What's in an Aggregate: Foundations for Description Logics with Tuples and Sets

Giuseppe De Giacomo and Maurizio Lenzerini
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italia
{degiacomo,lenzerini}@assi.dis.uniroma1.it

Abstract

Based on the research done in the last decade, attempts have been made to propose description logics as unifying formalisms for the various class-based representation languages used in different areas. These attempts have made apparent that sound, complete, and decidable description logics still suffer from several limitations, regarding modeling classes of aggregate objects, expressing general inclusion axioms, and the ability of navigating links between classes. In this paper we make description logics accomplish the necessary leap in order to become suitable for the new challenging applications they are faced with. In particular, we propose a powerful description logic overcoming the above limitations and we show that its reasoning tasks are decidable in worst case exponential time.

1 Introduction

Description logics are AI formalisms that allow one to represent domain knowledge by focusing on classes of objects [Brachman, 1977] and their relationships [Woods, 1975], and by offering specialized inferences on the class structure.

The research developed in the last decade offers a quite complete picture of several issues related to the expressive power of the logics and the computational complexity of the reasoning tasks (see [Woods and Schmolze, 1992]). Based on the outcome of this research, attempts have been made to propose description logics as unifying formalisms for the various class-based representation languages used in different areas, such as semantic networks, feature logics, conceptual and object-oriented database models, type systems, and other formalisms used in software engineering [Bergamaschi and Sartori, 1992; Piza *et al.*, 1992; Borgida, 1992; Calvanese *et al.*, 1994; Schreiber *et al.*, 1993]. However, these attempts have made apparent that description logics that are equipped with sound, complete, and terminating reasoning procedures still suffer from several limitations that are not acceptable when representing complex domains in the different fields mentioned above. Here is a list of the most important limitations.

• The domain of interpretation is flat, in the sense that the logics consider the world as constituted by elementary objects (grouped in concepts) and binary relations between them. One consequence of this property is that N-ary relations are not supported (an exception is the logic proposed in [Schmolze, 1989], for which no complete decision procedure was proposed). In fact, N-ary relations have been shown to be important in several contexts (see [Catarci and Lenzerini, 1993]), especially in databases and in natural language. For example, exam is correctly modeled as a ternary relation over student, professor and course. Note that supporting N-ary relations means that the logic offers suitable mechanisms for their definition and their characterization. For example, one has to ensure that no pair of instances of exam exist connecting the same triple of objects; also, one may want to assert that students linked to graduate courses by the relation exam are graduate students. These kinds of properties cannot be represented by simply modeling the N-ary relation in terms of N binary relations.

• Usually, general inclusion axioms are not supported. Although inclusion axioms are essential when we want to assert properties of classes and relations, as required in complex domains, most of the research on description logics either deals with class descriptions only, or impose severe restrictions, such as acyclicity, on axioms. Exceptions are, for example, [Nebel, 1991; Baader, 1991; Schild, 1991; De Giacomo and Lenzerini, 1994; Buchheit *et al.*, 1993]. An important outcome of this research is that reasoning with axioms is computationally hard, even for the simplest description logics (weaker than $\mathcal{FL}^-$ [Woods and Schmolze, 1992]). All these works, however, limit their attention to axioms on concepts, and do not consider the problem of expressing inclusion axioms on relations.

• Relationships between classes are generally described by means of poor representation mechanisms. Indeed, when trying to use description logics for capturing representation formalisms used in different fields, one realizes that three features are essential: the ability of *navigating* relationships (say of a semantic network or of an entity-relationship schema) in both directions. The ability of stating cardinality constraints of general forms on relationships. The possibility of conceiving a relationship like a set, thus applying set theoretic operators on

them (including the notorious role value map [Woods and Schmolze, 1992]).

The aim of the present work is to devise a description logic, called $\mathcal{CATS}$, that finally addresses the above issues. The basic ingredients of $\mathcal{CATS}$ are classes and links. Differently from traditional description logics, *classes* are abstractions not only for a set of individuals (corresponding to the usual notion of concept, called simple class here), but also for sets that have aggregates as instances (called aggregate classes). There are two types of aggregates: property aggregates and instance aggregates. A *property aggregate* is an abstraction for an object that is considered as an aggregation of other objects, one for each attribute belonging to a specified set [Smith and Smith, 1977]. A typical example of such an aggregate is a date, which is seen as an aggregation of three objects, one for the attribute day, one for the attribute month, and one for the attribute year. Another example of property aggregate is an exam, which again is seen as an aggregation of three objects (one professor, one student and one course). This makes it clear that N-ary relations can be modeled as classes whose instances are aggregates. An *instance aggregate* is an abstraction of a group of other objects belonging to a certain class [Brodie and Ridjanovic, 1984]. A typical example of such an aggregate is a team, which can be seen as a group of players. Like any other description logics, $\mathcal{CATS}$ allows one to form *complex classes* by applying suitable constructors to both simple and aggregate classes. Notably, $\mathcal{CATS}$ includes a form of role value map, and the most general form of number restrictions (called qualified).

Links are abstractions for atomic, basic, and complex relationships between classes. An *atomic link* (denoted simply by a name, and also called attribute) is the most elementary mean for establishing a relationship between classes. A *basic* link is formed by applying certain constructors (like inverse, union, intersection and difference) to atomic links, and a *complex* link is formed by applying more complex constructors (like chaining, transitive closure, and identity) to basic links.

A knowledge base in $\mathcal{CATS}$ is simply a set of inclusion axioms. We point out that $\mathcal{CATS}$ allows inclusion assertions to be stated on classes of all kinds (simple, aggregate, and complex), and on basic links, with no limitation (for example on cycles). A particular care is put in devising $\mathcal{CATS}$ so that its reasoning tasks remain decidable and even with the same computational complexity as the simplest description logics where inclusion axioms are allowed. Indeed, we describe a technique for computing logical implication in $\mathcal{CATS}$, and show that this problem is both EXPTIME-hard, and decidable with exponential time in the worst case.

The paper is organized as follows. Section 2 briefly recalls the description logics $\mathcal{CIF}$ [De Giacomo and Lenzerini, 1994], which is the basis of the present work. Section 3 presents our logic $\mathcal{CATS}$, which adds to $\mathcal{CIF}$ suitable constructors for representing aggregations, complex links, and qualified number restrictions. In Section 4, we show examples of knowledge bases built using $\mathcal{CATS}$, and compare the expressive power of the logic with other similar formalisms for knowledge representation. Section 5 illustrates the salient features of the technique we use for computing logical implication in $\mathcal{CATS}$, and discusses the computational complexity of this problem. Finally, conclusions are drawn in Section 6.

2 Preliminaries

Traditionally, description logics allow one to represent a domain of interest in terms of *concepts* and *roles*. Concepts model classes of individuals, while roles model binary relations between classes. Starting with atomic concepts and atomic roles, which are concepts and roles described simply by a name, complex concepts and roles can be build by means of suitable constructors.

In the following, we focus on the description logic $\mathcal{CIF}$ studied in [De Giacomo and Lenzerini, 1994], whose language has the following syntax:

$$\begin{array}{lcl} C & ::= & A \mid C_1 \sqcap C_2 \mid \neg C \mid \forall R.C \mid (\leq 1\ P) \mid (\leq 1\ P^-) \\ R & ::= & P \mid R_1 \cup R_2 \mid R_1 \circ R_2 \mid R^* \mid R^- \mid id(C) \end{array}$$

where A denotes an atomic concept, C (possibly with subscript) a generic concept, P an atomic role, R (possibly with subscript) a generic role.

In description logics, an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a nonempty *domain of interpretation* $\Delta^{\mathcal{I}}$, and an *interpretation function* $\cdot^{\mathcal{I}}$ satisfying the following conditions ($\sharp\{\}$ denotes the cardinality of a set, and $a = P \mid P^-$):

$$\begin{array}{l} A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \\ (C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} \\ (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} - C^{\mathcal{I}} \\ (\forall R.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \forall d'.(d,d') \in R^{\mathcal{I}} \supset d' \in C^{\mathcal{I}}\} \\ (\leq 1\ a)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \sharp\{d' \mid (d,d') \in a^{\mathcal{I}}\} \leq 1\} \\ P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\ (R_1 \cup R_2)^{\mathcal{I}} = R_1^{\mathcal{I}} \cup R_2^{\mathcal{I}} \\ (R_1 \circ R_2)^{\mathcal{I}} = R_1^{\mathcal{I}} \circ R_2^{\mathcal{I}} \\ (R^*)^{\mathcal{I}} = (R^{\mathcal{I}})^* \\ (R^-)^{\mathcal{I}} = \{(d_1,d_2) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid (d_2,d_1) \in R^{\mathcal{I}}\} \\ id(C)^{\mathcal{I}} = \{(d,d) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid d \in C^{\mathcal{I}}\}. \end{array}$$

Note that $\mathcal{CIF}$ has a very expressive language, comprising all usual concept constructs, and a rich set of role constructs, namely: union of roles $R_1 \cup R_2$, chaining of roles $R_1 \circ R_2$, reflexive-transitive closure of roles R^*, inverse roles R^-, and the identity role $id(C)$ projected on C. Moreover $\mathcal{CIF}$ supports the simplest form of cardinality constraints, namely *functional restrictions* of the form $(\leq 1\, P)$, interpreted as the set of individuals for which the role P is functional. Notably, functional restrictions can be applied to both atomic roles and inverse of atomic roles. In fact, in $\mathcal{CIF}$ there is a perfect symmetry between roles and inverse roles, that will be the basis for the extensions of the logic discussed in this paper.

Let $\mathcal{L}$ be any description logic. An $\mathcal{L}$ *TBox*[1] $\mathcal{K}$ is a finite set of axioms of the form $C_1 \sqsubseteq C_2$, called *inclusion assertions*, where C_1 and C_2 are $\mathcal{L}$ concepts. An interpretation $\mathcal{I}$ satisfies an inclusion assertion $C_1 \sqsubseteq C_2$

[1] TBox is the term traditionally used for naming the set of axioms constituting the intensional level (schema level) of a knowledge base.

if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{K}$ if $\mathcal{I}$ satisfies each inclusion assertion in $\mathcal{K}$.

The most important form of reasoning in a TBox is *logical implication*: a TBox $\mathcal{K}$ logically implies an assertion $C_1 \sqsubseteq C_2$, written $\mathcal{K} \models C_1 \sqsubseteq C_2$, if $C_1 \sqsubseteq C_2$ is satisfied by every model of $\mathcal{K}$. It is well known that all basic reasoning tasks can be (linearly) reformulated as logical implication in a TBox.[2]

In [De Giacomo and Lenzerini, 1994], it is shown that logical implication in $\mathcal{CIF}$ TBoxes, is an EXPTIME-complete problem. This result was devised in the setting of the correspondence between description logics and propositional dynamic logics, originally pointed out in [Schild, 1991]. One consequence is that logical implication in $\mathcal{CIF}$ can be decided by means of the decision procedures designed for satisfiability checking in dynamic logics. This is the first decidability result for a description logic including axioms, inverse roles, and functional restrictions, a fundamental combination as seen in the introduction, and also contributed to the research on modal logics of programs, since the decidability of the propositional dynamic logic corresponding to $\mathcal{CIF}$ was not known. Note that, $\mathcal{CIF}$ is one of the very few description logics that lacks the finite model property, which means that certain $\mathcal{CIF}$ TBoxes admit only infinite models. This is one of reasons why proving the above result required a quite sophisticated technique.

In the following, we will refer to a further description logic, called $\mathcal{CIQ}$, which is obtained from $\mathcal{CIF}$ by extending functional restrictions to *qualified number restrictions* $(\leq k\, a.C)$, whose meaning in an interpretation $\mathcal{I}$ is (k is any integer[3] ≥ 1):

$$(\leq k\, a.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \sharp\{d' \mid (d,d') \in a^{\mathcal{I}} \wedge d' \in C^{\mathcal{I}}\} \leq k\}.$$

3 The description logic $\mathcal{CATS}$

In this section, we provide the formal definition of $\mathcal{CATS}$. As we said in the introduction, the language of $\mathcal{CATS}$ supports classes and links. Classes are partitioned into simple classes and aggregate classes, which are further distinguished in property aggregate and instance aggregate classes. Links are partitioned into atomic (also called attributes), basic, and complex.

Let a nonempty finite alphabet $\mathcal{A}$ of atomic classes (classes denoted simply by a name, no matter if simple or aggregate) and a nonempty finite alphabet $\mathcal{U}$ of attributes be available. We use A for a generic element of $\mathcal{A}$, U (possibly with subscript) for a generic element of $\mathcal{U}$, C (possibly with subscript) for a generic class, b (possibly with subscript) for a generic basic link, and L (possibly with subscript) for a generic complex link. The language of $\mathcal{CATS}$ has the following syntax ($n, k \geq 1$):

$$\begin{array}{lcl} C & ::= & A \mid \tau(U_1,\ldots,U_n) \mid \chi(C,U_1,\ldots,U_n) \mid \sigma(C) \mid \\ & & C_1 \sqcap C_2 \mid \neg C \mid \forall L.C \mid (\leq k\, b.C) \mid (\leq k\, b^-.C) \mid \\ & & (b_1 \subseteq b_2) \mid (b_1^- \subseteq b_2^-) \\ b & ::= & U \mid \ni \mid b_1 \cup b_2 \mid b_1 \setminus b_2 \\ L & ::= & b \mid L_1 \circ L_2 \mid L_1 \cup L_2 \mid L^* \mid L^- \mid id(C) \end{array}$$

We use a (possibly with subscript) for b and b^-, and we adopt the following abbreviations: $\top \doteq A \sqcup \neg A$, $\bot \doteq \neg\top$, $\tau \doteq \tau(U_1) \sqcup \cdots \sqcup \tau(U_m)$ (where $\{U_1,\ldots,U_m\} = \mathcal{U}$), $\sigma \doteq \sigma(\top)$, $C_1 \sqcup C_2 \doteq \neg(\neg C_1 \sqcap \neg C_2)$, $\exists L.C \doteq \neg\forall L.\neg C$, $\emptyset \doteq \ni \setminus \ni$, $(\geq k\, a.C) \doteq \neg(\leq k+1\, a.C)$, $a_1 \cap a_2 \doteq a_1 \setminus (a_1 \setminus a_2)$, and $(a_1 = a_2) \doteq (a_1 \subseteq a_2) \sqcap (a_2 \subseteq a_1)$. Parentheses are used to disambiguate expressions.

The semantics for the language of $\mathcal{CATS}$ is based on an *interpretation* $\mathcal{I} = (\mathcal{O}^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\mathcal{O}^{\mathcal{I}}$ is the *universe* of the interpretation, and $\cdot^{\mathcal{I}}$ is the *interpretation function* over such a universe. Differently from the usual notion of interpretation, $\mathcal{O}^{\mathcal{I}}$ is a nonempty set of *polymorphic objects*, which means that every object in $\mathcal{O}^{\mathcal{I}}$ has none, one, or both of the following two forms:

1. The form of *tuple*: when an object has this form, it can be considered as a property aggregation, which is formally defined as a partial function from $\mathcal{U}$ to $\mathcal{O}^{\mathcal{I}}$. We use the term tuple to denote an object in $\mathcal{O}^{\mathcal{I}}$ that has the form of tuple, and we write $\langle U_1 : o_1, \ldots, U_n : o_n\rangle$[4] to denote any tuple t such that, for each $i \in \{1,\ldots,n\}$, $t(U_i)$ is defined and equal to o_i (which is called the U_i-component of t). Note that the tuple t may have other components as well, besides the U_i-components.

2. The form of *set*: when an object o has this form, it can be considered as an instance aggregate, which is formally defined as a nonempty finite collection of objects in $\mathcal{O}^{\mathcal{I}}$, with the following proviso: the view of o as a set is unique, in the sense that there is only one finite collection of objects of which o can be considered an aggregation, and no other object o' is the aggregation of the same collection. We use the term set to denote an object in $\mathcal{O}^{\mathcal{I}}$ that has the form of set, and we write $\{\!|o_1,\ldots,o_n|\!\}$ to denote the collection whose members are exactly $o_1,\ldots,o_n$.

Objects having none of these forms are called elementary objects - i.e., individuals with no structure.

The interpretation function $\cdot^{\mathcal{I}}$ is defined as follows:

- It assigns to $\ni$ a subset of $\mathcal{O}^{\mathcal{I}} \times \mathcal{O}^{\mathcal{I}}$ such that for each $\{\!|\ldots,o,\ldots|\!\} \in \mathcal{O}^{\mathcal{I}}$, we have that $(\{\!|\ldots,o,\ldots|\!\}, o) \in \ni^{\mathcal{I}}$.
- It assigns to every attribute U a subset of $\mathcal{O}^{\mathcal{I}} \times \mathcal{O}^{\mathcal{I}}$ such that, for each $\langle\ldots,U : o,\ldots\rangle \in \mathcal{O}^{\mathcal{I}}$, $(\langle\ldots,U:o,\ldots\rangle, o) \in U^{\mathcal{I}}$, and there is no $o' \in \mathcal{O}^{\mathcal{I}}$ different from o such that $(\langle\ldots,U:o,\ldots\rangle, o') \in U^{\mathcal{I}}$. Note that this implies that every U in a tuple is functional for the tuple.

[2] Actually, in description logics having a language rich enough, logical implication can in turn be (polynomially) reduced to satisfiability of a single concept. This is basically due to the ability of expressing reflexive-transitive closure of roles, together with the "connected model property", i.e. if a TBox has a model, it has a model which is connected (see [Schild, 1991; De Giacomo and Lenzerini, 1994]).

[3] We assume that integers are coded in unary.

[4] This notation makes it clear that a tuple is indeed a function assigning one element of $\mathcal{O}^{\mathcal{I}}$ to some of the elements of $\mathcal{U}$.

- It assigns to every basic link a subset of $\mathcal{O}^{\mathcal{I}} \times \mathcal{O}^{\mathcal{I}}$ such that the following conditions are satisfied:

$$\begin{array}{lcl}(b_1 \cup b_2)^{\mathcal{I}} & = & b_1^{\mathcal{I}} \cup b_2^{\mathcal{I}} \\ (b_1 \setminus b_2)^{\mathcal{I}} & = & b_1^{\mathcal{I}} - b_2^{\mathcal{I}} \\ (b^-)^{\mathcal{I}} & = & \{(o, o') \mid (o', o) \in b^{\mathcal{I}}\}.\end{array}$$

- It assigns to every complex link a subset of $\mathcal{O}^{\mathcal{I}} \times \mathcal{O}^{\mathcal{I}}$ such that the usual conditions for $\circ$, $\cup$, *, $^-$, and id are satisfied (see Section 2).
- It assigns to every class a subset of $\mathcal{O}^{\mathcal{I}}$ in such a way that the following conditions are satisfied:
  - $A^{\mathcal{I}} \subseteq \mathcal{O}^{\mathcal{I}}$
  - $\tau(U_1, \ldots, U_n)^{\mathcal{I}} = \{\langle U_1 : o_1, \ldots, U_n : o_n\rangle \in \mathcal{O}^{\mathcal{I}} \mid o_1, \ldots, o_n \in \mathcal{O}^{\mathcal{I}}\}$
  - $\chi(C, U_1, \ldots, U_n)^{\mathcal{I}} = S \subseteq \tau(U_1, \ldots, U_n) \cap C^{\mathcal{I}}$ and no distinct $s, s' \in S$ have the same $U_1, \ldots, U_n$-components
  - $\sigma(C)^{\mathcal{I}} = \{\{\!|o_1, \ldots, o_n|\!\} \in \mathcal{O}^{\mathcal{I}} \mid o_1, \ldots, o_n \in C^{\mathcal{I}}\}$
  - $(C_1 \sqcap C_2)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
  - $(\neg C)^{\mathcal{I}} = \mathcal{O}^{\mathcal{I}} - C^{\mathcal{I}}$
  - $(\forall L.C)^{\mathcal{I}} = \{o \in \mathcal{O}^{\mathcal{I}} \mid \forall o'.(o, o') \in R^{\mathcal{I}} \supset o' \in L^{\mathcal{I}}\}$
  - $(\leq k\, a.C)^{\mathcal{I}} = \{o \in \mathcal{O}^{\mathcal{I}} \mid \sharp\{(o, o') \in a^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\} \leq k\}$
  - $(a_1 \subseteq a_2)^{\mathcal{I}} = \{o \in \mathcal{O}^{\mathcal{I}} \mid \{o' \mid (o, o') \in a_1^{\mathcal{I}}\} \subseteq \{o' \mid (o, o') \in a_2^{\mathcal{I}}\}\}$.

A $\mathcal{CATS}$ TBox $\mathcal{K}$ is a finite set of inclusion assertions of the form $C_1 \sqsubseteq C_2$, where C_1 and C_2 are classes in $\mathcal{CATS}$ (we write $C_1 \equiv C_2$ for $C_1 \sqsubseteq C_2, C_2 \sqsubseteq C_1$). As usual, an interpretation $\mathcal{I}$ satisfies $C_1 \sqsubseteq C_2$ if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$, and $\mathcal{K} \models C_1 \sqsubseteq C_2$, if $C_1 \sqsubseteq C_2$ is satisfied by every model of $\mathcal{K}$, i.e. by every interpretation that satisfies all assertions in $\mathcal{K}$.

4 Examples and discussion

The goal of this section is to discuss, by means of examples, the most important modeling capabilities of $\mathcal{CATS}$, and to compare our logic with other representation formalisms.

Example 1: Propositional formulae

Figure 1 shows a simple TBox $\mathcal{H}$ defining several well known classes of propositional conjunctive normal form formulae. This example basically shows the power of instance aggregates and qualified number restrictions. None of the decidable description logics that we are aware of is rich enough to express the TBox in the figure. Note that $\texttt{PosLit} \sqsubseteq \chi(\texttt{PosLit}, \texttt{letter})$ imposes that no two positive literals have the same variable (the same for `NegLit`). As an example of inference, $\mathcal{H}$ correctly classifies $\sigma(\texttt{Clause}) \sqcap \forall\ni.((\leq 2\ \ni.\top) \sqcap \exists\ni.\texttt{NegLit})$ under both `Horn` and `Krom`.

Example 2: Families, persons, and cities

Figure 2 shows a TBox $\mathcal{K}$ modeling a world with persons, families and cities. The following observations help understanding the expressive power of $\mathcal{CATS}$.

$$\begin{array}{l}
\texttt{Formula} \equiv \sigma(\texttt{Clause}) \\
\texttt{Clause} \equiv \texttt{EmptyCl} \sqcup \sigma(\texttt{Literal}) \\
\texttt{EmptyCl} \sqsubseteq \forall\ni.\neg\texttt{Literal} \\
\texttt{Literal} \equiv (\texttt{PosLit} \sqcup \texttt{NegLit}) \sqcap \tau(\texttt{letter}) \sqcap \forall\texttt{letter}.\texttt{Var} \\
\texttt{PosLit} \sqsubseteq \chi(\texttt{PosLit}, \texttt{letter}) \sqcap \neg\texttt{NegLit} \\
\texttt{NegLit} \sqsubseteq \chi(\texttt{NegLit}, \texttt{letter}) \\
\texttt{Var} \sqsubseteq (\exists\texttt{letter}^-.\texttt{NegLit}) \sqcap (\exists\texttt{letter}^-.\texttt{PosLit}) \sqcap \\
\qquad (\leq 2\,\texttt{letter}^-.\top) \\
\texttt{HornCl} \equiv \texttt{Clause} \sqcap (\leq 1\ \ni.\texttt{PosLit}) \\
\texttt{TwoClause} \equiv \texttt{Clause} \sqcap (\leq 2\ \ni.\top) \\
\texttt{Horn} \equiv \texttt{Formula} \sqcap \forall\ni.\texttt{HornCl} \\
\texttt{Krom} \equiv \texttt{Formula} \sqcap \forall\ni.\texttt{TwoClause}
\end{array}$$

Figure 1: Propositional formulae

$$\begin{array}{l}
\top \sqsubseteq (\texttt{father} \cap \texttt{mother} \subseteq \emptyset) \sqcap \\
\qquad (\texttt{father} \cap \texttt{children} \subseteq \emptyset) \sqcap \\
\qquad (\texttt{children} \cap \texttt{mother} \subseteq \emptyset) \\
\top \sqsubseteq \forall\texttt{father}^- \cup \texttt{mother}^- \cup \texttt{children}^-.\texttt{Family} \\
\texttt{Date} \equiv \chi(\texttt{Date}, \texttt{day}, \texttt{month}, \texttt{year}) \\
\exists\texttt{date}^-.\top \sqsubseteq \texttt{Date} \\
\exists\texttt{day}^-.\top \sqsubseteq \texttt{Day} \\
\exists\texttt{month}^-.\top \sqsubseteq \texttt{Month} \\
\exists\texttt{year}^-.\top \sqsubseteq \texttt{Year} \\
\exists\texttt{city}^-.\top \sqsubseteq \texttt{City} \\
\texttt{Day} \sqcup \texttt{Month} \sqcup \texttt{Year} \sqsubseteq \neg\tau \sqcap \neg\sigma \\
\texttt{Mayor} \equiv \exists\texttt{mayor}^-.\top \\
\exists\texttt{mayor}.\top \sqsubseteq \texttt{City} \\
\texttt{City} \sqsubseteq \tau(\texttt{name}, \texttt{state}, \texttt{country}, \texttt{mayor}) \sqcap \\
\qquad \chi(\texttt{City}, \texttt{name}, \texttt{state}, \texttt{country}) \sqcap \chi(\texttt{City}, \texttt{mayor}) \\
\texttt{Family} \sqsubseteq \sigma(\texttt{Person}) \sqcap \tau(\texttt{father}, \texttt{mother}, \texttt{date}, \texttt{city}) \sqcap \\
\qquad \chi(\texttt{Family}, \texttt{father}, \texttt{mother}, \texttt{date}) \sqcap \\
\qquad (\ni= \texttt{father} \cup \texttt{mother} \cup \texttt{children}) \\
\texttt{StillFamily} \sqsubseteq \texttt{Family} \sqcap \chi(\texttt{StillFamily}, \texttt{father}, \texttt{mother}) \\
\texttt{PhdFamily} \equiv (\geq 3\ \ni.\texttt{PhdPerson}) \sqcap (\leq 1\ \ni.\neg\texttt{PhdPerson}) \\
\texttt{Person} \sqsubseteq (\exists\texttt{children}^-.\top) \sqcap (\leq 1\,\texttt{children}^-.\top) \\
\texttt{ChildOfMayor} \equiv \exists\texttt{children}^- \circ \texttt{father}.\texttt{Mayor} \\
\texttt{VeryPhd} \equiv \forall(\texttt{children}^- \circ (\texttt{father} \cup \texttt{mother}))^*.\texttt{PhdPerson}
\end{array}$$

Figure 2: Families, persons, and cities

- Objects are polymorphic. For example, every instance of `Family` (representing families resulting from a marriage) can be seen both as a set of persons, and as a tuple with attributes `father`, `mother`, `date` (of marriage) and `city` (of marriage). Note, however, that assertions can be used to impose that the instances of a certain class (`Day`, `Month` and `Year` in our example) can only be seen as elementary objects.
- Inclusion assertions on classes are used with no limitation. In particular, they can be stated for all kinds of classes, and cycles are allowed in the TBox. Notably, inclusion assertions can also be stated for basic links: indeed, $\top \sqsubseteq (b_1 \subseteq b_2)$ forces b_1 to a subset of b_2 in every model of $\mathcal{K}$. Inclusion assertions of this kind are used in the example to specify the properties of the attributes `father`, `mother` and `children`.
- N-ary relations are supported. Any instance of

`Family` can indeed be considered as a relation with four arguments. The χ constructor is used to define keys for (N-ary) relations: for example, the fact that every instance of `Family` is an instance of $\chi(\texttt{Family}, \texttt{father}, \texttt{mother}, \texttt{date})$ implies that the three attributes form a key for the class. On the other hand, `StillFamily`, representing families whose father and mother are still married, has a more specialized key, constituted by the attributes `father` and `mother`. Observe that several keys can be defined for a class (see `City`).

- Qualified number restrictions and role value maps on basic links can be used without any limitation. Indeed, $(\ni = \texttt{father} \cup \texttt{mother} \cup \texttt{children})$ is a role value map on basic links.
- Complex links can be used for modeling interesting relationships. For example, the relationship `hasfather` between a person and her/his father is captured in $\mathcal{K}$ by $\texttt{children}^- \circ \texttt{father}$ (similarly for `hasmother`). Also, `ancestor` is captured by $(\texttt{hasfather} \cup \texttt{hasmother}) \circ (\texttt{hasfather} \cup \texttt{hasmother})^*$ (see the definition of `VeryPhd`).

As an example of inference that can be draw from $\mathcal{K}$, observe that:

$$\mathcal{K} \models \exists \texttt{children}^-.\top \sqsubseteq \texttt{Person} \sqcap \exists \ni^- .\exists \texttt{father}.\top.$$

Indeed, note that every instance of $\exists \texttt{children}^-.\top$ is also an instance of $\exists \texttt{father}^- \cup \texttt{mother}^- \cup \texttt{children}^-.\top$ and therefore is an instance of `Family`. This means that $\mathcal{K} \models \exists \texttt{children}^-.\top \sqsubseteq \exists \texttt{children}^-.\texttt{Family}$. Observe that $\mathcal{K} \models \texttt{Family} \sqsubseteq (\texttt{children} \subseteq \ni)$, and, since $\mathcal{K} \models \texttt{Family} \sqsubseteq \forall \ni .\texttt{Person}$ (because $\mathcal{K} \models \texttt{Family} \sqsubseteq \sigma(\texttt{Person})$), we have that $\mathcal{K} \models \exists \texttt{children}^-.\top \sqsubseteq \exists \texttt{children}^-.(\forall \texttt{children}.\texttt{Person})$, which implies that $\mathcal{K} \models \exists \texttt{children}^-.\top \sqsubseteq \texttt{Person}$. The fact that $\mathcal{K} \models \exists \texttt{children}^-.\top \sqsubseteq \exists \ni^- .\exists \texttt{father}.\top$ easily follows from the fact that every `Family` is a tuple with attribute `father`.

Relationship with other formalisms

It is easy to see that we obtain $\mathcal{CIQ}$ from $\mathcal{CATS}$ by simply ignoring tuples, sets, and basic links (other than atomic ones). As another example of description logic supporting general inclusion assertions, the logic $\mathcal{ALCNR}$, studied in [Buchheit *et al.*, 1993], can be obtained from $\mathcal{CATS}$ by ignoring tuples, sets, complex links, the $(a_1 \subseteq a_2)$ constructs, and by allowing only basic links of the form $b_1 \cap b_2$, and number restrictions of the form $(\leq k\, b.\top)$. Also the formalism in [Schmolze, 1989] supporting N-ary terms can be easily expressed in our logic. More generally, it can be shown the vast majority of decidable description logics proposed in the literature are captured by $\mathcal{CATS}$.

Space limitation prevents us from showing how database models can be captured by $\mathcal{CATS}$. We simply observe that we obtain the Entity-Relationship model by allowing one level of nesting in tuples, and by partitioning elementary classes in entities and attributes. Analogously, the nested relational model, as well as complex objects data models are expressed in $\mathcal{CATS}$ by imposing suitable limitations on tuples and sets.

As a final observation, we would like to note that $\mathcal{CATS}$ can also be used for expressing knowledge bases that go beyond classes and links. In [De Giacomo and Lenzerini, 1995], we show that a weak version of $\mathcal{CATS}$ can be used as a sort of monotonic propositional situation calculus extended with complex and concurrent actions. Roughly speaking, states are modeled by elementary objects, and atomic actions are modeled by atomic links. A propositional fluent then simply becomes a class (whose instances are those states where the fluent is true), and preconditions, postconditions, effects of actions, as well as frame axioms are expressed by inclusion assertions. Finally, basic links, complex links, and qualified number restrictions are used to model complex properties of actions, including concurrency. For example, $a_1 \cap a_2$ denotes the concurrent execution of the actions a_1 and a_2), while an inclusion assertion of the form $\top \sqsubseteq (\leq 1\, (U_1 \cup \cdots \cup U_n)^-.\top)$ (where $U_1, \ldots, U_n$ are all the atomic actions) can be used to impose that the past is backward linear -i.e., that every state as at most one predecessor.

5 Decision procedure for $\mathcal{CATS}$

We investigate the decidability and the complexity of $\mathcal{CATS}$, showing that logical implication in $\mathcal{CATS}$ is polynomially reducible to logical implication in $\mathcal{CIF}$, which is decidable and EXPTIME-complete. The reduction is done in two steps: first, we "reify" basic links, tuples and sets, so that objects in the interpretations are elementary objects; then, we reduce qualified number restrictions to functional restrictions.

We define a mapping δ'' form $\mathcal{CATS}$ basic links to $\mathcal{CIQ}$ concepts defined as ($\top_U$, $\top_\ni$ are new atomic concepts):

$$\begin{array}{ll} \delta''(U) = \top_U & \delta''(b_1 \cup b_2) = \delta''(b_1) \sqcup \delta''(b_2) \\ \delta''(\ni) = \top_\ni & \delta''(b_1 \setminus b_2) = \delta''(b_1) \sqcap \neg\delta''(b_2), \end{array}$$

a mapping δ' form $\mathcal{CATS}$ links to $\mathcal{CIQ}$ roles defined as (V_1, V_2 are new atomic roles):

$$\begin{array}{ll} \delta'(b) = V_1^- \circ id(\delta''(b)) \circ V_2 & \delta'(L^*) = \delta'(L)^* \\ \delta'(L_1 \cup L_2) = \delta'(L_1) \cup \delta'(L_2) & \delta'(L^-) = \delta'(L)^- \\ \delta'(L_1 \circ L_2) = \delta'(L_1) \circ \delta'(L_2) & \delta'(id(C)) = id(\delta(C)), \end{array}$$

and a mapping δ form $\mathcal{CATS}$ classes to $\mathcal{CIQ}$ concepts defined as ($\top_{\tau(U_1,\ldots,U_n)}$, $\top_{\chi(C,U_1,\ldots,U_n)}$, $\top_\sigma$ are new atomic concepts):

$$\begin{array}{l} \delta(A) = A \\ \delta(\tau(U_1, \ldots, U_n)) = \top_{\tau(U_1,\ldots,U_n)} \\ \delta(\sigma(C)) = \top_\sigma \sqcap \forall (V_1^- \circ id(\top_\ni) \circ V_2).\delta(C) \\ \delta(\chi(C, U_1, \ldots, U_n)) = \top_{\chi(C,U_1,\ldots,U_n)} \\ \delta(C_1 \sqcap C_2) = \delta(C_1) \sqcap \delta(C_2) \\ \delta(\neg C) = \neg\delta(C) \\ \delta(\forall L.C) = \forall \delta'(L).\delta(C) \\ \delta((\leq k\, b.C)) = (\leq k\, V_1^-.(\delta''(b) \sqcap \forall V_2.\delta(C))) \\ \delta((\leq k\, b^-.C)) = (\leq k\, V_2^-.(\delta''(b) \sqcap \forall V_1.\delta(C))) \\ \delta((b_1 \subseteq b_2)) = \forall V_1.(\neg\delta''(b_1) \sqcup \delta''(b_2)) \\ \delta((b_1^- \subseteq b_2^-)) = \forall V_2.(\neg\delta''(b_1) \sqcup \delta''(b_2)). \end{array}$$

Making use of δ, we define the so called reified counterpart of a $\mathcal{CATS}$ TBox, which is, in fact, a $\mathcal{CIQ}$ TBox. We use the following abbreviations: $P_U \doteq V_1^- \circ id(\top_U) \circ V_2$, $memb \doteq V_1^- \circ id(\top_\ni) \circ V_2$, $(\leq k\, P_U.C) \doteq (\leq k\, V_1^-.(\top_U \sqcap \forall V_2.C))$ and $(\leq k\, P_U^-.C) \doteq (\leq k\, V_2^-.(\top_U \sqcap \forall V_1.C))$.

Definition Let $\mathcal{K}$ be a $\mathcal{CATS}$ TBox. We call its *reified counterpart* the TBox $\wp(\mathcal{K})$ defined as $\wp(\mathcal{K}) = \wp_1(\mathcal{K}) \cup \wp_2(\mathcal{K}) \cup \wp_3(\mathcal{K})$, where:

- $\wp_1(\mathcal{K}) = \{\delta(C_1) \sqsubseteq \delta(C_2) \mid C_1 \sqsubseteq C_2 \in \mathcal{K}\}$;
- $\wp_2(\mathcal{K})$ is the set constituted by:
  - $\top_{\tau(U_1,\ldots,U_n)} \equiv \exists P_{U_1}.\top \sqcap (\leq 1\, P_{U_1}.\top) \sqcap \ldots \sqcap \exists P_{U_n}.\top \sqcap (\leq 1\, P_{U_n}.\top)$ for each $\top_{\tau(U_1,\ldots,U_n)}$, with $n \geq 1$, in $\wp_1(\mathcal{K})$
  - $\top_{\chi(C,U)} \sqsubseteq \exists P_U.\top \sqcap (\leq 1\, P_U.\top) \sqcap (\forall P_U.(\leq 1\, P_U^-.\delta(C))) \sqcap \delta(C)$ for each $\top_{\chi(C,U)}$ in $\wp_1(\mathcal{K})$
  - $\top_{\chi(C,U_1,\ldots,U_n)} \sqsubseteq \top_{\tau(U_1,\ldots,U_n)} \sqcap \delta(C)$ for each $\top_{\chi(C,U_1,\ldots,U_n)}$, with $n \geq 2$, in $\wp_1(\mathcal{K})$
  - $\top_\sigma \equiv \exists memb.\top$
  - $\top_\sigma \sqcap (\leq 1\, memb.\top) \sqsubseteq \forall memb.(\leq 1\, memb^-.(\leq 1\, memb.\top))$;
- $\wp_3(\mathcal{K})$ is the set constituted by ($\top_\mathcal{C}, \top_\mathcal{L}$ are new atomic concepts): $\top \sqsubseteq \top_\mathcal{L} \sqcup \top_\mathcal{C}$, $\top_\mathcal{C} \sqsubseteq \forall V_1.\bot \sqcap \forall V_2.\bot$, $\top_\mathcal{L} \sqsubseteq \exists V_1.\top_\mathcal{C} \sqcap (\leq 1\, V_1.\top) \sqcap \exists V_2.\top_\mathcal{C} \sqcap (\leq 1\, V_2.\top)$, one $\top_U \sqsubseteq \top_\mathcal{L}$ for each $\top_U$ in $\wp_1(\mathcal{K}) \cup \wp_2(\mathcal{K})$, $\top_\ni \sqsubseteq \top_\mathcal{C}$, one $A \sqsubseteq \top_\mathcal{C}$ for each A in $\wp_1(\mathcal{K}) \cup \wp_2(\mathcal{K})$, one $\top_{\tau(U_1,\ldots,U_n)} \sqsubseteq \top_\mathcal{C}$ for each A in $\wp_1(\mathcal{K}) \cup \wp_2(\mathcal{K})$, and $\top_\sigma \sqsubseteq \top_\mathcal{C}$. □

Observe that $\wp(\mathcal{K})$ has size which is polynomially related to the size of $\mathcal{K}$, and is interpreted over usual interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a set of individuals. However, to represent the intended meaning of the constructs $\chi(C, U_1, \ldots, U_n)$ and $\sigma(C)$, $\wp(\mathcal{K})$ must be interpreted over a special kind of interpretations.

Definition We call a model $\mathcal{I}$ of a $\wp(\mathcal{K})$ an *aggregate-descriptive model*, if for all distinct individuals d, d' the following conditions hold:

$d, d' \in \top_\mathcal{L}^\mathcal{I} \supset$
$\quad \neg((d,d_1) \in V_1^\mathcal{I} \wedge (d',d_1) \in V_1^\mathcal{I} \wedge$
$\quad\quad (d',d_2) \in V_2^\mathcal{I} \wedge (d',d_2) \in V_2^\mathcal{I})$
$d, d' \in \top_\sigma^\mathcal{I} \supset$
$\quad \exists d'' \in \Delta^\mathcal{I}.\neg((d,d'') \in memb^\mathcal{I} \wedge (d',d'') \in memb^\mathcal{I})$
$d, d' \in \top_{\chi(C,U_1,\ldots,U_n)}^\mathcal{I} \supset$
$\quad \neg((d,d_1) \in P_{U_1}^\mathcal{I} \wedge (d',d_1) \in P_{U_1}^\mathcal{I} \wedge$
$\quad\quad \ldots$
$\quad\quad (d',d_n) \in P_{U_n}^\mathcal{I} \wedge (d',d_n) \in P_{U_n}^\mathcal{I})$
for every $\top_{\chi(C,U_1,\ldots,U_n)}$, with $n \geq 2$, in $\wp(\mathcal{K})$ □

Theorem 1 *Let $\mathcal{K}$ be a $\mathcal{CATS}$ TBox, and $\wp(\mathcal{K})$ its reified counterpart. Then there is a mapping from the models of $\mathcal{K}$ to the aggregate-descriptive models of $\wp(\mathcal{K})$ and vice versa.*

A consequence of Theorem 1 is that $\mathcal{K} \models C_1 \sqsubseteq C_2$[5] is equivalent to $\wp(\mathcal{K}) \models \wp(\{C_1 \sqsubseteq C_2\})$. However, this does not directly yields a reasoning procedure for $\mathcal{CATS}$, since it is not known how to reason with a $\wp(\mathcal{K})$ interpreted over aggregate-descriptive models.

Theorem 2 below is the first substantial step towards devising such a reasoning procedure. It guarantees that on one hand we can interpret $\wp(\mathcal{K})$ on aggregate-descriptive models only, and on the other hand, we can forget about aggregate-descriptive models when computing logical consequence in $\wp(\mathcal{K})$. The proof of Theorem 2 is based on the *disjoint union model property*: let $\mathcal{K}$ be a $\mathcal{CIQ}$ TBox 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 $\mathcal{I} \uplus \mathcal{J} = (\Delta^\mathcal{I} \uplus \Delta^\mathcal{J}, \cdot^\mathcal{I} \uplus \cdot^\mathcal{J})$ made up by the disjoint union of $\mathcal{I}$ and $\mathcal{J}$, is a model of $\mathcal{K}$. We remark that most description logics have such a property, which is, in fact, typical of Modal Logics.

Theorem 2 *Let $\mathcal{K}$ be a $\mathcal{CATS}$ TBox and $\wp(\mathcal{K})$ its reified counterpart. If $\wp(\mathcal{K})$ has a model, then it has an aggregate-descriptive model.*

The reified counterpart $\wp(\mathcal{K})$ of a $\mathcal{CATS}$ TBox $\mathcal{K}$ is not yet a $\mathcal{CIF}$ TBox, because it contains qualified number restrictions. In order to deal with them, we represent the role V_i^- $(i = 1, 2)$, which is not functional (while V_i is so), by the role $F_{V_i} \circ F'^*_{V_i}$, where F_{V_i}, F'_{V_i} are new functional roles. The main point of such transformation is that now qualified number restrictions can be encoded as constraints on the chain $F_{V_i} \circ F'^*_{V_i}$. Formally, we define the $\mathcal{CIF}$ counterpart of a $\mathcal{CATS}$ TBox as follows.

Definition Let $\mathcal{K}$ be a $\mathcal{CATS}$ TBox and $\wp(\mathcal{K})$ its reified counterpart. We define the *$\mathcal{CIF}$ counterpart* $\upsilon(\mathcal{K})$ of $\mathcal{K}$ as $\upsilon(\mathcal{K}) = \upsilon_1(\mathcal{K}) \wedge \upsilon_2(\mathcal{K})$, where:

- $\upsilon_1(\mathcal{K})$ is obtained from $\wp(\mathcal{K})$ by recursively replacing:
  - every occurrence of V_i with $(F_{V_i} \circ F'^*_{V_i})^-$, where F_{V_i}, F'_{V_i} are new atomic roles,
  - every $(\leq 1\, V_i.\top)$ with $\top$,
  - every $(\leq k\, V_i^-.C)$ with $\forall(F_{V_i} \circ F'^*_{V_i} \circ (id(C) \circ F'^+_{V_i})^k.\neg C$, where r^+ stands for $r; r*$ and r^k stands for $r \circ \ldots \circ r$ (k times);
- $\upsilon_2(\mathcal{K})$ is the set constructed by one $\top \sqsubseteq (\leq 1\, F_{V_i}) \sqcap (\leq 1\, F'_{V_i}) \sqcap (\leq 1\, F_{V_i}^-) \sqcap (\leq 1\, F'^-_{V_i}) \sqcap \neg(\exists F_{V_i}^-.\top \sqcap \exists F'^-_{V_i}.\top)$ for each $F_{V_i} \circ F'^*_{V_i}$ in $\upsilon_1(\mathcal{K})$. □

Note that, the inclusion assertions in $\upsilon_2(\mathcal{K})$ constrain the roles $F_{V_i}, F'_{V_i}, F_{V_i}^-, F'^-_{V_i}$ to be functional, and impose that each individual cannot be linked to other individuals through both $F_{V_i}^-$ and $F'^-_{V_i}$, thus implying that $(F_{V_i} \circ F'^*_{V_i})^-$ is functional. Observe also that the size of $\upsilon(\mathcal{K})$ is polynomially related to the size of $\mathcal{K}$.

Theorem 3 *Let $\mathcal{K}$ be a $\mathcal{CATS}$ TBox, $\wp(\mathcal{K})$ its reified counterpart, and $\upsilon(\mathcal{K})$ its $\mathcal{CIF}$ counterpart. Then there*

[5] Without loss of generality, we assume that in a logical implication $\mathcal{K} \models C_1 \sqsubseteq C_2$, the atomic concepts and links, as well as the $\tau(U_1, \ldots, U_n)$ and $\chi(C, U_1, \ldots, U_n)$ occurring in $C_1 \sqsubseteq C_2$, also occur in $\mathcal{K}$.

is a mapping from the models of $\wp(\mathcal{K})$ to the models of $v(\mathcal{K})$, and vice versa.

Now we can establish the decidability and the computational complexity of logical implication in $\mathcal{CATS}$ TBoxes. Indeed, by Theorem 3, $\mathcal{K} \models C_1 \sqsubseteq C_2$ is reducible to $v(\mathcal{K}) \models v(\{C_1 \sqsubseteq C_2\})$. Since the size of $v(\mathcal{K}) \cup v(\{C_1 \sqsubseteq C_2\})$ is polynomially related to $\mathcal{K} \cup \{C_1 \sqsubseteq \mathcal{C}_2\}$, we get the following result.

Theorem 4 *Logical implication in $\mathcal{CATS}$ is EXPTIME-complete.*

6 Conclusions

It is our opinion that the work described in this paper makes description logics accomplish the necessary leap in order to be well equipped for the new challenging applications they are faced with. Our first investigations show that $\mathcal{CATS}$ can indeed capture and extend most class-based representation formalisms used in different areas as AI, databases, software engineering, etc.. One main issue still remains to be addressed, namely, the possibility of adding to $\mathcal{CATS}$ suitable constructs for expressing finiteness of nested aggregates, and, correspondingly, suitable techniques for reasoning in finite models (in the style of [Calvanese *et al.*, 1994]). This will be the subject of further research.

References

[Baader, 1991] F. Baader. Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles. In *Proc. of IJCAI-91* 1991.

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

[Borgida, 1992] A. Borgida. From type systems to knowledge representation: Natural semantics specifications for description logics. *J. of Intelligent and Cooperative Information Systems*, 1(1):93–126, 1992.

[Brachman and Levesque, 1985] R. J. Brachman and H. J. Levesque. *Readings in Knowledge Representation.* Morgan Kaufmann, Los Altos, 1985.

[Brachman, 1977] R. J. Brachman. What's in a concept: Structural foundations for semantic networks. *International Journal of Man-Machine Studies*, 9(2):127–152, 1977.

[Brodie and Ridjanovic, 1984] M. L. Brodie and D. Ridjanovic. On the design and specification of database transactions. In *On Conceptual Modelling*, pages 277–306. Springer-Verlag, 1984.

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

[Calvanese *et al.*, 1994] D. Calvanese, M. Lenzerini, and D. Nardi. A unified framework for class based representation formalisms. In *Proc. of KR-94*, pages 109–120, 1994.

[Catarci and Lenzerini, 1993] T. Catarci and M. Lenzerini. Representing and using interschema knowledge in cooperative information systems. *J. of Intelligent and Cooperative Information Systems*, 2(4):375–398, 1993.

[De Giacomo and Lenzerini, 1995] G. De Giacomo and M. Lenzerini. 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*, 1995.

[De Giacomo and Lenzerini, 1994] G. De Giacomo and M. Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proc. of AAAI-94*, pages 205–212, 1994.

[Nebel, 1991] B. Nebel. Terminological cycles: Semantics and computational properties. In *Principles of Semantic Networks*, pages 331–361. Morgan Kaufmann, Los Altos, 1991.

[Piza *et al.*, 1992] B. Piza, K. D. Schewe, and J. W. Schmidt. Term subsumption with type constructors. In *Proc. of CIKM-92*, pages 449–456, 1992.

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

[Schmolze, 1989] J. G. Schmolze. Terminological knowledge representation systems supporting n-ary terms. In *Proc. of KR-89*, pages 432–443, 1989.

[Schreiber *et al.*, 1993] G. Schreiber, B. Wielinga, and J. Breuker. *KADS: A principled approach to knowledge-based system development.* Academic Press, 1993.

[Smith and Smith, 1977] J. M. Smith and D. C. P. Smith. Database abstractions: Aggregation and generalization. *ACM Transactions on Database Systems*, 2(2):105–133, 1977.

[Woods and Schmolze, 1992] W. A. Woods and J. G. Schmolze. The KL-ONE family. In *Semantic Networks in Artificial Intelligence*, pages 133–178. Pergamon Press, 1992.

[Woods, 1975] W. A. Woods. What's in a link: Foundations for semantic networks. In *Representation and Understanding: Studies in Cognitive Science*, pages 35–82. Academic Press, 1975.