) structure, which is defined as a triple $M = (\mathcal{S}, \{\mathcal{R}_P\}, \Pi)$, where $\mathcal{S}$ denotes a non-empty set of states, $\{\mathcal{R}_P\}$ is a family of binary relations over $\mathcal{S}$ such that each atomic program P is given a meaning through $\mathcal{R}_P$, and Π is a mapping from $\mathcal{S}$ to propositional letters such that $\Pi(s)$ determines the letters that are true in the state s. The basic semantical relation is "a formula ϕ holds at a state s of a structure M", which is written $M, s \models \phi$ and is defined by induction on the structure of ϕ:

$$\begin{array}{ll} M, s \models A & \text{iff } A \in \Pi(s) \\ M, s \models \top & \textit{always} \\ M, s \models \bot & \textit{never} \\ M, s \models \phi \wedge \phi' & \text{iff } M, s \models \phi \text{ and } M, s \models \phi' \\ M, s \models \phi \vee \phi' & \text{iff } M, s \models \phi \text{ or } M, s \models \phi' \\ M, s \models \phi \rightarrow \phi' & \text{iff } M, s \models \phi \text{ implies } M, s \models \phi' \\ M, s \models \neg\phi & \text{iff } M, s \not\models \phi \\ M, s \models \langle r\rangle\phi & \text{iff } \exists s' \colon (s, s') \in \mathcal{R}_r \text{ and } M, s' \models \phi \\ M, s \models [r]\phi & \text{iff } \forall s' \colon (s, s') \in \mathcal{R}_r \text{ implies } M, s' \models \phi \end{array}$$

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 structure of r:

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

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 PDL [Ben-Ari, Halpern and Pnueli 1982, Vardi and Wolper 1986, Goldblatt 1992].

A structure $M = (\mathcal{S}, \{\mathcal{R}_P\}, \Pi)$ is called a *model* of a formula ϕ if there exists a state $s \in \mathcal{S}$ such that $M, s \models \phi$. A formula ϕ is *satisfiable* if there exists a model of ϕ, otherwise the formula is unsatisfiable.

The following two theorems show that satisfiability is EXPTIME-complete for both PDL and Converse-PDL.

3.1. THEOREM ([Fischer and Ladner 1979]). *Satisfiability in PDL is EXPTIME-hard.*

3.2. THEOREM ([Pratt 1979, Vardi and Wolper 1986]). *Satisfiability in Converse-PDL and in Deterministic-Converse-PDL can be decided in deterministic exponential time.*

3.2. The Correspondence between DLs and PDLs

The correspondence between DLs and PDLs, first pointed out by Schild [1991], is based on the similarity between the interpretation structures of the two logics: at the extensional level, individuals (members of $\Delta^{\mathcal{I}}$) in DLs 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.

More precisely, Schild [1991] showed that Converse-PDL corresponds to the DL $\mathcal{ALCI}_{reg}$ obtained from $\mathcal{ALCQI}$ by dropping qualified number restrictions and adding the constructs to form regular expressions on roles:

$$R, R' \longrightarrow P \mid R^- \mid R \sqcup R' \mid R \circ R' \mid R^* \mid id(C)$$

corresponding directly to the PDLs constructs on programs.

Formally, the correspondence is realized through a (one-to-one and onto) mapping δ from $\mathcal{ALCI}_{reg}$ concepts to Converse-PDL formulas, and from $\mathcal{ALCI}_{reg}$ roles to Converse-PDL programs. The mapping δ is defined inductively as follows (we assume $\sqcup$ and $\Rightarrow$ to be expressed by means of $\sqcap$ and $\neg$):

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

Given the above correspondence, all the DLs constructs that we shall consider can naturally be mapped into their PDLs analogues. However, DLs are normally used to define a knowledge base, while in PDLs such a notion has not been considered explicitly [see Fischer and Ladner 1979]. Consequently, the above correspondence is not sufficient to relate the reasoning problems. Therefore, Schild [1991] extends the mapping in such a way that a knowledge base formed by a set of assertions can be viewed as a PDL formula, and reasoning with respect to the knowledge base can be rephrased in terms of reasoning on such a PDL formula. We address this problem in the next section by showing how the assertions in a knowledge base can be turned into a DL concept.

Notice that, although the correspondence between PDLs and DLs has been exploited to provide reasoning methods for DLs, it has also lead to a number of interesting extensions to PDLs in terms of those constructs that are typical of DLs and have never been considered in PDLs. In particular, there is a tight relation between qualified number restrictions and *graded modalities* in modal logic [Van der Hoek 1992, Van der Hoek and de Rijke 1995, Fattorosi-Barnaba and De Caro 1985, Fine 1972].

3.3. Internalization of the Knowledge Base

One of the main results of the correspondence between DLs and PDLs is the understanding that a knowledge base can be "internalized" into a single concept, i.e., it is possible to build a concept that expresses all the assertions of the knowledge base [Schild 1991, Baader 1991].

Below we show that for the DLs $\mathcal{ALC}\cdot_{reg}$, with "$\cdot$" standing for either nothing, $\mathcal{I}$, or additional constructs (e.g., number restrictions), reasoning on knowledge bases is in fact reducible to reasoning over concept expressions. We will see that such a reduction is possible because we consider DLs that are closed under negation and that are equipped with both union and reflexive-transitive closure of roles. Specifically, these constructs allow for denoting a "universal role", and this enables quantifying universally and existentially over all the objects in the interpretation domain, as shown below.

In the following we assume that $Q_1, \ldots, Q_m$ are all atomic roles and inverses of atomic roles present in the language, i.e., for logics without the inverse, $Q_1, \ldots, Q_m$ are all atomic roles, while for those equipped with inverse, $Q_1, \ldots, Q_m$ are all atomic roles and their inverse. Typically we assume that the atomic roles of the language are those occurring in the knowledge base and in the concepts under analysis.

Next we reformulate a standard result in modal logic, the so called generated submodel lemma [Goldblatt 1992], in terms of DLs, introducing the notion of *generated subinterpretation*.

Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation and $r \in \Delta^{\mathcal{I}}$ an object in $\Delta^{\mathcal{I}}$. We call *r-generated subinterpretation of $\mathcal{I}$* the interpretation $\mathcal{I}_r = (\Delta^{\mathcal{I}_r}, \cdot^{\mathcal{I}_r})$ defined as follows:

$$\begin{aligned}
\Delta^{\mathcal{I}_r} &= \{o \in \Delta^{\mathcal{I}} \mid (r, o) \in (\bigcup_{i=1,\ldots,m} Q_i^{\mathcal{I}})^*\} \\
A^{\mathcal{I}_r} &= A^{\mathcal{I}} \cap \Delta^{\mathcal{I}_r} \quad \text{for each atomic concept } A \\
P^{\mathcal{I}_r} &= P^{\mathcal{I}} \cap \Delta^{\mathcal{I}_r} \times \Delta^{\mathcal{I}_r} \quad \text{for each atomic role } P
\end{aligned}$$

3.3. Proposition. *For every $\mathcal{ALC}\cdot_{reg}$ concept C, for every interpretation $\mathcal{I}$, and for every object $r \in \Delta^{\mathcal{I}}$, $r \in C^{\mathcal{I}}$ iff $r \in C^{\mathcal{I}_r}$.*

Prop. 3.3 is a direct consequence of the fact that DL concepts describe properties of an object r only in terms of those objects linked to it through role chains. The proof is a straightforward variant of that of the generated submodel lemma by Goldblatt [1992]. (Cf. also with the notion of *bisimulation* described in [Clarke and Schlingloff 2001], Chapter 24 of this Handbook.)

The main consequence of Prop. 3.3 is that, without loss of generality, we may restrict our attention to a special class of interpretations that we call "rooted connected interpretations". An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a *rooted connected interpretation* if and only if, for some $r \in \Delta^{\mathcal{I}}$ called "root"

$$\Delta^{\mathcal{I}} \;=\; \{o \mid (r, o) \in (\bigcup_{i=1,\ldots,m} Q_i^{\mathcal{I}})^*\}.$$

Observe that if the DL we are considering is equipped with the inverse constructor then all $o \in \Delta^{\mathcal{I}}$ can be considered roots, while if it does not have inverse then there could be in fact only one root. Note also that generated subinterpretations are, by definition, rooted connected models. The following proposition ensures that we can restrict our attention to rooted connected interpretations.

3.4. Proposition. *Let $\mathcal{K}$ be an $\mathcal{ALC}\cdot_{reg}$ knowledge base and C, C' two $\mathcal{ALC}\cdot_{reg}$ concepts. Then the following holds:*

- Knowledge base satisfiability. *$\mathcal{K}$ is satisfiable iff for some rooted connected interpretation $\mathcal{I}$, $\mathcal{I}$ is a model of $\mathcal{K}$.*
- Concept consistency. *C is consistent in $\mathcal{K}$ iff for some rooted connected interpretation $\mathcal{I}$ and some $s \in \Delta^{\mathcal{I}}$, $\mathcal{I}$ is a model of $\mathcal{K}$ and $s \in C^{\mathcal{I}}$.*
- Logical implication. *$\mathcal{K} \models C \sqsubseteq C'$ iff every rooted connected interpretation $\mathcal{I}$ that is a model of $\mathcal{K}$ is also a model of $C \sqsubseteq C'$.*

Observe that in $\mathcal{ALC}\cdot_{reg}$, the role $U = (\bigsqcup_{i=1,\ldots,m} Q_i)^*$ is part of the language, and that such a role is interpreted as

$$(\bigcup_{i=1,\ldots,m} Q_i^{\mathcal{I}})^*.$$

This means that, at least with respect to rooted connected interpretations, we have in fact a *universal role*, i.e., a role that reaches from the root every object in the domain[3]. Hence $\forall U.C$ expresses that every object in the domain is an instance of C, and $\exists U.C$ expresses that there exists an object of the domain that is an instance of C. With the notion of universal role in place we can *internalize* the knowledge base, i.e., we can encode tasks involving the knowledge base into tasks not involving it.

[3]The universal role corresponds to a *master modality* in modal logic, which becomes the so-called universal modality when connected models are considered [Blackburn, de Rijke and Venema 2000].

3.5. Proposition. *Let $\mathcal{K} = \{C_1 \sqsubseteq D_1, \dots, C_n \sqsubseteq D_n\}$ be an $\mathcal{ALC}\cdot_{reg}$ knowledge base, C, C' two $\mathcal{ALC}\cdot_{reg}$ concepts, and let $C_\mathcal{K} = (C_1 \Rightarrow D_1) \sqcap \cdots \sqcap (C_n \Rightarrow D_n)$. Then the following holds:*

- Knowledge base satisfiability. *$\mathcal{K}$ is satisfiable iff $\forall U.C_\mathcal{K}$ is satisfiable.*
- Concept consistency. *C is consistent in $\mathcal{K}$ iff $C \sqcap \forall U.C_\mathcal{K}$ is satisfiable.*
- Logical implication. *$\mathcal{K} \models C \sqsubseteq D$ iff $(\forall U.C_\mathcal{K}) \Rightarrow (\forall U.(C \Rightarrow D))$ is satisfiable, or equivalently $(\forall U.C_\mathcal{K}) \sqcap (\exists U.(C \sqcap \neg D))$ is unsatisfiable, which can be further simplified to checking unsatisfiability of $(\forall U.C_\mathcal{K}) \sqcap (C \sqcap \neg D)$.*

An immediate consequence of Prop. 3.5 and of Th. 3.2 is the following.

3.6. Theorem. *Concept satisfiability and knowledge base satisfiability (and hence concept subsumption, concept consistency, and logical implication) in $\mathcal{ALCI}_{reg}$ are EXPTIME-complete.*

Observe that for DLs that do not include constructs for building regular expressions over roles, there is a sharp difference between reasoning techniques used in the presence of knowledge bases, and techniques used to reason on concept expressions. For example the logic $\mathcal{ALN}$ admits simple structural algorithms for deciding reasoning tasks not involving assertions that are sound and complete and require polynomial time [Borgida and Patel-Schneider 1994]. However, if free assertions are considered then decision procedures developed involve suitable termination strategies [Donini et al. 1996]. This difference is reflected by the computational properties of the associated decision problems. In particular, for the logic $\mathcal{ALC}$, reasoning tasks not involving knowledge bases are PSPACE-complete, while for knowledge base reasoning the following hardness result holds.

3.7. Theorem. *Concept consistency (and hence knowledge base satisfiability and logical implication) in $\mathcal{ALC}$ is EXPTIME-hard.*

This result follows directly from the proof of Thm. 3.1, by observing that the PDL formula used in the reduction has the form $\phi \wedge [P^*]\phi'$, where the only program appearing in ϕ and ϕ' is the atomic program P. By considering Prop. 3.5, satisfiability of such a formula corresponds to concept consistency in $\mathcal{ALC}$.

4. Unrestricted Model Reasoning

In this section we discuss the problem of reasoning with respect to unrestricted models in knowledge bases expressed in $\mathcal{ALCQI}$. We show that reasoning in $\mathcal{ALCQI}$ knowledge bases can be "compiled" into satisfiability in standard PDL. In order to do so we consider $\mathcal{ALC}\cdot$, with "$\cdot$" standing for either nothing, $\mathcal{F}$, $\mathcal{I}$, or $\mathcal{FI}$, augmented by the reflexive-transitive closure operator on atomic and inverse roles (if present in the logic), with the proviso that such a role construct cannot appear in functional restrictions. In other words, functional restrictions (if present in the logic) are only applied to atomic roles or their inverse. We denote such logics by $\mathcal{ALC}\cdot_*$.

Without loss of generality we concentrate on knowledge base satisfiability. We provide a cascade of encodings[4] starting from knowledge base satisfiability in $\mathcal{ALCQI}$ and arriving to knowledge base satisfiability in $\mathcal{ALC}_*$. Since $\mathcal{ALC}_*$ is a sublanguage of $\mathcal{ALC}_{reg}$, which in turn corresponds to standard PDL, by internalizing the knowledge base we get to satisfiability in standard PDL.

The cascade of encodings of knowledge base satisfiability problems is as follows:

$$\mathcal{ALCQI} \longrightarrow \mathcal{ALCFI}_* \longrightarrow \mathcal{ALCI}_* \longrightarrow \mathcal{ALC}_*$$

More precisely, we first eliminate qualified number restrictions in favor of (unqualified) functional restrictions, which are the simplest form of number restriction considered in DLs –in doing this we need to introduce the reflexive-transitive closure of atomic and inverse roles; second, we eliminate completely functional restrictions; third, we eliminate inverse roles thus arriving to $\mathcal{ALC}_*$. Notably in encoding $\mathcal{ALCFI}_*$ knowledge bases into $\mathcal{ALCI}_*$ knowledge bases, we go from a logic that does not have the finite model property to a logic that has it.

The logics $\mathcal{ALC}_*$ and $\mathcal{ALCI}_*$ correspond to a restricted form of standard PDL and Converse-PDL respectively, which are both well known. We encode $\mathcal{ALCI}_*$ knowledge bases into $\mathcal{ALC}_*$ knowledge bases mainly for two reasons. First, the conceptual simplicity of encoding allows us to explain a general encoding technique in simple terms – the same encoding technique is also used to encode $\mathcal{ALCFI}_*$ knowledge bases into $\mathcal{ALCI}_*$ knowledge bases, although its application is much more subtle in that case. Second, since, as mentioned, $\mathcal{ALC}_*$ is a sublanguage of $\mathcal{ALC}_{reg}$, we can apply internalization and transform knowledge base reasoning in $\mathcal{ALC}_*$ into satisfiability in standard PDL, for which decision procedures can be effectively implemented (e.g., [Pratt 1980, De Giacomo and Massacci 2000] based on tableaux).

Each of the above encodings is polynomial[5]. Hence the overall compilation is polynomial as well. This allows us to conclude that knowledge base satisfiability (and hence concept consistency and logical implication) in $\mathcal{ALCQI}$ is EXPTIME-decidable, and hence EXPTIME-complete since knowledge base satisfiability in $\mathcal{ALC}$ is already EXPTIME-hard. Moreover such a cascade of encodings provides an algorithm to reason in $\mathcal{ALCQI}$ knowledge bases: first compile the $\mathcal{ALCQI}$ knowledge base into a $\mathcal{ALC}_*$ knowledge base (a polynomial step); then by internalization transform it into a standard PDL satisfiability problem (again a polynomial step); then run a PDL decision procedure (a possibly exponential step).

The rest of the section is organized as follows. First we present the notion of Fischer-Ladner closure of a knowledge base that is used in formulating the encodings. Then we present the encoding of knowledge bases, in order, from $\mathcal{ALCI}_*$ to $\mathcal{ALC}_*$, from $\mathcal{ALCFI}_*$ to $\mathcal{ALCI}_*$, and finally from $\mathcal{ALCQI}$ to $\mathcal{ALCFI}_*$.

[4] On encoding non-classical logics in classical logic see also [Ohlbach et al. 2001] (Chapter 21 of this Handbook) and on encoding modal logics in other modal logics see [Kracht and Wolter 2000].

[5] With the assumption, which is standard in DLs, that number restrictions are encoded in unary. Numbers encoded in binary are considered e.g., in [Tobies 1999*b*, Tobies 1999*a*].

4.1. Fisher-Ladner Closure of a Knowledge Base

We assume, without loss of generality, $\sqcup, \forall$ to be expressed by means of $\neg, \sqcap, \exists$. The *Fisher-Ladner closure* of an $\mathcal{ALCI}_*$ knowledge base $\mathcal{K}$, denoted $CL(\mathcal{K})$, is the least set of concepts F such that:

$$
\begin{array}{llll}
\text{if} & C \sqsubseteq C' \in \mathcal{K} & \text{then} & C, C' \in F \\
\text{if} & C \sqcap C' \in F & \text{then} & C, C' \in F \\
\text{if} & \neg C \in F & \text{then} & C \in F \\
\text{if} & C \in F & \text{then} & \neg C \in F \quad (\text{if } C \text{ is not of the form } \neg C') \\
\text{if} & \exists R.C \in F & \text{then} & C \in F \\
\text{if} & \exists R^*.C \in F & \text{then} & C, \exists R.\exists R^*.C \in F
\end{array}
$$

The notion of Fisher-Ladner closure of a knowledge base is closely related to the notion of Fisher-Ladner closure of a PDL formula [Fischer and Ladner 1979], which in turn is closely related to that of set of subformulas in simpler modal logics. Intuitively, given a knowledge base $\mathcal{K}$, $CL(\mathcal{K})$ includes all the concepts that play some role in establishing the truth-value of the concepts occurring in $\mathcal{K}$. In other words, $CL(\mathcal{K})$ includes all the concepts occurring in $\mathcal{K}$ and those generated by unfolding each concept of the form $\exists R^*.C$ into $C \sqcup \exists R.\exists R^*.C$. Both the number and the size of the concepts in $CL(\mathcal{K})$ are linearly bounded by the size of $\mathcal{K}$.

4.2. Inverse Roles

We show now that it is possible to eliminate the "inverse" operator from $\mathcal{ALCI}_*$ knowledge bases, while preserving the soundness and completeness of inference. Specifically, we present a polynomial encoding of $\mathcal{ALCI}_*$ knowledge bases into $\mathcal{ALC}_*$ knowledge bases that eliminates inverse roles but adds enough information so as not to destroy the original meaning of concepts with respect to the reasoning tasks. Such an encoding, first presented by De Giacomo [1996] in the context of PDLs, is the simplest illustration of a general technique for deriving reasoning procedures for expressive logics based on a (possibly polynomial) encoding of logics into simpler ones. Such a technique has led to several decidability and complexity results, as well as reasoning procedures in DLs. In particular, in the Sect. 4.3 it is used to eliminate functional restrictions.

Intuitively, the technique is based on two main points. Let the "Source Logic" be *SL* and the "Target Logic" be *TL* (in this section these logics are $\mathcal{ALCI}_*$ and $\mathcal{ALC}_*$ respectively):

1. Identify a finite set of assertion schemas in the language of *TL* capturing those characteristics that distinguish *SL* from *TL* (in the present case such assertion schemas are of the form $C \sqsubseteq \forall P.(\exists P^c.C)$, $C \sqsubseteq \forall P^c.(\exists P.C)$, and force the binary relation interpreting P^c to be the inverse of the one interpreting P).

2. Devise a function that, given an *SL* knowledge base $\mathcal{K}$, returns a finite "closed"[6] set of *SL* concepts, whose interpretation uniquely determines that of the concepts in $\mathcal{K}$, and that is used to instantiate the assertion schemas in (1) (in the present case such a set is simply the Fisher-Ladner closure of $\mathcal{K}$).

Indeed, by instantiating the assertion schemas in (1) to the concepts in (2), we can derive a *TL* knowledge base (in the present case, the so called $\mathcal{ALC}_*$-counterpart of an $\mathcal{ALCI}_*$ knowledge base, see below) which corresponds to the original *SL* knowledge base, in the sense that it preserves knowledge base satisfiability, concept consistency, and logical implication. If both the cardinality of the sets in (1) and (2) and the size of their elements are polynomially bounded by the original knowledge base, then so is the knowledge base we get.

We define the encoding ζ from $\mathcal{ALCI}_*$ knowledge bases $\mathcal{K}$ to $\mathcal{ALC}_*$ knowledge bases $\zeta(\mathcal{K})$, such that $\mathcal{K}$ is satisfiable if and only if $\zeta(\mathcal{K})$ is satisfiable. The knowledge base $\zeta(\mathcal{K})$, whose size is polynomial in the size of $\mathcal{K}$, is called the *$\mathcal{ALC}_*$-counterpart of K*.

4.1. Definition. Let $\mathcal{K}$ be an $\mathcal{ALCI}_*$ knowledge base. The *$\mathcal{ALC}_*$-counterpart* $\zeta(\mathcal{K})$ of $\mathcal{K}$ is the union of two knowledge bases, $\zeta(\mathcal{K}) = \zeta_1(\mathcal{K}) \cup \zeta_2(\mathcal{K})$, where:

- $\zeta_1(\mathcal{K})$ is obtained from the original knowledge base $\mathcal{K}$ by replacing each occurrence of P^- with a new atomic role P^c, for every atomic role P occurring in $\mathcal{K}$.
- $\zeta_2(\mathcal{K})$ constitutes of a pair of assertions

$$\begin{aligned} C &\sqsubseteq \forall P.\exists P^c.C \\ C &\sqsubseteq \forall P^c.\exists P.C \end{aligned}$$

  for every $C \in CL(\zeta_1(\mathcal{K}))$ and every atomic program P occurring in $\mathcal{K}$.

In $\zeta_1(\mathcal{K})$ the inverse of atomic roles in $\mathcal{K}$ is replaced with new atomic roles. Each new role P^c is intended to represent P^- in $\zeta_1(\mathcal{K})$. $\zeta_2(\mathcal{K})$ constrains the models $\mathcal{I}$ of $\zeta(\mathcal{K})$ so that, for all $C \in CL(\zeta_1(\mathcal{K}))$, for all objects $o \in \Delta^{\mathcal{I}}$, if o is an instance of C then all the P-successors of o have a P^c-successor which is an instance of C as well, and similarly all the P^c-successors of o have a P-successor which is an instance of C. We show that, as far as knowledge base satisfiability is concerned (and hence concept consistency and logical implication), this allows us to correctly represent the inverse of P by means of P^c.

First of all, observe that if, instead of $\zeta_2(\mathcal{K})$, we impose the two *assertion schemas*:

$$\begin{aligned} X &\sqsubseteq \forall P.\exists P^c.X \\ X &\sqsubseteq \forall P^c.\exists P.X \end{aligned}$$

where X is to be replaced by *every* concept of the language defined by the atomic concepts and roles in $\zeta_1(\mathcal{K})$, then the models of $\zeta_1(\mathcal{K})$ would be isomorphic to

[6] That is, the interpretation of each concept in the set depends only on the interpretation of the concepts already in the set.

the models of $\mathcal{K}$. In fact, the above two assertion schemas correspond to the ones used in the axiomatization of Converse-PDL to force each program P^- to be the converse of the program P. Hence the resulting logic would not be $\mathcal{ALC}_*$ but $\mathcal{ALCI}_*$ (where P^c instead of P^- denotes inverse programs). $\zeta_2(\mathcal{K})$ can be thought of as a *finite instantiation* of the above two schemas, with one instance for each concept in $CL(\zeta_1(\mathcal{K}))$. Although imposing the validity of such a finite instantiation does not suffice to guarantee the isomorphism of the models of $\zeta_1(\mathcal{K})$ and $\mathcal{K}$, it suffices to guarantee that $\mathcal{K}$ has a model if and only if $\zeta(\mathcal{K})$ has one.

To prove this result we define an operation that, given a model $\mathcal{I}$ of $\zeta(\mathcal{K})$, transforms it into a new special model $\mathcal{I}'$ of $\zeta(\mathcal{K})$, called c-closure of $\mathcal{I}$, which is isomorphic to a model of $\mathcal{K}$.

Let $\tilde{P}$ be an abstraction for both P and P^c, such that $\tilde{P}^c$ denotes P^c if $\tilde{P} = P$, and $\tilde{P}^c$ denotes P if $\tilde{P} = P^c$. Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a model of $\zeta(\mathcal{K})$. We call the *c-closure* of $\mathcal{I}$, the interpretation $\mathcal{I}' = (\Delta^{\mathcal{I}'}, \cdot^{\mathcal{I}'})$, defined as follows:

- $\Delta^{\mathcal{I}'} = \Delta^{\mathcal{I}}$;
- $A^{\mathcal{I}'} = A^{\mathcal{I}}$, for each atomic concept A;
- $\tilde{P}^{\mathcal{I}'} = \tilde{P}^{\mathcal{I}} \cup \{(o', o) \mid (o, o') \in (\tilde{P}^c)^{\mathcal{I}}\}$, for each atomic role $\tilde{P}$.

Note that in the c-closure $\mathcal{I}'$ of a model $\mathcal{I}$, $P^{\mathcal{I}'}$ is obtained from $P^{\mathcal{I}}$ by including, for each pair (o, o') in $(P^c)^{\mathcal{I}}$, the pair (o', o) in $P^{\mathcal{I}'}$, and similarly $(P^c)^{\mathcal{I}'}$ is obtained from $(P^c)^{\mathcal{I}}$ by including, for each pair (o, o') in $P^{\mathcal{I}}$, the pair (o', o) in $(P^c)^{\mathcal{I}'}$. As a consequence, in the c-closure of an interpretation each atomic role P^c is interpreted as the inverse of P.

The next lemma relates models of $\zeta(\mathcal{K})$ to models of $\mathcal{K}$.

4.2. Lemma. *Let $\mathcal{K}$ be an $\mathcal{ALCI}_*$ knowledge base, $\zeta(\mathcal{K})$ its $\mathcal{ALC}_*$-counterpart, and $\mathcal{I}$ a model of $\zeta(\mathcal{K})$. Then the c-closure $\mathcal{I}'$ of $\mathcal{I}$ is a model of $\mathcal{K}$.*

From this lemma, and by observing that every model of $\mathcal{K}$ can be trivially transformed into a model of $\zeta(\mathcal{K})$ by interpreting P^c as P^-, we get the following result.

4.3. Theorem. *An $\mathcal{ALCI}_*$ knowledge base $\mathcal{K}$ is satisfiable if and only if its $\mathcal{ALC}_*$-counterpart $\zeta(\mathcal{K})$. is satisfiable.*

As a corollary of this theorem we get that knowledge base satisfiability (and hence concept consistency and logical implication) in $\mathcal{ALCI}_*$ and in $\mathcal{ALC}_*$ have the same computational complexity: they are both EXPTIME-complete. Observe however that such a result also follows directly from observing that reasoning with $\mathcal{ALCI}_*$ knowledge bases corresponds to reasoning with a restricted form of Converse-PDL and similarly, reasoning with $\mathcal{ALC}_*$ knowledge bases corresponds to reasoning with a restricted form of standard PDL.

4.3. Functional Restrictions

We now study reasoning in $\mathcal{ALCFI}_*$ knowledge bases. $\mathcal{ALCFI}_*$ is the DL obtained from $\mathcal{ALCI}_*$ by adding functional restrictions, i.e., number restrictions of the form

$\exists^{\leq 1} R$, where R is either an atomic role or the inverse of an atomic role. We show that knowledge base satisfiability (and hence concept consistency and logical implication) in $\mathcal{ALCFI}_*$ is EXPTIME-decidable, by following the technique already introduced above, and exhibiting an encoding of $\mathcal{ALCFI}_*$ knowledge bases into $\mathcal{ALCI}_*$ knowledge bases [De Giacomo and Lenzerini 1994*a*, De Giacomo 1995]. Although such an encoding has a simple form, proving its correctness requires quite sophisticated manipulations on interpretations. In particular, we observe that $\mathcal{ALCFI}_*$ knowledge bases *do not have* the finite model property, while $\mathcal{ALCI}_*$ knowledge bases *do have* it. Hence filtration arguments, usual in modal logics, cannot be applied directly.

We formally define the encoding γ from $\mathcal{ALCFI}_*$ knowledge bases $\mathcal{K}$ to $\mathcal{ALCI}_*$ knowledge bases $\gamma(\mathcal{K})$, such that $\mathcal{K}$ satisfiable if and only if $\gamma(\mathcal{K})$ is satisfiable. The size of $\gamma(\mathcal{K})$ is polynomial with respect to the size of $\mathcal{K}$. Since knowledge base satisfiability in $\mathcal{ALCI}_*$ can be decided in EXPTIME, this ensures that knowledge base satisfiability (and hence concept consistency and logical implication) in $\mathcal{ALCFI}_*$ can be decided in EXPTIME too.

We assume, without loss of generality, that $\mathcal{K}$ is in negation normal form (i.e., negations in concepts occurring in $\mathcal{K}$ are pushed inside as much as possible). It is easy to check that the transformation of any concept into negation normal form can be performed in linear time in the size of the concept.

4.4. Definition. Let $\mathcal{K}$ be an $\mathcal{ALCFI}_*$ knowledge base in negation normal form. The *$\mathcal{ALCI}_*$-counterpart* $\gamma(\mathcal{K})$ of $\mathcal{K}$ is the union of two knowledge base, $\gamma(\mathcal{K}) = \gamma_1(\mathcal{K}) \cup \gamma_2(\mathcal{K})$, where:

- $\gamma_1(\mathcal{K})$ is obtained from the original knowledge base $\mathcal{K}$ by replacing each $\exists^{\leq 1} R$ with a new atomic concept $A_{\exists^{\leq 1} R}$, and each $\neg\exists^{\leq 1} R$ with $(\exists R.H_{\exists^{\leq 1} R}) \sqcap (\exists R.\neg H_{\exists^{\leq 1} R})$, where $H_{\exists^{\leq 1} R}$ is again a new atomic concept[7].
- $\gamma_2(\mathcal{K})$ constitutes of one assertion

$$(A_{\exists^{\leq 1} R} \sqcap \exists R.C) \sqsubseteq \forall R.C$$

for every $A_{\exists^{\leq 1} R}$ occurring in $\gamma_1(\mathcal{K})$ and every $C \in CL(\gamma_1(\mathcal{K}))$, with the proviso that every concept of the form $\neg A_{\exists^{\leq 1} R}$ is replaced by $(\exists R.H_{\exists^{\leq 1} R}) \sqcap (\exists R.\neg H_{\exists^{\leq 1} R})$.

$\gamma_1(\mathcal{K})$ introduces the new concepts $A_{\exists^{\leq 1} R}$ and $H_{\exists^{\leq 1} R}$ in place of $\exists^{\leq 1} R$, so that positive occurrences of $\exists^{\leq 1} R$ are represented by the concept $A_{\exists^{\leq 1} R}$, and negative occurrences are represented by $(\exists R.H_{\exists^{\leq 1} R}) \sqcap (\exists R.\neg H_{\exists^{\leq 1} R})$. Note that every instance of $(\exists R.H_{\exists^{\leq 1} R}) \sqcap (\exists R.\neg H_{\exists^{\leq 1} R})$ has at least two R-successors.

The purpose of $\gamma_2(\mathcal{K})$ is less obvious. Intuitively, it constrains the models $\mathcal{I}$ of $\gamma(\mathcal{K})$ so that: for every object $o \in \Delta^{\mathcal{I}}$, if o is an instance of $A_{\exists^{\leq 1} R}$ and o_1 and o_2 are two R-successors of o, then o_1 and o_2 are instances of exactly the same concepts of $CL(\gamma_1(\mathcal{K}))$. This condition is sufficient to build a new model in which o_1 and o_2 are merged into a single object.

[7] We recall that only atomic roles or inverse of atomic roles can appear inside functional restrictions.

Observe that if, instead of adding $\gamma_2(\mathcal{K})$, we impose the assertion schema

$$A_{\exists^{\leq 1}R} \sqcap \exists R.X \sqsubseteq \forall R.X$$

where X is to be replaced by every concept of the language defined by the atomic concepts and roles in $\gamma_1(\mathcal{K})$, then the models of $\gamma_1(C)$ would be isomorphic to models of the original $\mathcal{ALCFI}_*$ knowledge base. However, the problem of verifying whether an $\mathcal{ALCI}_*$ concept is logically implied by an assertion schema is in general undecidable [Kozen and Tiuryn 1990]. So, adding the above schema to $\mathcal{ALCI}_*$ is of no use in establishing the decidability of reasoning in $\mathcal{ALCFI}_*$ knowledge bases.

Instead, the knowledge base $\gamma_2(\mathcal{K})$ can be thought of as a finite instantiation of the schema above, with one instance for each concept in $CL(\gamma_1(\mathcal{K})))$. Intuitively, imposing the validity of such finite instantiation is sufficient to guarantee that if $\gamma(\mathcal{K})$ has a model then $\mathcal{K}$ has a model as well, and vice-versa.

The main part in showing this result is showing that, given a model of $\gamma(\mathcal{K})$ we can build a model of $\mathcal{K}$. We proceed in two main steps:

1. Given a model $\mathcal{I}$ of $\gamma(\mathcal{K})$, we construct a special tree-like model $\mathcal{I}^t$.
2. Then, by suitably modifying $\mathcal{I}^t$, we construct a new model $\mathcal{I}^f$, in which all functional restriction requirements are satisfied, i.e., every object in which $A_{\exists^{\leq 1}R}$ holds has at most one R-successor.

The model of $\gamma(\mathcal{K})$ thus obtained is isomorphic to a model of the original $\mathcal{ALCFI}_*$ knowledge base $\mathcal{K}$. The constructions of $\mathcal{I}^t$ and $\mathcal{I}^f$ are quite sophisticated and we state only the final result.

4.5. THEOREM. *An $\mathcal{ALCFI}_*$ knowledge base $\mathcal{K}$ is satisfiable if and only if its $\mathcal{ALCI}_*$-counterpart $\gamma(\mathcal{K})$. is satisfiable.*

Observe that we cannot use a standard filtration argument [Goldblatt 1992] to prove the theorem above. In general, the ability to get a filtration of a model by a finite set of concepts (as $CL(\gamma_1(\mathcal{K})))$ leads to a finite model property. But $\mathcal{ALCFI}_*$ knowledge bases do not have the finite model property, so filtration techniques are not suitable to prove decidability. The construction sketched above instead builds, from a given model of $\gamma(\mathcal{K})$, a model of $\mathcal{K}$ that can be an infinite tree. Overall it is notable that reasoning in $\mathcal{ALCFI}_*$ knowledge bases, which do not have the finite model property, can be encoded, in a simple way, to reasoning in $\mathcal{ALCI}_*$ knowledge bases, which do have it.

The above result allows for establishing the decidability of reasoning in $\mathcal{ALCFI}_*$ knowledge bases. Moreover, since the encoding is polynomial, we get a characterization of the computational complexity.

4.6. THEOREM. *Knowledge base satisfiability (and hence concept consistency and logical implication) in $\mathcal{ALCFI}_*$ is EXPTIME-complete.*

4.4. Qualified Number Restrictions

We now study reasoning in $\mathcal{ALCQI}$ knowledge bases by exhibiting an encoding from $\mathcal{ALCQI}$ knowledge bases to $\mathcal{ALCFI}_*$ knowledge bases [De Giacomo and Lenzerini 1995*b*, De Giacomo 1995]. The encoding is based on the notion of reification (see later). As the encoding is polynomial, we get as a result the EXPTIME-decidability of reasoning in $\mathcal{ALCQI}$ knowledge bases. However, before discussing reasoning in $\mathcal{ALCQI}$ knowledge bases, we discuss some of the issues involved in the simpler logic $\mathcal{ALCQ}$ obtained by dropping inverse roles. This allows us to gain some intuition about results for $\mathcal{ALCQI}$.

4.4.1. Reasoning in $\mathcal{ALCQ}$ Knowledge Bases

Let us first ignore qualified number restrictions. Concepts of $\mathcal{ALCQ}$ without qualified number restrictions –i.e., concepts of $\mathcal{ALC}$– correspond directly to standard PDL formulas (of a restricted form). It is well-known that PDL formulas can be reduced to Deterministic-PDL formulas [Parikh 1981]. The basic idea, reformulated in DLs terms, is to replace each atomic role P by $F_P \circ G_P^*$, where F_P and G_P are new atomic roles that are globally functional (the assertion $\top \sqsubseteq \exists^{\leq 1} F_P \sqcap \exists^{\leq 1} G_P$ holds). Thus we encode $\mathcal{ALC}$ knowledge bases into $\mathcal{ALCF}_*$ knowledge bases of a special form. Let us call the resulting knowledge base $\mathcal{K}'$. We have that $\mathcal{K}$ is satisfiable if and only if it $\mathcal{K}'$ is satisfiable.

We briefly sketch the reasoning behind the proof of this statement. The if direction is straightforward, since any model $\mathcal{I}'$ of $\mathcal{K}'$ can be transformed into a model of $\mathcal{K}$ by interpreting P as $(F_P \circ G_P^*)^{\mathcal{I}'}$. The only if direction is as follows. Both $\mathcal{ALC}$ knowledge bases and $\mathcal{ALCF}_*$ knowledge bases have the *tree model property*: if a knowledge base has a model then it has a tree model, i.e., a model having the form of a *tree*. Hence, without loss of generality, we can restrict our attention to tree models only. Now, there is a one-to-one transformation from tree models $\mathcal{I}_T$ of $\mathcal{K}$ to (tree) models $\mathcal{I}_B$ of $\mathcal{K}'$. Indeed, we take $\Delta^{\mathcal{I}_B} = \Delta^{\mathcal{I}_T}$, $A^{\mathcal{I}_B} = A^{\mathcal{I}_T}$, and given an object $o \in \Delta^{\mathcal{I}_T}$ having as P-successors $o_1, \ldots, o_l$ we take $(o, o_1) \in F_P^{\mathcal{I}_B}$, and $(o_i, o_{i+1}) \in G_P^{\mathcal{I}_B}$, for $i = 1, \ldots, l-1$. In this way we have $(o, o_i) \in P^{\mathcal{I}_T}$ if and only if $(o, o_i) \in (F_P \circ G_P^*)^{\mathcal{I}_B}$. Note that this construction is similar to the one often used in programming to transform an n-ary tree into a binary tree by coding children of a node as the combination of one child and its siblings.

We remark that $\mathcal{I}_T$ is required to be a tree because, once we get $\mathcal{I}_B$, we need to recover the "original" P-predecessor o of an object o_i, and thus $(F_P \circ G_P^*)^-$ needs to be *functional*. Otherwise, given an object o_i, we would not know which of the various $(F_P \circ G_P^*)^-$-successors is its original P-predecessor o, and therefore we would not be able to reconstruct $\mathcal{I}_T$ from $\mathcal{I}_B$.

Now let us consider again $\mathcal{ALCQ}$ knowledge bases. Representing an atomic role P as $F_P \circ (F'_P)^*$, where F_P and F'_P are functional roles, makes it easy to express qualified number restrictions as constraints on the chain of $F_P \circ G_P^*$-successors of an object. For example, denoting the transitive closure of a role F as F^+, i.e.,

$F^+ \doteq F \circ F^*$, the concept $\exists^{\leq 3} P.C$ can be expressed by

$$\forall(F_P \circ G_P^* \circ id(C) \circ G_P^+ \circ id(C) \circ G_P^+ \circ id(C) \circ G_P^+).\neg C$$

which is equivalent to

$$\forall F_P.\forall G_P^*.(C \Rightarrow \forall G_P^+.(C \Rightarrow \forall G_P.\forall G_P^*.(C \Rightarrow \forall G_P.\forall G_P^*.\neg C))).$$

This concept can be read as "everywhere along the chain $F_P \circ G_P^*$ there are at most three objects that are instances of C", which corresponds exactly to the intended meaning.

The concept $\exists^{\geq 3} P.C$ can be expressed by

$$\exists(F_P \circ G_P^* \circ id(C) \circ G_P^+ \circ id(C) \circ G_P^+).C$$

which is equivalent to

$$\exists F_P.\exists G_P^*.(C \sqcap \exists G_P.\exists G_P^*.(C \sqcap \exists G_P.\exists G_P^*.C))$$

and can be read as "somewhere along the chain $F_P \circ G_P^*$ there are at least three objects that are instances of C", which again corresponds exactly to the intended meaning.

4.4.2. Reification of Roles

The construction above does not apply directly to $\mathcal{ALCQI}$ knowledge bases, since the presence of inverse roles does not allow us to restrict the attention to tree-like interpretations of the above form[8]. In order to carry out a similar construction we first need to *reify* atomic roles.

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

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

where A_P is a new atomic concept denoting objects representing the tuples of the relation associated with P, and V_1 and V_2 denote two functional roles that connect each object in A_P to the first and the second component respectively of the tuple represented by the object. Note that there is a clear symmetry between the role

[8] Indeed the presence of inverse roles allows for restricting the attention to "two-way" tree interpretations, as opposed to the "one-way" tree interpretations needed here.

$V_1^- \circ id(A_P) \circ V_2$ and its inverse $V_2^- \circ id(A_P) \circ V_1$. Observe that such complex roles are not part of $\mathcal{ALCQI}$, however given a concept of the form $\exists(V_1^- \circ id(A_P) \circ V_2).C$ we can immediately transform it into an $\mathcal{ALCQI}$ concept, namely $\exists V_1^-.(A_P \sqcap \exists V_2.C)$. Similarly for universal and number restrictions.

4.7. Definition. Let $\mathcal{K}$ be an $\mathcal{ALCQI}$ knowledge base. The *reified-counterpart* $\xi_1(\mathcal{K})$ of $\mathcal{K}$ is the union of two knowledge bases, $\xi_1(\mathcal{K}) = \xi_0(\mathcal{K}) \cup \Theta_1$, where:

- $\xi_0(\mathcal{K})$ is obtained from the original knowledge base $\mathcal{K}$ by recursively replacing, for every atomic role P, every existential restriction, universal restriction and qualified number restriction as follows:

$$\begin{array}{lll}
\exists P.C & \text{by} & \exists V_1^-.(A_P \sqcap \exists V_2.C),\\
\forall P.C & \text{by} & \forall V_1^-.(A_P \Rightarrow \forall V_2.C),\\
\exists^{\leq n} P.C & \text{by} & \exists^{\leq n} V_1^-.(A_P \sqcap \exists V_2.C),\\
\exists^{\geq n} P.C & \text{by} & \exists^{\geq n} V_1^-.(A_P \sqcap \exists V_2.C),\\
\exists P^-.C & \text{by} & \exists V_2^-.(A_P \sqcap \exists V_1.C),\\
\forall P^-.C & \text{by} & \forall V_2^-.(A_P \Rightarrow \forall V_1.C),\\
\exists^{\leq n} P^-.C & \text{by} & \exists^{\leq n} V_2^-.(A_P \sqcap \exists V_1.C),\\
\exists^{\geq n} P^-.C & \text{by} & \exists^{\geq n} V_2^-.(A_P \sqcap \exists V_1.C),
\end{array}$$

  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.
- $\Theta_1 = \{\top \sqsubseteq \exists^{\leq 1} V_1 \sqcap \exists^{\leq 1} V_2\}$.

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

4.8. Lemma. *If $\xi_1(\mathcal{K})$ 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 $o_{oo'}$ 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:*

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

The proof of Lem. 4.8 exploits the *disjoint union model property*: let $\mathcal{K}$ be an $\mathcal{ALCQI}$ knowledge base and $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ and $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be two models of $\mathcal{K}$, 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 $\mathcal{K}$. We remark that most DLs have such a property, which is, in fact, typical of modal logics. Without going into details, we 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 object in one of the copies of $\mathcal{I}$. To prove that $\mathcal{I}'$ is indeed a model one can exploit, e.g., bisimulation, see also [Clarke and Schlingloff 2001] (Chapter 24 of this Handbook).

By using Lem. 4.8 we can prove the result below.

4.9. Lemma. *Let $\mathcal{K}$ be an $\mathcal{ALCQI}$ knowledge base and $\xi_1(\mathcal{K})$ its reified-counterpart. Then $\mathcal{K}$ is satisfiable if and only if $\xi_1(\mathcal{K})$ is satisfiable.*

4.4.3. Reducing $\mathcal{ALCQI}$ Knowledge Bases to $\mathcal{ALCFI}_*$ Knowledge Bases

By Lemma 4.9, we can concentrate on the reified-counterparts of $\mathcal{ALCQI}$ knowledge bases. Note that these are $\mathcal{ALCQI}$ knowledge bases themselves, but their special form allows us to convert them into $\mathcal{ALCFI}_*$ knowledge bases. We adopt a technique resembling the one exploited for encoding $\mathcal{ALCQ}$ knowledge bases into $\mathcal{ALCF}_*$ knowledge bases. Intuitively, we represent the role V_i^- $(i = 1, 2)$, which is not functional (while V_i is so), by the role $F_{V_i} \circ G^*_{V_i}$, where F_{V_i} and G_{V_i} are new functional roles. The main point of such a transformation is that now qualified number restrictions can be encoded as constraints on the chain $F_{V_i} \circ G^*_{V_i}$. Formally, we define the $\mathcal{ALCFI}_*$-counterpart of an $\mathcal{ALCQI}$ knowledge base as follows.

4.10. Definition. Let $\mathcal{K}$ be an $\mathcal{ALCQI}$ knowledge base and $\xi_1(\mathcal{K}) = \xi_0(\mathcal{K}) \cup \Theta_1$ its reified-counterpart. The *$\mathcal{ALCFI}_*$-counterpart* $\xi_2(\mathcal{K})$ of $\mathcal{K}$ is the union of two knowledge bases, $\xi_2(\mathcal{K}) = \xi_0'(\mathcal{K}) \cup \Theta_2$, where:

- $\xi_0'(\mathcal{K})$ is obtained from $\xi_0(\mathcal{K})$ by recursively replacing, for V_i $(i = 1, 2)$, every existential restriction, universal restriction and qualified number restriction as follows:

$$\begin{array}{lll}
\exists V_i.C & \text{by} & \exists (G^-_{V_i})^*.\exists F^-_{V_i}.C \\
\forall V_i.C & \text{by} & \forall (G^-_{V_i})^*.\forall F^-_{V_i}.C \\
\exists V_i^-.C & \text{by} & \exists F_{V_i}.\exists G^*_{V_i}.C \\
\forall V_i^-.C & \text{by} & \forall F_{V_i}.\forall G^*_{V_i}.C \\
\exists^{\leq n} V_i^-.C & \text{by} & \forall F_{V_i}.\forall G^*_{V_i}.(C \Rightarrow \forall G^+_{V_i})^n.\neg C \\
\exists^{\geq n} V_i^-.C & \text{by} & \exists F_{V_i}.\exists G^*_{V_i}.(C \sqcap \exists G^+_{V_i})^{n-1}.C
\end{array}$$

  where F_{V_i} and G_{V_i} are new atomic roles and $(C \Rightarrow \forall G^+_{V_i})^n.\neg C$ stands for

$$(C \Rightarrow \forall G_{V_i}.\forall G^*_{V_i}.(\cdots (C \Rightarrow \forall G_{V_i}.\forall G^*_{V_i}.\neg C)\cdots))$$

  in which the pattern $C \Rightarrow \forall G_{V_i}.\forall G^*_{V_i}.\cdot$ is repeated n times. Similarly, for $(C \sqcap \exists G^+_{V_i})^{n-1}.C$.
- $\Theta_2 = \{\theta_1, \theta_2\}$, with θ_i of the form:

$$\top \sqsubseteq \exists^{\leq 1} F_{V_i} \sqcap \exists^{\leq 1} G_{V_i} \sqcap \exists^{\leq 1} F^-_{V_i} \sqcap \exists^{\leq 1} G^-_{V_i} \sqcap \neg(\exists F^-_{V_i}.\top \sqcap \exists G^-_{V_i}.\top).$$

Observe that Θ_2 constrains the models $\mathcal{I}$ of $\xi_2(\mathcal{K})$ so that the relations $F^{\mathcal{I}}_{V_i}$, $G^{\mathcal{I}}_{V_i}$, $(F^-_{V_i})^{\mathcal{I}}$, $(G^-_{V_i})^{\mathcal{I}}$ are partial functions, and each object cannot be linked to other objects by both $(F^-_{V_i})^{\mathcal{I}}$ and $(G^-_{V_i})^{\mathcal{I}}$. As a consequence we get that $((F_{V_i} \circ G^*_{V_i})^-)^{\mathcal{I}}$ is a partial function. This condition is required to prove the lemma below.

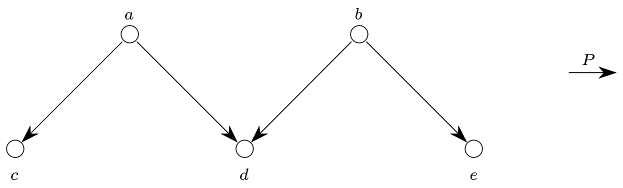

Figure 1: A model of the knowledge base $\mathcal{K} = \{C_0 \equiv \exists P.(\exists^{=2} P^-.(\exists^{=2} P.\top))\}$

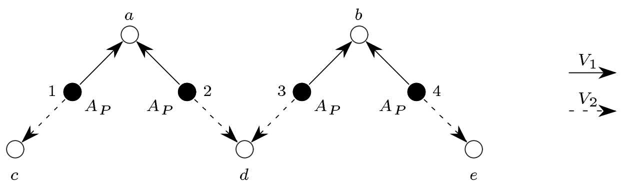

Figure 2: A model of the reified-counterpart $\xi_1(\mathcal{K})$ of $\mathcal{K}$

4.11. LEMMA. *Let $\mathcal{K}$ be a $\mathcal{ALCQI}$ knowledge base, $\xi_1(\mathcal{K})$ its reified-counterpart, and $\xi_2(\mathcal{K})$ its $\mathcal{ALCFI}_*$-counterpart. Then $\xi_1(\mathcal{K})$ is satisfiable if and only if $\xi_2(\mathcal{K})$ is satisfiable.*

As an immediate consequence of Lem. 4.11 and Lem. 4.9 we get the main result of this subsection.

4.12. THEOREM. *Let $\mathcal{K}$ be a $\mathcal{ALCQI}$ knowledge base and $\xi_2(\mathcal{K})$ its $\mathcal{ALCFI}_*$-counterpart. Then $\mathcal{K}$ is satisfiable if and only if its $\xi_2(\mathcal{K})$ is satisfiable.*

Since the size of $\xi_2(\mathcal{K})$ is polynomial in the size of $\mathcal{K}$, we get the following complexity characterization for reasoning in $\mathcal{ALCQI}$ knowledge bases [9].

4.13. THEOREM. *Knowledge base satisfiability (and hence concept consistency and logical implication) in $\mathcal{ALCQI}$ is EXPTIME-complete.*

Let us illustrate with an example the basic relationships between models of an $\mathcal{ALCQI}$ knowledge base and those of its reified-counterpart and $\mathcal{ALCFI}_*$-counterpart.

4.14. EXAMPLE. Let $\mathcal{K}$ be the following $\mathcal{ALCQI}$ knowledge base:

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

Figure 1 depicts a model $\mathcal{I}$ of $\mathcal{K}$ with $a \in C_0^{\mathcal{I}}$. In Figure 2 the model $\mathcal{I}$ of $\mathcal{K}$ is transformed into a model $\mathcal{I}'$ of its reified-counterpart $\xi_1(\mathcal{K})$. Finally, in Figure 3

[9] We remind that we assume unary coding of numbers in number restrictions.

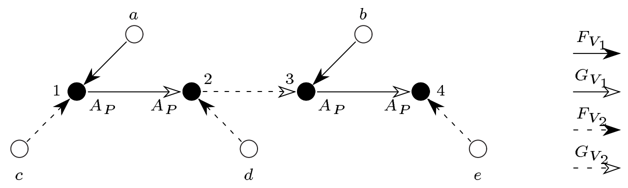

Figure 3: A model of the $\mathcal{ALCFI}_*$-counterpart $\xi_2(\mathcal{K})$ of $\mathcal{K}$

the model $\mathcal{I}'$ of $\xi_1(\mathcal{K})$ is transformed into a model $\mathcal{I}''$ of the $\mathcal{ALCFI}_*$-counterpart $\xi_2(\mathcal{K})$ of $\mathcal{K}$. Notice that from $\mathcal{I}''$ we can easily reconstruct $\mathcal{I}'$, and from $\mathcal{I}'$ the model $\mathcal{I}$ of the original knowledge base.

5. Finite Model Reasoning

As shown in Sect. 2, a DL which contains constructs for inverse roles and functionality in combination with cyclic assertions lacks the finite model property. Therefore, it becomes necessary to devise specific techniques to reason with respect to finite models only. In this section we discuss the problem of reasoning with respect to finite models in knowledge bases, and propose methods to solve this problem for various sublanguages of $\mathcal{ALCQI}$. These methods are based on an idea introduced by Lenzerini and Nobili [1990] and further developed and extended by Calvanese and Lenzerini [1994*a*, 1994*b*] and Calvanese [1996*a*].

When we restrict the attention to finite models, some properties that are essential for the reasoning techniques developed in Sect. 4 fail. In particular, all reductions exploiting the tree-model property 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. The following example shows that there are indeed cases where any finite model of a knowledge base must necessarily contain a cycle.

5.1. Example. Let $\mathcal{K}_c$ be the $\mathcal{ALUNI}$ knowledge base consisting of one assertion:

$$\texttt{Node} \sqsubseteq \forall\texttt{edge}^-.\texttt{Node} \sqcap \exists^{\leq 1}\texttt{edge}^- \sqcap \forall\texttt{edge}.\texttt{Node} \sqcap \exists\texttt{edge}$$

As in Example 2.2, the above assertion forces in any model of $\mathcal{K}_c$ in which `Node` has a nonempty extension the existence of a sequence of objects each connected to the following one by means of the role `edge`. However, differently from Example 2.2, the assertion does not rule out that the $\texttt{edge}^\mathcal{I}$ relation forms a cycle. It is easy to verify that each finite model of $\mathcal{K}_c$ will be constituted by a finite number of such cycles.

If number restrictions (and in particular functionality) are not used, the finite model property does not fail. Therefore, a procedure for finite model reasoning must specifically address the presence of number restrictions. The method we present here is based on an encoding of the number restrictions that appear in a knowledge base into a system of linear inequalities. The method represents a generalization of the one developed by Lenzerini and Nobili [1990] for a simple data model based on disjoint classes and relationships. We first briefly describe the method on knowledge bases of a restricted form, which captures the data model of Lenzerini and Nobili [1990]. We then show how to extend it to capture finite model reasoning in knowledge bases built using more expressive languages, namely $\mathcal{ALUNI}$ and $\mathcal{ALCNI}$.

5.1. Finite Model Reasoning Using Linear Inequalities

To illustrate the reasoning technique we consider $\mathcal{ALNI}$ knowledge bases containing only assertions of a simplified form. Such knowledge bases essentially correspond to schemas expressed in the data model used by Lenzerini and Nobili [1990] (with roles representing binary relationships), which is a simplified form of the Entity-Relationship model [Chen 1976] without ISA relationships between classes. A knowledge base $\mathcal{K}$ of this form contains for each atomic role P an assertion

$$\top \sqsubseteq \forall P.A_2 \sqcap \forall P^-.A_1, \tag{5.1}$$

where A_1 and A_2 are two atomic concepts, and for each pair of distinct atomic concepts A and A', an assertion

$$A \sqsubseteq \neg A'. \tag{5.2}$$

These assertions enforce that in all models of $\mathcal{K}$ the following properties hold:

(Cond_1) The atomic concepts have pairwise disjoint extensions.

(Cond_2) Each role is "typed", i.e., 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.

The only additional assertions that are present in $\mathcal{K}$ are used to impose cardinality constraints on roles and inverse roles, and are of the form

$$\begin{array}{lcl} \top & \sqsubseteq & \exists^{\geq m_1} P \sqcap \exists^{\leq n_1} P \\ \top & \sqsubseteq & \exists^{\geq m_2} P^- \sqcap \exists^{\leq n_2} P^-, \end{array} \tag{5.3}$$

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 restrictions on the use of constructs and assertions (which reflect the limited expressiveness of the data model), a knowledge base of the above form is always both satisfiable and finitely satisfiable, and all atomic concepts in it are consistent. However, due to the use of number restrictions some of the atomic concepts may not be finitely consistent.

In order to check for finite concept consistency , we construct from $\mathcal{K}$ a system $\Psi_{\mathcal{K}}$ of linear inequalities with one unknown $\text{Var}(X)$ for each atomic concept or role

X in $\mathcal{K}$. The system contains inequalities

$$\begin{array}{rcccl} m_1 \cdot \mathrm{Var}(A_1) & \leq & \mathrm{Var}(P) & \leq & n_1 \cdot \mathrm{Var}(A_1) \\ m_2 \cdot \mathrm{Var}(A_2) & \leq & \mathrm{Var}(P) & \leq & n_2 \cdot \mathrm{Var}(A_2) \end{array} \tag{5.4}$$

corresponding to the assertions (5.3), and for each atomic concept or role X, an inequality

$$\mathrm{Var}(X) > 0. \tag{5.5}$$

It can be shown that all atomic concepts in $\mathcal{K}$ are simultaneously consistent if and only if the associated system of inequalities $\Psi_{\mathcal{K}}$ admits a solution.

The proof of this result exploits the possibility of associating to a positive integer solution of $\Psi_{\mathcal{K}}$ a model of $\mathcal{K}$ in such a way that the cardinality of the extension of each atomic concept and role equals the value assigned by the solution to the corresponding unknown. One direction is easy, since from any model of $\mathcal{K}$ one can obtain an integer solution of $\Psi_{\mathcal{K}}$ by assigning to each unknown the cardinality of the extension of the corresponding atomic concept or role. The converse direction needs more attention. First of all, since the system is homogeneous, if it admits a solution then it admits also a solution $\mathcal{S}$ such that (i) for each atomic concept and role E the value $\mathcal{S}(E)$ assigned by $\mathcal{S}$ to $\mathrm{Var}(E)$ is a positive integer, and (ii) for each P, A_1, A_2 appearing in an assertion of the form (5.1), it holds that $\mathcal{S}(P) \geq \mathcal{S}(A_1) \cdot \mathcal{S}(A_2)$ [Lenzerini and Nobili 1990]. From such a solution $\mathcal{S}$ one can construct a model $\mathcal{I}$ of the knowledge base in which $X^{\mathcal{I}}$ has cardinality $\mathcal{S}(X)$, for each atomic concept and role X. Assertions of the form (5.2) are satisfied by assigning to all atomic concepts disjoint extensions. In order to satisfy assertions (5.1), the extension $P^{\mathcal{I}}$ of each atomic role P is obtained as a set of pairs of instances of A_1 and A_2. Moreover, since $\mathcal{S}$ satisfies all inequalities (5.4), $P^{\mathcal{I}}$ can be constructed in such a way that also all assertions (5.3) involving number restrictions are satisfied. The reason is that conditions (Cond_1) and (Cond_2) allow one to regard all instances of an atomic concept as equivalent with respect to the participation in roles. Therefore the number restrictions (5.3), which are local conditions on the number of connections of single objects, are correctly encoded by means of the inequalities (5.4), which express global conditions on the total number of instances.

Since verifying the existence of a solution of $\Psi_{\mathcal{K}}$ can be done in time polynomial in the size of $\Psi_{\mathcal{K}}$[10], and hence time polynomial in the size of $\mathcal{K}$, we have an efficient method for verifying concept consistency in the knowledge bases considered so far.

The technique presented above is only suitable if one needs to verify the simultaneous consistency of *all* atomic concepts, which is reflected in the use of inequalities (5.5). In general, however, one may want to check the consistency of single concepts without posing any restrictions on other concepts of the knowledge base. Therefore, the solutions of the system of inequalities need to reflect also the case where in a model some atomic concept A, and hence all roles that are typed with A, have an empty extension. This means that when a solution of the system assigns the value 0 to the unknown corresponding to A, it must assign the value 0 also to all

[10] We assume to use a suitable encoding of $\Psi_{\mathcal{K}}$ in which numbers are represented in binary.

unknowns corresponding to the roles typed with A. Such a condition is not satisfied by every solution of the system, and we call a solution in which the condition holds *acceptable*.

To verify the existence of acceptable solutions of the system of inequalities derived from an $\mathcal{ALNI}$ knowledge base of the simple form above one can exploit a result about the complexity of integer programming, which essentially states that the existence of integer solutions of a system of inequalities implies the existence of solutions of bounded size [Papadimitriou 1981]. Such a result, combined with the technique presented by Calvanese and Lenzerini [1994*b*] allows one to show that the existence of acceptable solutions can be verified in polynomial time in the size of the system of inequalities [Calvanese 1996*c*].

5.2. Finite Model Reasoning on Primitive $\mathcal{ALUNI}$ Knowledge Bases

The method we have presented in Sect. 5.1 is not immediately applicable to more expressive languages and more general forms of knowledge bases. The reason is that conditions (Cond_1) and (Cond_2), which are necessary to construct a model of a knowledge base from a solution of the corresponding system of linear inequalities, are in general not satisfied. In fact, to correctly encode number restrictions by means of linear inequalities, it is sufficient that the following generalization of condition (Cond_2) is satisfied:

(Cond'_2) For each atomic role P and each concept expression C appearing in $\mathcal{K}$, 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 conditions encoded by number restrictions can be correctly captured by the global constraints represented by the system of inequalities.

It is possible to enforce conditions (Cond_1) and (Cond'_2) by performing a transformation on the knowledge base, and deriving from the transformed version the system of inequalities. A technique that makes use of this idea for finite model reasoning in expressive modeling formalisms was first introduced by Calvanese and Lenzerini [1994*b*] for an extended semantic database model, and successively extended to deal with primitive $\mathcal{ALUNI}$ knowledge bases [Calvanese et al. 1994] and for reasoning in an expressive object-oriented database model [Calvanese and Lenzerini 1994*a*]. We illustrate the technique for reasoning on primitive $\mathcal{ALUNI}$ knowledge bases by means of an example, using a slight variation of the knowledge base in Example 5.1.

5.2. Example. Let $\mathcal{K}_b$ be the $\mathcal{ALUNI}$ knowledge base consisting of the assertions:

$$
\begin{array}{rcl}
\mathtt{Root} & \sqsubseteq & \exists^{\leq 0}\mathtt{edge}^- \sqcap \forall\mathtt{edge}.\mathtt{OtherNode} \sqcap \exists^{\geq 1}\mathtt{edge} \sqcap \exists^{\leq 2}\mathtt{edge} \\
\mathtt{OtherNode} & \sqsubseteq & \forall\mathtt{edge}^-.(\mathtt{Root} \sqcup \mathtt{OtherNode}) \sqcap \exists^{=1}\mathtt{edge}^- \sqcap \\
 & & \forall\mathtt{edge}.\mathtt{OtherNode} \sqcap \exists^{\geq 1}\mathtt{edge} \sqcap \exists^{\leq 2}\mathtt{edge} \sqcap \neg\mathtt{Root}.
\end{array}
$$

$\mathcal{K}_b$ describes the properties of binary trees, where each node has at most one predecessor and at most two successors. In fact, we distinguish between nodes that have no predecessor (`Root`) and those that have one (`OtherNode`). Additionally, we require that each node has at least one successor. This combination of requirements makes the concept `Root` finitely inconsistent. The concept `OtherNode`, on the other hand, can be populated in a finite model of the knowledge base, although the model we obtain is somewhat counterintuitive, since it necessarily contains a cycle in the `edge` relation.

First of all, it is easy to see that, by introducing at most a linear number of new atomic concepts, we can transform the knowledge base into an equivalent one in which the nesting of constructs is eliminated. Specifically, in such a knowledge base, which we call *normalized*, the concept on the right hand side of an inclusion assertion is of the form L, $L_1 \sqcup L_2$, $\forall R.L$, $\exists^{\geq n} R$, or $\exists^{\leq n} R$, where L is an atomic or negated atomic concept.

5.3. Example (5.2 continued). In the normalized $\mathcal{ALUNI}$ knowledge base $\mathcal{K}_n$ corresponding to $\mathcal{K}_b$ we introduce an additional concept `Node` to replace the disjunction `Root` $\sqcup$ `OtherNode` nested within the universal quantification, thus obtaining:

$$
\begin{array}{lcllcl}
\mathtt{Node} & \sqsubseteq & \mathtt{Root} \sqcup \mathtt{OtherNode} & \mathtt{OtherNode} & \sqsubseteq & \forall \mathtt{edge}^-.\mathtt{Node} \\
 & & & \mathtt{OtherNode} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge}^- \\
\mathtt{Root} & \sqsubseteq & \exists^{\leq 0}\mathtt{edge}^- & \mathtt{OtherNode} & \sqsubseteq & \exists^{\leq 1}\mathtt{edge}^- \\
\mathtt{Root} & \sqsubseteq & \forall \mathtt{edge}.\mathtt{OtherNode} & \mathtt{OtherNode} & \sqsubseteq & \forall \mathtt{edge}.\mathtt{OtherNode} \\
\mathtt{Root} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge} & \mathtt{OtherNode} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge} \\
\mathtt{Root} & \sqsubseteq & \exists^{\leq 2}\mathtt{edge} & \mathtt{OtherNode} & \sqsubseteq & \exists^{\leq 2}\mathtt{edge} \\
 & & & \mathtt{OtherNode} & \sqsubseteq & \neg\mathtt{Root}.
\end{array}
$$

Then, to ensure that conditions (Cond_1) and (Cond_2') are satisfied, we use instead of atomic concepts, sets of atomic concepts, called *compound concepts*, and instead of atomic roles, so called *compound roles*. Each compound role is a triple $(P, \widehat{D}_1, \widehat{D}_2)$ constituted by an atomic role P and two compound concepts $\widehat{D}_1$ and $\widehat{D}_2$ typing respectively the first and second component of the instances of P. Intuitively, the instances of a compound concept $\widehat{D}$ are all those objects of the domain that are instances of all concepts in $\widehat{D}$ and are not instances of any concept not in $\widehat{D}$. A compound role $(P, \widehat{D}_1, \widehat{D}_2)$ is interpreted as the restriction of role P to the pairs whose first component is an instance of $\widehat{D}_1$ and whose second component is an instance of $\widehat{D}_2$.

This ensures that two different compound concepts have necessarily disjoint extensions, and hence that the property corresponding to (Cond_1) holds. The same observation holds for two different compound roles $(P, \widehat{D}_1, \widehat{D}_2)$ and $(P, \widehat{D}_1', \widehat{D}_2')$ that correspond to the same role P. Moreover, for compound roles the property corresponding to property (Cond_2) holds by definition, and, considering that the knowledge base is primitive and has been normalized, also (Cond_2') holds.

The assertions in the knowledge base that do not involve number restrictions, force certain compound concepts and compound roles to be *inconsistent*, i.e., to have an empty extension in all models of the knowledge base. For example, the assertion $A_1 \sqsubseteq \neg A_2$ makes all compound concepts that contain both A_1 and A_2 inconsistent. Similarly, the assertion $A_1 \sqsubseteq \forall P.A_2$ makes all compound roles $(P, \widehat{D}_1, \widehat{D}_2)$ such that $\widehat{D}_1$ contains A_1 and $\widehat{D}_2$ does not contain A_2 inconsistent. One can check in polynomial time in the size of the knowledge base whether a compound concept or 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 knowledge base, doing the check for all compound concepts and roles takes in general exponential time. One can then encode the assertions involving number restrictions in the normalized knowledge base into assertions involving number restrictions on compound concepts.

5.4. EXAMPLE (5.3 CONTINUED). For the normalized knowledge base $\mathcal{K}_n$, the set of consistent compound concepts is $\widehat{\mathcal{D}}_n = \{\mathtt{E}, \mathtt{R}, \mathtt{O}, \mathtt{RN}, \mathtt{ON}\}$, where $\mathtt{E} = \emptyset$, $\mathtt{R} = \{\mathtt{Root}\}$, $\mathtt{O} = \{\mathtt{OtherNode}\}$ $\mathtt{RN} = \{\mathtt{Node}, \mathtt{Root}\}$, and $\mathtt{ON} = \{\mathtt{Node}, \mathtt{OtherNode}\}$, and the set of consistent compound roles is $\widehat{\mathcal{U}}_n = \{(\mathtt{edge}, \mathtt{RN}, \mathtt{O}), (\mathtt{edge}, \mathtt{RN}, \mathtt{ON}), (\mathtt{edge}, \mathtt{ON}, \mathtt{O}), (\mathtt{edge}, \mathtt{ON}, \mathtt{ON}), (\mathtt{edge}, \mathtt{E}, \mathtt{R}), (\mathtt{edge}, \mathtt{E}, \mathtt{RN}), (\mathtt{edge}, \mathtt{E}, \mathtt{E})\}$. The assertions that derive from the assertions in $\mathcal{K}_n$ and which impose number restrictions on compound concepts are the following:

$$
\begin{array}{lllclll}
\mathtt{R} & \sqsubseteq & \exists^{\leq 0}\mathtt{edge}^- & \qquad & \mathtt{RN} & \sqsubseteq & \exists^{\leq 0}\mathtt{edge}^- \\
\mathtt{R} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge} & & \mathtt{RN} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge} \\
\mathtt{R} & \sqsubseteq & \exists^{\leq 2}\mathtt{edge} & & \mathtt{RN} & \sqsubseteq & \exists^{\leq 2}\mathtt{edge} \\
\mathtt{O} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge}^- & & \mathtt{ON} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge}^- \\
\mathtt{O} & \sqsubseteq & \exists^{\leq 1}\mathtt{edge}^- & & \mathtt{ON} & \sqsubseteq & \exists^{\leq 1}\mathtt{edge}^- \\
\mathtt{O} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge} & & \mathtt{ON} & \sqsubseteq & \exists^{\geq 1}\mathtt{edge} \\
\mathtt{O} & \sqsubseteq & \exists^{\leq 2}\mathtt{edge} & & \mathtt{ON} & \sqsubseteq & \exists^{\leq 2}\mathtt{edge}.
\end{array}
$$

We construct now a system $\Psi_{\mathcal{K}}$ of linear inequalities which contains one unknown for each consistent compound concept and role (the inconsistent compound concepts and roles are not considered anymore). The inequalities are obtained by encoding the assertions in the knowledge base involving number restrictions on compound concepts, in a way similar to inequalities (5.4). Differently from the previous case, now each inequality involves one unknown corresponding to a compound concept and a sum of unknowns corresponding to compound roles. For example, the assertion $\widehat{D} \sqsubseteq \exists^{\geq n}P$, where $\widehat{D}$ is a compound concept and P is an atomic role, results in the inequality

$$n \cdot \mathrm{Var}(\widehat{D}) \leq \sum_{(P,\widehat{D},\widehat{D}_2)} \mathrm{Var}((P, \widehat{D}, \widehat{D}_2)),$$

where the sum ranges over all consistent compound roles involving P and typed with $\widehat{D}$ on the first component.

5.5. EXAMPLE (5.4 CONTINUED). The system $\Psi_{\mathcal{K}_n}$ derived from $\mathcal{K}_n$ consists of the inequalities

$$
\begin{array}{lcl@{\qquad\qquad}lcl}
0{\cdot}\mathtt{r} & \geq & \mathtt{e}_{\mathtt{E,R}} & 0{\cdot}\mathtt{rn} & \geq & \mathtt{e}_{\mathtt{E,RN}} \\
1{\cdot}\mathtt{r} & \leq & 0 & 1{\cdot}\mathtt{rn} & \leq & \mathtt{e}_{\mathtt{RN,O}} + \mathtt{e}_{\mathtt{RN,ON}} \\
2{\cdot}\mathtt{r} & \geq & 0 & 2{\cdot}\mathtt{rn} & \geq & \mathtt{e}_{\mathtt{RN,O}} + \mathtt{e}_{\mathtt{RN,ON}} \\
1{\cdot}\mathtt{o} & \leq & \mathtt{e}_{\mathtt{RN,O}} + \mathtt{e}_{\mathtt{ON,O}} & 1{\cdot}\mathtt{on} & \leq & \mathtt{e}_{\mathtt{RN,ON}} + \mathtt{e}_{\mathtt{ON,ON}} \\
1{\cdot}\mathtt{o} & \geq & \mathtt{e}_{\mathtt{RN,O}} + \mathtt{e}_{\mathtt{ON,O}} & 1{\cdot}\mathtt{on} & \geq & \mathtt{e}_{\mathtt{RN,ON}} + \mathtt{e}_{\mathtt{ON,ON}} \\
1{\cdot}\mathtt{o} & \leq & 0 & 1{\cdot}\mathtt{on} & \leq & \mathtt{e}_{\mathtt{ON,O}} + \mathtt{e}_{\mathtt{ON,ON}} \\
2{\cdot}\mathtt{o} & \geq & 0 & 2{\cdot}\mathtt{on} & \geq & \mathtt{e}_{\mathtt{ON,O}} + \mathtt{e}_{\mathtt{ON,ON}},
\end{array}
$$

where we have denoted the unknown corresponding to a compound concept with the name of the compound concept in lower-case letters, and the unknown corresponding to a compound role $(\mathtt{edge}, X, Y)$ with $\mathtt{e}_{X,Y}$.

It is possible to show that *acceptable* nonnegative integer solutions of $\Psi_{\mathcal{K}}$ (suitably defined), can be put into correspondence with finite models of the knowledge base $\mathcal{K}$, in which each compound concept and compound role has a number of instances that is equal to the value assigned by the solution to the corresponding unknown.

Hence, in order to check whether a concept A in a primitive $\mathcal{ALUNI}$ knowledge base is finitely consistent, we can proceed as follows: We derive from the knowledge base the system of inequalities. We then add an additional (non-homogeneous) inequality that forces the solutions of the system to assign a positive value to at least one of the unknowns corresponding to the compound concepts containing A. By exploiting the correspondence between acceptable solutions of the system of inequalities and models of the knowledge base we get the following characterization of finite concept consistency in primitive $\mathcal{ALUNI}$ knowledge bases.

5.6. THEOREM. *Let $\mathcal{K}$ be a knowledge base, A an atomic concept in $\mathcal{K}$, and $\Psi_{\mathcal{K}}$ the system of inequalities derived from $\mathcal{K}$. Then A is finitely consistent in $\mathcal{K}$ if and only if*

$$\Psi_{\mathcal{K}}^{A} = \Psi_{\mathcal{K}} \bigcup \left\{ \sum_{\widehat{D} \mid A \in \widehat{D}} \mathit{Var}(\widehat{D}) \geq 1 \right\}$$

admits an acceptable nonnegative integer solution.

5.7. EXAMPLE (5.5 CONTINUED). To check whether the concept $\mathtt{Root}$ is finitely consistent in $\mathcal{K}_0$, we add to the system $\Psi_{\mathcal{K}_n}$ the inequality $\mathtt{r} + \mathtt{rn} \geq 1$, which forces $\mathtt{R}$ or $\mathtt{RN}$ to be populated, obtaining (after some simplifications)

$$
\begin{array}{l}
\mathtt{r} = \mathtt{o} = 0 \\
\mathtt{e}_{\mathtt{RN,O}} = \mathtt{e}_{\mathtt{ON,O}} = \mathtt{e}_{\mathtt{E,R}} = \mathtt{e}_{\mathtt{E,RN}} = 0 \\
\mathtt{rn} \;\leq\; \mathtt{e}_{\mathtt{RN,ON}} \;\leq\; 2{\cdot}\mathtt{rn} \\
\mathtt{on} \;\leq\; \mathtt{e}_{\mathtt{ON,ON}} \;\leq\; 2{\cdot}\mathtt{on}
\end{array}
$$

$$\begin{aligned} \texttt{on} &= \texttt{e}_{\texttt{RN},\texttt{ON}} + \texttt{e}_{\texttt{ON},\texttt{ON}} \\ \texttt{r} + \texttt{rn} &\geq 1. \end{aligned}$$

It is easy to see that the inequalities in $\Psi_{\mathcal{K}_n}$ force both $\texttt{r}$ and $\texttt{rn}$ to get assigned value 0, and therefore the whole system admits no solution. This shows that the concept $\texttt{Root}$ cannot be populated in any finite model of $\mathcal{K}_0$.

Similarly, to check whether the concept $\texttt{OtherNode}$ is finitely consistent in $\mathcal{K}_0$, we can add the following inequality to $\Psi_{\mathcal{K}_n}$:

$$\texttt{o} + \texttt{on} \geq 1.$$

All solutions of the resulting system are acceptable. By simplifying the system we can verify that every solution assigns 0 to all unknowns except to $\texttt{on}$ and to $\texttt{e}_{\texttt{ON},\texttt{ON}}$, to which it assigns the same value. Let *Sol* be such a solution which assigns to $\texttt{on}$ and to $\texttt{e}_{\texttt{ON},\texttt{ON}}$ the positive integer k. We can obtain from *Sol* a finite model $\mathcal{I}$ of $\mathcal{K}_0$. Since $\texttt{ON}$ must have k instances and it is the only compound concept with a nonempty extension, we define $\Delta^{\mathcal{I}} = \texttt{ON}^{\mathcal{I}} = \texttt{OtherNode}^{\mathcal{I}} = \{o_1, \ldots, o_k\}$. By setting $(\texttt{edge}, \texttt{ON}, \texttt{ON})^{\mathcal{I}} = \texttt{edge}^{\mathcal{I}} = \{(o_i, o_i) \mid i \in \{1, \ldots, k\}\}$, we obtain indeed a model of $\mathcal{K}_0$, although not one in which the $\texttt{edge}$ relation has a tree-like structure.

The above result can be used for arbitrary concepts as well. An arbitrary concept C is consistent in $\mathcal{K}$ if and only if A_C is consistent in $\mathcal{K} \cup \{A_C \sqsubseteq C\}$, where A_C is an atomic concept not appearing in $\mathcal{K}$. As for the computational complexity of the method, since the existence of an acceptable solution of a system of inequalities can be verified in polynomial time in the size of the system, by Th. 5.6 we obtain the following upper bound for finite concept consistency.

5.8. Theorem. *Whether a concept C is finitely consistent in a primitive $\mathcal{ALUNI}$ knowledge base $\mathcal{K}$ can be decided in worst case deterministic exponential time in the size of $\mathcal{K}$ plus the size of C.*

Since already for primitive $\mathcal{ALU}$ knowledge bases verifying concept consistency is EXPTIME-hard [Calvanese 1996*b*], the above method provides a computationally optimal reasoning procedure.

5.9. Theorem. *Finite concept consistency in primitive $\mathcal{ALUNI}$ knowledge bases is EXPTIME-complete.*

The method we have described to decide concept consistency with respect to primitive $\mathcal{ALUNI}$ knowledge bases can be extended to handle also a wider class of knowledge bases, 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 assertions. In particular, this makes it possible to deal also with knowledge bases containing definitions of concepts that are boolean combinations of atomic concepts, and perform finite model reasoning on such knowledge bases in deterministic exponential time.

Since $\mathcal{ALUNI}$ is not closed under negation we cannot immediately reduce logical implication to concept consistency. However, by making use of the observation above, the technique for deciding concept consistency can be extended to decide also finite logical implication $\mathcal{K} \models_f C_1 \sqsubseteq C_2$ in relevant cases in deterministic exponential time. More precisely, C_1 can be an arbitrary $\mathcal{ALUNI}$ concept and C_2 is required either to not contain the construct for universal quantification or to be of the form $\forall R.C'$ with C' a boolean combination of atomic concepts (see [Calvanese 1996*c*] for details). For the more general case, where also C_2 is an arbitrary concept expression, we need to resort to more involved techniques of higher computational complexity, which we briefly sketch in the next section.

5.3. Finite Model Reasoning in $\mathcal{ALCNI}$ and $\mathcal{ALCQI}$ Knowledge Bases

The method described in Section 5.2 can also be extended to solve the problem of finite model reasoning on free $\mathcal{ALCNI}$ knowledge bases [Calvanese 1996*a*]. Again, one can construct a system of linear inequalities and relate the existence of models for the knowledge base to the existence of particular solutions of the system. However the presence of arbitrary free inclusion assertions, and the higher expressiveness of the underlying language (in particular the presence of qualified existential quantification) make the construction of the system more involved than in the previous case. Indeed, while for primitive $\mathcal{ALUNI}$ knowledge bases it is sufficient to construct a system of inequalities whose size is simply exponential in the size of the knowledge base, the extension of the technique to $\mathcal{ALCNI}$ introduced by Calvanese [1996*a*] is based on and additional "expansion" step, which introduces an additional exponential blowup.

The additional expansion step becomes necessary due to the presence of assertions of the form $L_1 \sqsubseteq \exists R.L_2$. The most direct way to handle such assertions would be to leave their treatment to the system of inequalities (similarly to what is done for number restrictions). A natural extension of the system of inequalities would be to add for each assertion of the form $L_1 \sqsubseteq \exists P.L_2$ an inequality

$$\sum_{\substack{(P,\widehat{D}_1,\widehat{D}_2)\ \mid \\ L_1\in\widehat{D}_1 \wedge L_2\in\widehat{D}_2}} \mathrm{Var}((P,\widehat{D}_1,\widehat{D}_2)) \;\geq\; \sum_{\widehat{D}\ \mid\ L_1\in\widehat{D}} \mathrm{Var}(\widehat{D}),$$

and a similar inequality for each assertion of the form $L_1 \sqsubseteq \exists P^-.L_2$. Unfortunately, imposing such conditions does not guarantee that from each acceptable nonnegative integer solution of the system a model of the knowledge base can be constructed, in which the number of instances of each compound concept and compound role is given by the value assigned by the solution to the corresponding unknown. The intuitive reason why this simple approach does not lead to the desired result for $\mathcal{ALCNI}$ knowledge bases, is that it relies on the fact that all objects that are instances of the same compound concept are characterized by the same properties. As already observed, the system of inequalities encodes local constraints on the number of connections that a single object may have, by means of global constraints on the total number of connections of a certain type. The differences on the types of connections between instances of different concepts are removed by considering compound concepts and roles. Once this is done, all instances of the same compound concept can be regarded as equivalent. The approach works for $\mathcal{ALUNI}$ knowledge bases, where no concept expression can distinguish between different instances of the same compound concept. This is no longer true if the concept expressions may contain qualified existential quantification, in which case splitting the domain into compound concepts is not sufficient to ensure that the instances have the same properties.

The problem can be dealt with by making a more fine-grained splitting of concepts and roles, which takes into account also the existence of connections of certain types. The resulting system of linear inequalities, whose acceptable solutions can be put into correspondence with finite models of the knowledge base, is doubly exponential in the size of the knowledge base (see [Calvanese 1996*a*] for details).

5.10. Theorem. *Whether a concept C is finitely consistent in an $\mathcal{ALCNI}$ knowledge base $\mathcal{K}$ can be decided in worst case deterministic double exponential time in the size of $\mathcal{K}$ plus the size of C.*

Since $\mathcal{ALCNI}$ is closed under negation, the previous result provides also an upper bound for finite logical implication.

5.11. Theorem. *Finite logical implication $\mathcal{K} \models_f C_1 \sqsubseteq C_2$ in $\mathcal{ALCNI}$ knowledge bases can be decided in worst case deterministic double exponential time in the sum of the sizes of $\mathcal{K}$, C_1, and C_2.*

The method can also be extended to knowledge bases containing qualified number restrictions, by making a separation of concepts based not only on the existence but also on the number of links of a certain type. It turns out that it is in fact sufficient to consider only intervals of numbers of links, where the ranges of these intervals are given by the numbers that effectively appear in the knowledge base. In this way it is still possible to keep the size of the resulting system of linear inequalities doubly exponential in the size of the knowledge base, and the same upper bounds as for $\mathcal{ALCNI}$ hold also for finite model reasoning over $\mathcal{ALCQI}$ knowledge bases.

An EXPTIME lower bound for finite model reasoning in $\mathcal{ALCNI}$ can be obtained by observing that e.g., $\mathcal{ALC}$ is a sublanguage of $\mathcal{ALCNI}$ which has the finite model property and for which (unrestricted and hence finite model) reasoning over a knowledge base is EXPTIME-hard. This leaves an exponential complexity gap between the known upper and lower bounds for finite model reasoning over $\mathcal{ALCNI}$ knowledge bases.

6. Beyond Basic Description Logics

Several additional constructs for building concept expressions besides those already present in $\mathcal{ALCQI}$ have been proposed in the literature. In this section we discuss the most important of these extensions, how they influence the reasoning process, and what modifications to the reasoning procedures are needed to take the additional constructs into account. Roughly speaking, the extensions for which a logic remains decidable are those for which the tree-model property is preserved [Vardi 1997], possibly after having performed a satisfiability preserving transformation of the knowledge base[11]. We remind the reader that this chapter concentrates on TBox reasoning only. Other extensions to $\mathcal{ALCQI}$ have been investigated that allow one to model and reason on individuals and ABoxes. We refer to [De Giacomo and Lenzerini 1996] for basic results on this subject.

6.1. Complex Roles

The EXPTIME-completeness result presented in Sect. 4 for unrestricted model reasoning on $\mathcal{ALCQI}$ knowledge bases has been extended to $\mathcal{ALCQI}_{reg}$, which is the DL obtained from $\mathcal{ALCQI}$ by adding the role constructs of PDLs, i.e., chaining, union, reflexive-transitive closure, and the identity role (tests in PDLs). In particular, qualified number restrictions, which are allowed only on atomic roles and inverse of atomic roles (called *basic roles* in the following), can be encoded as shown in Sect. 4.4, while reasoning in the resulting logic $\mathcal{ALCFI}_{reg}$ can be dealt with techniques based on automata on infinite trees [Calvanese 1996*c*, Calvanese, De Giacomo and Lenzerini 1999], which extend the techniques used for Converse-PDL by Vardi and Wolper [1986] and Vardi [1985].

$\mathcal{ALCQI}_{reg}$ and $\mathcal{ALCFI}_{reg}$ correspond to variants of PDLs whose computational properties had not been studied before. 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] on atomic programs and their converse. While the PDL corresponding to $\mathcal{ALCFI}_{reg}$ is an extension of Deterministic-Converse-PDL [Vardi and Wolper 1986], in which determinism of both atomic programs and their inverse is determined by the properties of the starting state.

6.2. Boolean Constructs on Roles

The role constructs of $\mathcal{ALC}\cdot_{reg}$ allow one to build roles which are regular expressions over a set of basic roles. Since regular languages are closed under intersection and complementation, the intersection of roles and the complement of a role are already expressible in $\mathcal{ALC}\cdot_{reg}$, provided that we consider them as operators on the regular languages denoted by the role expressions. However, the more common approach both in PDLs and DLs is to consider boolean operators as applied to the binary relations denoted by the roles. The logics thus obtained are more expressive than traditional PDL [Harel 1984], but they lack for example the tree model property, and reasoning in the presence of these constructs is usually harder. We notice that

[11] The logics discussed by Danecki [1984] and Lutz and Sattler [2000] are an exception, being decidable without having the tree model property.

the semantics immediately implies that intersection of roles can be expressed by means of union and complementation.

Unrestricted satisfiability in PDL augmented with intersection of programs is decidable in deterministic double exponential time [Danecki 1984], and so is satisfiability in $\mathcal{ALC}_{reg}$ augmented with intersection of 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] (Th. 2.34), and so is reasoning in $\mathcal{ALC}_{reg}$ augmented with complementation of roles. Also, Deterministic-PDL augmented with intersection of roles is highly undecidable [Harel 1985, Harel 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.

The proof of undecidability by Harel [1985] exploits a 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[12]. We sketch the idea of the proof using the terminology of DLs, instead of that of PDLs. The reduction uses two roles `right` and `up` which are globally functional (i.e., $\forall(\texttt{right} \sqcup \texttt{up})^*.(\exists^{\leq 1}\texttt{right} \sqcap \exists^{\leq 1}\texttt{up})$) and denote pairs of tiles that are adjacent in the x and y directions, respectively. By means of intersection of roles (i.e., $\forall(\texttt{right} \sqcup \texttt{up})^*.\exists((\texttt{right} \circ \texttt{up}) \sqcap (\texttt{up} \circ \texttt{right})))$, `right` and `up` are constrained to effectively define a two-dimensional grid. Notice that to achieve this, it is necessary to impose for each point of the grid, that by following $\texttt{right} \circ \texttt{up}$ one reaches the same point as by following $\texttt{up} \circ \texttt{right}$. The use of intersection of role chains turns out to be essential to force this condition. Transitive closure (i.e., $\forall(\texttt{right} \sqcup \texttt{up})^*.C$) is then exploited also to impose the required local matching constraints on all tiles of the grid.

The reduction above requires intersection to be applied to complex roles containing concatenations. The question therefore arises if decidability can be preserved if one restricts boolean operations to basic roles. This is indeed the case if complementation of basic roles is used only to express difference of roles. In fact, decidability in deterministic exponential time of unrestricted model reasoning holds for full $\mathcal{ALCQI}_{reg}$ extended with intersection and difference of basic roles, with the additional limitation that intersection or difference between an atomic role and an inverse atomic role may not be used [De Giacomo and Lenzerini 1995*b*, De Giacomo 1995]. The proof is based on the reification of roles (see Sect. 4.4.2) and exploits the possibility to impose boolean conditions on the objects that represent the reified instances of a role. Intersection of basic roles can also be expressed in the $\mathcal{SHIQ}$ DL [Horrocks and Sattler 1999], which extends $\mathcal{ALCQI}$ with transitive roles and role hierarchies.

[12]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.

Using a direct encoding into nonemptiness of automata on infinite trees, Lutz and Sattler [2000] show that $\mathcal{ALC}$ extended with full boolean operators on atomic roles (which are the only roles in the logic) is EXPTIME-complete. Observe that, by means of role negation, axioms can be internalized: the concept $\forall P.(C \Rightarrow C') \sqcap \forall \neg P.(C \Rightarrow C')$ corresponds to the assertion $C \sqsubseteq C'$.

Reification of roles can also be exploited to show decidability of finite model reasoning in knowledge bases in $\mathcal{ALCQI}$ augmented with intersection and difference of basic roles, with the same limitations as above. To this end qualified number restrictions are needed in order to express even unqualified number restrictions on the reified roles. For example, the concept $\exists^{\leq 2}\texttt{edge}$ becomes, after reifying the role $\texttt{edge}$, $\exists^{\leq 2} V_1^{-}.A_{\texttt{edge}}$.

6.3. Role Value Maps

Another construct that stems from frame-systems and that provides additional useful means to specify structural properties of concepts is the so called *role value map* [Brachman and Schmolze 1985]. An object o is an instance of a role value map $(R_1 = R_2)$ if the set of objects that are connected to o via role R_1 equals the set of objects connected to o via role R_2. A generalization of role value map is *role value inclusion*, denoted $(R_1 \subseteq R_2)$, whose semantics is defined analogously to that of role value map, using set inclusion instead of set equality. Using these constructs one can denote, for example, by means of $(\texttt{owns} \subseteq \texttt{lives\_in} \circ \texttt{made\_in}^{-})$ the set of all persons that own only products manufactured in the country they live in.

When role value map is added, the logic looses the tree model property, and this construct leads immediately to undecidability of reasoning. For $\mathcal{ALC}_{reg}$ this can be shown by a reduction from the tiling problem in a similar way as to what is done by Harel [1985] for Deterministic-PDL with intersection of roles. In this case, role value map (i.e., $\forall(\texttt{right} \sqcup \texttt{up})^*.(\texttt{right} \circ \texttt{up} = \texttt{up} \circ \texttt{right})$) is used instead of role intersection to define the constraints on the grid. Undecidability holds however already for concept subsumption (with respect to the empty knowledge base) in $\mathcal{AL}$ augmented with role value map, where the involved roles are concatenations of atomic roles [Schmidt-Schauß 1989]. This is shown by a reduction from the word problem for semigroups, which is undecidable [Boone 1959].

Again, in order to show undecidability it is necessary to apply role value map to concatenations of roles. Indeed, if the application of role value map 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 basic roles, by observing that $(R_1 \subseteq R_2)$ 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 by 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.

Another way to take role value map into account without loosing decidability is to restrict the use of such construct to functional roles, as usually done in the area of Feature Logics [Carpenter 1992]. Finally, we note that the role value map construct is also related to the notion of *path constraints* in database modeling [Buneman, Fan and Weinstein 1998].

6.4. Number Restrictions on Complex Roles

In $\mathcal{ALCQI}_{reg}$, the use of (qualified) number restrictions is limited to basic roles, which guarantees that the logic has the tree model property. This property is lost, together with decidability, if number 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., $\exists^{\leq 1}(\texttt{right} \circ \texttt{up}) \sqcup (\texttt{up} \circ \texttt{right})$) is used instead of role intersection to define the grid.

Again, if one limits number restrictions to basic roles, as in $\mathcal{ALCQI}_{reg}$, then reasoning remains decidable. By exploiting reification of roles, De Giacomo and Lenzerini [1995*b*] show that decidability is not lost if one can impose number restrictions on roles that are composed by applying union, intersection, and difference to basic roles, with the same proviso as above, that direct and inverse roles may not appear together in such a boolean combination (see also [De Giacomo 1995] for details).

Another example of a decidable logic that does not have the tree model property is obtained by allowing the use of concatenation (but not transitive closure) inside number restrictions. Let us denote with $\mathcal{N}(X)$, where X is a subset of $\{\circ, \sqcup, \sqcap, ^-\}$, unqualified number restrictions applied to roles that are obtained by applying to atomic roles the role constructs in X. As shown by Baader and Sattler [1996, 1999], concept satisfiability in $\mathcal{ALCN}(\circ, \sqcap, \sqcup)$ is decidable in deterministic exponential time if intersection and union are restricted to *role chains* (i.e., concatenations of atomic roles) of the same length. Notice that, by the considerations above, decidability holds only for reasoning on concept expressions and is lost if one considers reasoning with respect to a knowledge base (or alternatively adds reflexive-transitive closure of roles). However, reasoning even with respect to the empty knowledge base is undecidable if one adds to $\mathcal{ALCN}$ number restrictions on more complex roles. In particular, this holds for $\mathcal{ALCN}(\circ, \sqcap)$ (if no constraints on the lengths of the chains are imposed) and for $\mathcal{ALCN}(\circ, \sqcup, ^-)$ [Baader and Sattler 1996, Baader and Sattler 1999]. The reductions again exploit the tiling problem, and make use of number restrictions on complex roles to simulate a universal role that is used for imposing conditions that hold for all points of the grid. Using a more complex reduction, Baader and Sattler [1999] show also that concept consistency in $\mathcal{ALCN}(\circ)$ (or concept satisfiability in $\mathcal{ALC}_{reg}\mathcal{N}(\circ)$) is undecidable.

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. In particular, it is not known whether concept satisfiability in $\mathcal{ALCN}(\circ,^-)$, $\mathcal{ALC}_{reg}\mathcal{N}(\sqcap)$, or $\mathcal{ALC}_{reg}\mathcal{N}(\sqcup)$ is decidable [Baader and Sattler 1999]. Notice that decidability of knowledge base reasoning in $\mathcal{ALCN}(\sqcap,\sqcup,^-)$ follows from the decidability of $\mathcal{C}2$, i.e., first order logic with two variables and counting quantifiers [Grädel, Otto and Rosen 1997, Grädel and Otto 1999].

6.5. Relations of Arbitrary Arity

DLs allow one to define concepts which denote classes of objects with common properties. These properties are defined by establishing relationships to other objects by means of roles. In traditional DLs, roles denote binary relations only, while some real world situations would be modeled more naturally by making use of relations of arity greater that two. Extensions of DLs with relations of arbitrary arity, which are interpreted as tuples of objects of the domain of interpretation, have been proposed in [Schmolze 1989, Catarci and Lenzerini 1993, De Giacomo and Lenzerini 1994*b*, De Giacomo 1995]. In order to use these relations for the specification of concepts, the notion of role is generalized to include also projections of relations on two of their components.

By exploiting reification (cfr. Sect. 4.4.2), reasoning in the presence of relations can be reduced to reasoning on ordinary DLs. A relation is reified by introducing a new concept and as many roles as the arity of the relation. A tuple of the relation is represented in the reified counterpart of the knowledge base by an instance of the corresponding concept, which is linked through each of the associated roles to an object representing the component of the tuple. Performing the reification requires however some attention, since, according to the standard semantics, in a relation there may not be two equal tuples (i.e., constituted by the same components in the same positions) in its extension. In the reified counterpart, on the other hand, one cannot explicitly rule out (e.g., by using specific assertions) that there are two objects o_1 and o_2 "representing" the same tuple, i.e., that are connected to exactly the same objects denoting the components of the tuple. A model of the reified counterpart of a knowledge base in which this situation occurs may not correspond directly to a model of the original knowledge base containing relations, since the two equivalent objects in general cannot be collapsed into a single object (representing the tuple) without violating assertions (e.g., cardinality constraints). However, a result analogous to Lem. 4.8 holds for relations of arbitrary arity. Therefore one does not need to take this constraint explicitly into account when reasoning on the reified counterpart of a knowledge base with relations.

By exploiting this idea, Calvanese, De Giacomo and Lenzerini [1997] introduce the DL $\mathcal{DLR}$, which extends $\mathcal{ALCQI}$ by boolean combinations of n-ary relations, and show that reasoning over $\mathcal{DLR}$ knowledge bases is again EXPTIME-decidable. Calvanese, De Giacomo and Lenzerini [1998*a*] propose a mechanism for specifying (unions of) conjunctive queries in $\mathcal{DLR}$, and present techniques for checking query containment. The problem of answering conjunctive queries using views over a $\mathcal{DLR}$ knowledge base is addressed by Calvanese, De Giacomo and Lenzerini [2000].

A method to reason with respect to finite models in the presence of relations of arbitrary arity is presented by Calvanese and Lenzerini [1994*b*, 1994*a*] in the context of a semantic and an object-oriented database model, respectively. The reasoning procedure, which represents a direct generalization of the one proposed in Sect. 5.2 to relations of arbitrary arity, does not exploit reification to handle relations but encodes directly the constraints on them into a system of linear inequalities.

6.6. Structured Objects, Well Foundedness, and Fixpoints

An alternative way to overcome the limitations that result from the restriction to unary and binary relationships, is to consider the interpretation domain as being constituted by objects with a complex structure, and extend the DLs with constructs that allow one to specify such structure [De Giacomo and Lenzerini 1995*b*]. This approach is in the spirit of object-oriented data models [Lecluse and Richard 1989, Bancilhon and Khoshafian 1989, Hull 1988, Bergamaschi and Nebel 1994]. In contrast with the idea of introducing n-ary relations, all aspects of the domain to be modeled can be represented in a uniform way, namely 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.

Reasoning in the presence of constructs for specifying structured objects is based on a reduction to reasoning in traditional DLs by exploiting reification of tuples and sets, and thus can be done in deterministic exponential time [De Giacomo and Lenzerini 1995*a*].

A limitation of the above approach is that there is no way to limit the depth of the structure of an object, and thus one cannot rule out non well-founded structures, such as a set that has itself as one of its elements, or a tuple that refers to itself via one of its components. To overcome this problem, Calvanese et al. [1995] and Calvanese [1996*a*] propose a *well-founded* construct $wf(R)$, which allows one to state that certain substructures of a model have to be finite. Formally $wf(R)$ is interpreted as the set of those objects which are not the starting point of an infinite sequence of objects, each connected to the next by means of role R. Thus it corresponds directly to the negation of a *repeat formula* in ΔPDL [Harel and Sherman 1982, Streett 1982]. By means of the well-founded construct one can express not only that the instances of a certain concept have a structure of finite depth, but also define inductive structures such as lists, finite trees, or directed acyclic graphs [Calvanese et al. 1995]. The technique to reason in $\mathcal{ALCQI}_{reg}$ extended with the well-founded construct is based on a reduction by means of reification to reasoning in ΔPDL extended with local functionality on direct and inverse programs, which is dealt with using automata-theoretic techniques [Calvanese 1996*c*].

Formulas including the well-founded construct can be considered as particular forms of fixpoint formulas. Fixpoints incorporated directly in the semantics of DLs have been first studied by Nebel [1991] and Baader [1996] for simple DLs. Fixpoints on concepts in their full generality have been investigated by Schild [1994] and De Giacomo and Lenzerini [1997]. The logics studied in these articles include constructs to define concepts which are least or greatest fixpoints of concept-equations. Decidability in deterministic exponential time for variants of these logics which include number restrictions has been established by exploiting a correspondence with the propositional μ-calculus [Kozen 1983, Streett and Emerson 1989]. Finally, Calvanese, De Giacomo and Lenzerini [1999] present the logic $\mathcal{ALCQI}_\mu$ which extends $\mathcal{ALCQI}_{reg}$ with the most general form of fixpoint on concepts. It also introduces $\mathcal{DLR}_\mu$, which in addition includes n-ary relations. Reasoning in $\mathcal{ALCQI}_\mu$ and in $\mathcal{DLR}_\mu$ is proved to be EXPTIME-decidable by a reduction to nonemptiness of two-way alternating automata on infinite trees [Vardi 1998].

7. Conclusions

DLs have been thoroughly investigated in the last fifteen years, especially with the goal of devising logic-based Knowledge Representation formalisms with a good compromise between expressive power and complexity of reasoning. Although the first studies on DLs concentrated on logics with limited sets of constructs, and tractable reasoning procedures, recent investigations on the features required in applications show the need for expressive DLs with decidable reasoning tasks. In this chapter, we have illustrated the main ideas that lead to the development of reasoning techniques for expressive DLs. Most of the described techniques rely on the correspondence between expressive DLs and Propositional Dynamic Logics. Such a correspondence does not allow one to directly derive sound and complete reasoning procedures for DLs. Indeed, we have illustrated several sophisticated techniques for reducing reasoning in expressive DLs to reasoning in Propositional Dynamic Logics. Also, we have presented specialized techniques for finite model reasoning in expressive DLs.

Besides being of interest from a scientific point of view, the decidability results described in this chapter have stimulated the investigation of implemented systems that are able to reason on expressive DLs. Interesting results on this aspect are reported by Horrocks and Sattler [1999], Horrocks et al. [1999], and Horrocks and Patel-Schneider [1999].

Bibliography

Artale A. and Franconi E. [1994], A computational account for a description logic of time and action, *in* J. Doyle, E. Sandewall and P. Torasso, eds, 'Proc. of the 4th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'94)', Morgan Kaufmann, Los Altos, Bonn (Germany), pp. 3–14.

Baader F. [1991], Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles, *in* 'Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)', Sydney (Australia).

Baader F. [1996], 'Using automata theory for characterizing the semantics of terminological cycles', *Ann. of Mathematics and Artificial Intelligence* **18**, 175–219.

Baader F. and Hollunder B. [1995], 'Embedding defaults into terminological knowledge representation formalisms', *J. of Automated Reasoning* **14**, 149–180.

Baader F., Hollunder B., Nebel B., Profitlich H.-J. and Franconi E. [1992], An empirical analysis of optimization techniques for terminological representation systems, *in* 'Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'92)', Morgan Kaufmann, Los Altos, pp. 270–281.

Baader F. and Sattler U. [1996], Number restrictions on complex roles in description logics: A preliminary report, *in* 'Proc. of the 5th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'96)', pp. 328–338.

Baader F. and Sattler U. [1999], 'Expressive number restrictions in description logics', *J. of Logic and Computation* **9**(3), 319–350.

Baader F. and Sattler U. [2000], Tableau algorithms for description logics, *in* R. Dyckhoff, ed., 'Proc. of the 4th Int. Conf. on Analytic Tableaux and Related Methods (TABLEAUX 2000)', Vol. 1847 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, pp. 1–18.

Bancilhon F. and Khoshafian S. [1989], 'A calculus for complex objects', *J. of Computer and System Sciences* **38**(2), 326–340.

Ben-Ari M., Halpern J. Y. and Pnueli A. [1982], 'Deterministic propositional dynamic logic: Finite models, complexity, and completeness', *J. of Computer and System Sciences* **25**, 402–417.

Bergamaschi S. and Nebel B. [1994], 'Acquisition and validation of complex object database schemata supporting multiple inheritance', *Applied Intelligence* **4**(2), 185–203.

Bergamaschi S. and Sartori C. [1992], 'On taxonomic reasoning in conceptual design', *ACM Trans. on Database Systems* **17**(3), 385–422.

Berger R. [1966], 'The undecidability of the dominoe problem', *Mem. Amer. Math. Soc.* **66**, 1–72.

Blackburn P., de Rijke M. and Venema Y. [2000], Modal logic. To appear.

Blanco J. L., Illarramendi A. and Goñi A. [1994], 'Building a federated relational database system: An approach using a knowledge-based system', *J. of Intelligent and Cooperative Information Systems* **3**(4), 415–455.

Boone W. W. [1959], 'The word problem', *Ann. of Mathematics* **2**(70), 207–265.

Borgida A. [1995], 'Description logics in data management', *IEEE Trans. on Knowledge and Data Engineering* **7**(5), 671–682.

Borgida A. and Patel-Schneider P. F. [1994], 'A semantics and complete algorithm for subsumption in the CLASSIC description logic', *J. of Artificial Intelligence Research* **1**, 277–308.

Brachman R. J. and Levesque H. J. [1984], The tractability of subsumption in frame-based description languages, *in* 'Proc. of the 4th Nat. Conf. on Artificial Intelligence (AAAI'84)', pp. 34–37.

Brachman R. J. and Schmolze J. G. [1985], 'An overview of the KL-ONE knowledge representation system', *Cognitive Science* **9**(2), 171–216.

Buchheit M., Donini F. M., Nutt W. and Schaerf A. [1994], Terminological systems revisited: Terminology = schema + views, *in* 'Proc. of the 12th Nat. Conf. on Artificial Intelligence (AAAI'94)', Seattle (USA), pp. 199–204.

Buchheit M., Donini F. M. and Schaerf A. [1993], 'Decidable reasoning in terminological knowledge representation systems', *J. of Artificial Intelligence Research* **1**, 109–138.

Buneman P., Fan W. and Weinstein S. [1998], Path constraints on semistructured and structured data, *in* 'Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)', pp. 129–138.

Calvanese D. [1996*a*], Finite model reasoning in description logics, *in* L. C. Aiello, J. Doyle and S. C. Shapiro, eds, 'Proc. of the 5th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'96)', Morgan Kaufmann, Los Altos, pp. 292–303.

Calvanese D. [1996*b*], Reasoning with inclusion axioms in description logics: Algorithms and complexity, *in* W. Wahlster, ed., 'Proc. of the 12th Eur. Conf. on Artificial Intelligence (ECAI'96)', John Wiley & Sons, pp. 303–307.

Calvanese D. [1996*c*], Unrestricted and Finite Model Reasoning in Class-Based Representation Formalisms, PhD thesis, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza". Available at http://www.dis.uniroma1.it/pub/calvanes/thesis.ps.gz.

Calvanese D., De Giacomo G. and Lenzerini M. [1995], Structured objects: Modeling and reasoning, *in* 'Proc. of the 4th Int. Conf. on Deductive and Object-Oriented Databases (DOOD'95)', Vol. 1013 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 229–246.

Calvanese D., De Giacomo G. and Lenzerini M. [1997], Conjunctive query containment in Description Logics with n-ary relations, *in* 'Proc. of the 1997 Description Logic Workshop (DL'97)', pp. 5–9.

Calvanese D., De Giacomo G. and Lenzerini M. [1998*a*], On the decidability of query containment under constraints, *in* 'Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)', pp. 149–158.

Calvanese D., De Giacomo G. and Lenzerini M. [1998*b*], What can knowledge representation do for semi-structured data?, *in* 'Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI'98)', pp. 205–210.

Calvanese D., De Giacomo G. and Lenzerini M. [1999], Reasoning in expressive description logics with fixpoints based on automata on infinite trees, *in* 'Proc. of the 16th Int. Joint Conf. on Artificial Intelligence (IJCAI'99)', pp. 84–89.

Calvanese D., De Giacomo G. and Lenzerini M. [2000], Answering queries using views over description logics knowledge bases, *in* 'Proc. of the 17th Nat. Conf. on Artificial Intelligence (AAAI 2000)', pp. 386–391.

Calvanese D. and Lenzerini M. [1994*a*], Making object-oriented schemas more expressive, *in* 'Proc. of the 13th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'94)', ACM Press and Addison Wesley, Minneapolis (Minnesota, USA), pp. 243–254.

Calvanese D. and Lenzerini M. [1994*b*], On the interaction between ISA and cardinality constraints, *in* 'Proc. of the 10th IEEE Int. Conf. on Data Engineering (ICDE'94)', IEEE Computer Society Press, Houston (Texas, USA), pp. 204–213.

Calvanese D., Lenzerini M. and Nardi D. [1994], A unified framework for class based representation formalisms, *in* J. Doyle, E. Sandewall and P. Torasso, eds, 'Proc. of the 4th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'94)', Morgan Kaufmann, Los Altos, Bonn (Germany), pp. 109–120.

Calvanese D., Lenzerini M. and Nardi D. [1999], 'Unifying class-based representation formalisms', *J. of Artificial Intelligence Research* **11**, 199–240.

Carpenter B. [1992], *The Logic of Typed Feature Structures*, Cambridge University Press.

Catarci T. and Lenzerini M. [1993], 'Representing and using interschema knowledge in cooperative information systems', *J. of Intelligent and Cooperative Information Systems* **2**(4), 375–398.

Chen P. P. [1976], 'The Entity-Relationship model: Toward a unified view of data', *ACM Trans. on Database Systems* **1**(1), 9–36.

Clarke E. and Schlingloff H. [2001], Model checking, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. II, Elsevier Science, chapter 24, pp. 1635–1790.

Cosmadakis S. S., Kanellakis P. C. and Vardi M. [1990], 'Polynomial-time implication problems for unary inclusion dependencies', *J. of the ACM* **37**(1), 15–46.

Danecki R. [1984], Nondeterministic Propositional Dynamic Logic with intersection is decidable, *in* 'Proc. of the 5th Symp. on Computation Theory', Vol. 208 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 34–53.

De Giacomo G. [1995], Decidability of Class-Based Knowledge Representation Formalisms, PhD thesis, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza".

De Giacomo G. [1996], 'Eliminating "converse" from Converse PDL', *J. of Logic, Language and Information* **5**, 193–208.

De Giacomo G. and Lenzerini M. [1994*a*], Boosting the correspondence between description logics and propositional dynamic logics, *in* 'Proc. of the 12th Nat. Conf. on Artificial Intelligence (AAAI'94)', AAAI Press/The MIT Press, pp. 205–212.

De Giacomo G. and Lenzerini M. [1994*b*], Description logics with inverse roles, functional restrictions, and n-ary relations, *in* 'Proc. of the 4th Eur. Workshop on Logics in Artificial Intelligence (JELIA'94)', Vol. 838 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, pp. 332–346.

De Giacomo G. and Lenzerini M. [1995*a*], Enhanced propositional dynamic logic for reasoning about concurrent actions (extended abstract), *in* 'Working Notes of the AAAI 1995 Spring Symposium on Extending Theories of Action: Formal and Practical Applications', pp. 62–67.

De Giacomo G. and Lenzerini M. [1995*b*], What's in an aggregate: Foundations for description logics with tuples and sets, *in* 'Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI'95)', pp. 801–807.

De Giacomo G. and Lenzerini M. [1996], TBox and ABox reasoning in expressive description logics, *in* L. C. Aiello, J. Doyle and S. C. Shapiro, eds, 'Proc. of the 5th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'96)', Morgan Kaufmann, Los Altos, pp. 316–327.

De Giacomo G. and Lenzerini M. [1997], 'A uniform framework for concept definitions in description logics', *J. of Artificial Intelligence Research* **6**, 87–110.

De Giacomo G. and Massacci F. [2000], 'Combining deduction and model checking into tableaux and algorithms for converse-pdl', *Information and Computation* **160**(1–2).

Devambu P., Brachman R. J., Selfridge P. J. and Ballard B. W. [1991], 'LASSIE: A knowledge-based software information system', *Communications of the ACM* **34**(5), 36–49.

Donini F. M., Lenzerini M., Nardi D. and Nutt W. [1991], Tractable concept languages, *in* 'Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)', Sydney (Australia), pp. 458–463.

Donini F. M., Lenzerini M., Nardi D. and Nutt W. [1997], 'The complexity of concept languages', *Information and Computation* **134**, 1–58.

Donini F. M., Lenzerini M., Nardi D., Nutt W. and Schaerf A. [1992], Adding epistemic operators to concept languages, *in* 'Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'92)', Morgan Kaufmann, Los Altos, pp. 342–353.

Donini F. M., Lenzerini M., Nardi D. and Schaerf A. [1994], 'Deduction in concept languages: From subsumption to instance checking', *J. of Logic and Computation* **4**(4), 423–452.

Donini F. M., Lenzerini M., Nardi D. and Schaerf A. [1996], Reasoning in description logics, *in* G. Brewka, ed., 'Principles of Knowledge Representation', Studies in Logic, Language and Information, CSLI Publications, pp. 193–238.

Doyle J. and Patil R. S. [1991], 'Two theses of knowledge representation: Language restrictions, taxonomic classification, and the utility of representation services', *Artificial Intelligence* **48**, 261–297.

Ebbinghaus H.-D. and Flum J. [1999], *Finite Model Theory*, second edition edn, Springer-Verlag.

Fattorosi-Barnaba M. and De Caro F. [1985], 'Graded modalities I', *Studia Logica* **44**, 197–221.

Fine K. [1972], 'In so many possible worlds', *Notre Dame J. of Formal Logic* **13**(4), 516–520.

Fischer M. J. and Ladner R. E. [1979], 'Propositional dynamic logic of regular programs', *J. of Computer and System Sciences* **18**, 194–211.

Goldblatt R. [1992], *Logics of Time and Computation*, Vol. 7 of *Lecture Notes*, second edn, Center for the Study of Language and Information.

Grädel E. and Otto M. [1999], 'On logics with two variables', *Theoretical Computer Science* **224**, 73–113.

Grädel E., Otto M. and Rosen E. [1997], Two-variable logic with counting is decidable, *in* 'Proc. of the 12th IEEE Symp. on Logic in Computer Science (LICS'97)', pp. 306–317.

Hanschke P. [1992], Specifying role interaction in concept languages, *in* 'Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'92)', Morgan Kaufmann, Los Altos, pp. 318–329.

Harel D. [1984], Dynamic logic, *in* 'Handbook of Philosophical Logic', Vol. 2, D. Reidel, Dordrecht (Holland), pp. 497–640.

Harel D. [1985], 'Recurring dominoes: Making the highly undecidable highly understandable', *Ann. of Discrete Mathematics* **24**, 51–72.

Harel D. [1986], 'Effective transformations of infinite trees, with applications to high undecidability, dominoes, and fairness', *J. of the ACM* **33**(1), 224–248.

Harel D. and Sherman R. [1982], 'Looping vs. repeating in dynamic logic', *Information and Computation* **55**, 175–192.

Hollunder B. [1996], 'Consistency checking reduced to satisfiability of concepts in terminological systems', *Ann. of Mathematics and Artificial Intelligence* **18**(2–4), 133–157.

Hollunder B. and Baader F. [1991], Qualifying number restrictions in concept languages, Technical Report RR-91-03, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Kaiserslautern (Germany). An abridged version appeared in *Proc. of the 2nd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'91)*.

Horrocks I. [1998], Using an expressive description logic: FaCT or fiction?, *in* 'Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98)', pp. 636–647.

Horrocks I. and Patel-Schneider P. F. [1999], 'Optimizing description logic subsumption', *J. of Logic and Computation* **9**(3), 267–293.

Horrocks I. and Sattler U. [1999], 'A description logic with transitive and inverse roles and role hierarchies', *J. of Logic and Computation* **9**(3), 385–410.

Horrocks I., Sattler U. and Tobies S. [1999], Practical reasoning for expressive description logics, *in* H. Ganzinger, D. McAllester and A. Voronkov, eds, 'Proc. of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99)', number 1705 *in* 'Lecture Notes in Artificial Intelligence', Springer-Verlag, pp. 161–180.

Hull R. [1988], A survey of theoretical research on typed complex database objects, *in* J. Paredaens, ed., 'Databases', Academic Press, pp. 193–256.

Kozen D. [1983], 'Results on the propositional μ-calculus', *Theoretical Computer Science* **27**, 333–354.

Kozen D. and Tiuryn J. [1990], Logics of programs, *in* J. van Leeuwen, ed., 'Handbook of Theoretical Computer Science — Formal Models and Semantics', Elsevier Science Publishers (North-Holland), Amsterdam, pp. 789–840.

Kracht M. and Wolter F. [2000], 'Normal modal logics can simulate all others', *J. of Symbolic Logic* . To appear.

Lecluse C. and Richard P. [1989], Modeling complex structures in object-oriented databases, *in* 'Proc. of the 8th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'89)', pp. 362–369.

Lenzerini M. and Nobili P. [1990], 'On the satisfiability of dependency constraints in entity-relationship schemata', *Information Systems* **15**(4), 453–461.

Levesque H. J. and Brachman R. J. [1987], 'Expressiveness and tractability in knowledge representation and reasoning', *Computational Intelligence* **3**, 78–93.

Levy A. Y., Rajaraman A. and Ordille J. J. [1996], Query answering algorithms for information agents, *in* 'Proc. of the 13th Nat. Conf. on Artificial Intelligence (AAAI'96)', pp. 40–47.

Lutz C. and Sattler U. [2000], Mary likes all cats, *in* F. Baader and U. Sattler, eds, 'Proc. of the 2000 Description Logic Workshop (DL 2000)', CEUR Electronic Workshop Proceedings http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-33/, pp. 213–226.

Nebel B. [1988], 'Computational complexity of terminological reasoning in BACK', *Artificial Intelligence* **34**(3), 371–383.

NEBEL B. [1990], 'Terminological reasoning is inherently intractable', *Artificial Intelligence* **43**, 235–249.

NEBEL B. [1991], Terminological cycles: Semantics and computational properties, *in* J. F. Sowa, ed., 'Principles of Semantic Networks', Morgan Kaufmann, Los Altos, pp. 331–361.

OHLBACH H., NONNENGART A., DE RIJKE M. AND GABBAY D. [2001], Encoding two-valued nonclassical logics in classical logic, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. II, Elsevier Science, chapter 21, pp. 1403–1486.

PAPADIMITRIOU C. H. [1981], 'On the complexity of integer programming', *J. of the ACM* **28**(4), 765–768.

PARIKH R. [1981], Propositional dynamic logic of programs: A survey, *in* 'Proc. of the 1st Workshop on Logics of Programs', Vol. 125 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 102–144.

PATEL-SCHNEIDER P. F. [1989], 'A four-valued semantics for terminological logic', *Artificial Intelligence* **38**(1), 319–351.

PRATT V. R. [1979], Models of program logic, *in* 'Proc. of the 20th Annual Symp. on the Foundations of Computer Science (FOCS'79)', pp. 115–122.

PRATT V. R. [1980], 'A near-optimal method for reasoning about action', *J. of Computer and System Sciences* **20**, 231–255.

ROBINSON R. [1971], 'Undecidability and nonperiodicity of tilings on the plane', *Inventiones Math.* **12**, 177–209.

SCHAERF A. [1994], 'Reasoning with individuals in concept languages', *Data and Knowledge Engineering* **13**(2), 141–176.

SCHILD K. [1991], A correspondence theory for terminological logics: Preliminary report, *in* 'Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)', Sydney (Australia), pp. 466–471.

SCHILD K. [1994], Terminological cycles and the propositional μ-calculus, *in* J. Doyle, E. Sandewall and P. Torasso, eds, 'Proc. of the 4th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'94)', Morgan Kaufmann, Los Altos, Bonn (Germany), pp. 509–520.

SCHMIDT-SCHAUSS M. [1989], Subsumption in KL-ONE is undecidable, *in* R. J. Brachman, H. J. Levesque and R. Reiter, eds, 'Proc. of the 1st Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'89)', Morgan Kaufmann, Los Altos, pp. 421–431.

SCHMIDT-SCHAUSS M. AND SMOLKA G. [1991], 'Attributive concept descriptions with complements', *Artificial Intelligence* **48**(1), 1–26.

SCHMOLZE J. G. [1989], Terminological knowledge representation systems supporting n-ary terms, *in* 'Proc. of the 1st Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'89)', pp. 432–443.

SHETH A. P., GALA S. K. AND NAVATHE S. B. [1993], 'On automatic reasoning for schema integration', *J. of Intelligent and Cooperative Information Systems* **2**(1), 23–50.

STREETT R. S. [1982], 'Propositional dynamic logic of looping and converse is elementarily decidable', *Information and Computation* **54**, 121–141.

STREETT R. S. AND EMERSON E. A. [1989], 'An automata theoretic decision procedure for the propositional μ-calculus', *Information and Computation* **81**, 249–264.

TOBIES S. [1999*a*], On the complexity of counting in description logics, *in* 'Proc. of the 1999 Description Logic Workshop (DL'99)', CEUR Electronic Workshop Proceedings http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-22/, pp. 105–109.

TOBIES S. [1999*b*], A PSPACE algorithm for graded modal logic, *in* H. Ganzinger, ed., 'Proc. of the 16th Int. Conf. on Automated Deduction (CADE'99)', Vol. 1632 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, pp. 52–66.

ULLMAN J. D. [1997], Information integration using logical views, *in* 'Proc. of the 6th Int. Conf. on Database Theory (ICDT'97)', Vol. 1186 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 19–40.

Van der Hoek W. [1992], 'On the semantics of graded modalities', *J. of Applied Non-Classical Logics* **2**(1), 81–123.

Van der Hoek W. and de Rijke M. [1995], 'Counting objects', *J. of Logic and Computation* **5**(3), 325–345.

Vardi M. Y. [1985], The taming of converse: Reasoning about two-way computations, *in* R. Parikh, ed., 'Proc. of the 4th Workshop on Logics of Programs', Vol. 193 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 413–424.

Vardi M. Y. [1997], Why is modal logic so robustly decidable, *in* 'DIMACS Series in Discrete Mathematics and Theoretical Computer Science', Vol. 31, American Mathematical Society, pp. 149–184.

Vardi M. Y. [1998], Reasoning about the past with two-way automata, *in* 'Proc. of the 25th Int. Coll. on Automata, Languages and Programming (ICALP'98)', Vol. 1443 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 628–641.

Vardi M. Y. and Wolper P. [1984], Automata-theoretic techniques for modal logics of programs, *in* 'Proc. of the 16th ACM SIGACT Symp. on Theory of Computing (STOC'84)', pp. 446–455.

Vardi M. Y. and Wolper P. [1986], 'Automata-theoretic techniques for modal logics of programs', *J. of Computer and System Sciences* **32**, 183–221.

Weida R. and Litman D. [1992], Terminological reasoning with constraint networks and an application to plan recognition, *in* 'Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'92)', Morgan Kaufmann, Los Altos, pp. 282–293.

Woods W. A. and Schmolze J. G. [1992], The KL-ONE family, *in* F. W. Lehmann, ed., 'Semantic Networks in Artificial Intelligence', Pergamon Press, pp. 133–178. Published as a special issue of *Computers & Mathematics with Applications*, Volume 23, Number 2–9.

Index