Full Satisfiability of UML Class Diagrams

Alessandro Artale, Diego Calvanese, and Angélica Ibáñez-García

KRDB Research Centre, Free University of Bozen-Bolzano, Italy
{artale,calvanese,ibanezgarcia}@inf.unibz.it

Abstract. UML class diagrams (UCDs) are the de-facto standard formalism for the analysis and design of information systems. By adopting formal language techniques to capture constraints expressed by UCDs one can exploit automated reasoning tools to detect relevant properties, such as schema and class satisfiability and subsumption between classes. Among the reasoning tasks of interest, the basic one is detecting *full satisfiability* of a diagram, i.e., whether there exists an instantiation of the diagram where *all* classes and associations of the diagram are non-empty and all the constraints of the diagram are respected. In this paper we establish tight complexity results for full satisfiability for various fragments of UML class diagrams. This investigation shows that the full satisfiability problem is ExpTime-complete in the full scenario, NP-complete if we drop isa between relationships, and NLogSpace-complete if we further drop covering over classes.[1]

Keywords: Reasoning over Conceptual Models, Description Logics, Complexity Analysis.

1 Introduction

UML (Unified Modeling Language - `http://www.omg.org/spec/UML/`) is the de-facto standard formalism for the analysis and design of information systems. One of the most important components of UML are *class diagrams* (UCDs). UCDs describe the domain of interest in terms of objects organized in classes and associations between them. The semantics of UCDs is by now well established, and several works propose to represent it using various kinds of formal languages, e.g., [2,3,4,5,6,7]. Thus, one can in principle reason on UCDs. The reasoning tasks that one is interested in are, e.g., *subsumption* between two classes, i.e., the fact that each instance of one class is necessarily also an instance of another class, *satisfiability* of a *specific* class (or association) in the diagram, i.e., the fact that the information encoding that class (or association) in the diagram is not contradictory, *diagram satisfiability*, which requires that *at least one* class in the diagram is satisfiable, and *full satisfiability* of the diagram [8,9], i.e., the fact that there exists an instantiation of the diagram where *all* classes and associations of the diagram are non-empty.

[1] A preliminary and shortened version of this paper has been presented at the 2009 Int. Workshop on Logic in Databases (LID 2009), with informal proceedings printed as a technical report [1].

The latter property is of importance since the presence of some unsatisfiable class or association actually means either that the diagram contains unnecessary information that should be removed, or that there is some modeling error that leads to the loss of satisfiability.

In this paper, we adopt the well established formalization of UCDs in terms of Description Logics (DLs). DLs [10] are decidable logics that are specifically tailored for capturing various forms of conceptual data models (cf. [11,12,13,14,15,16,5]), and they allow one to exploit state-of-the-art DL reasoners [17] to provide automated reasoning support over such data models.

The complexity of reasoning over UCDs has been addressed in [5] where it has been shown that in the presence of the standard UML/EER constructs, such as ISA, disjointness and covering between entities and associations, cardinality constraints (also called participation constraints) for associations, and multiplicity constraints for attributes makes checking *class satisfiability* and *schema satisfiability* ExpTime-complete. This result has been strengthened in [6] to UCDs[2] with simple ISA between associations (and both disjointness and completeness constraints on class hierarchies only), where it was also shown that by dropping ISA between associations reasoning becomes NP-complete, and by further forbidding completeness in class hierarchies it drops to NLogSpace-complete.

The only works that addressed explicitly the complexity of full satisfiability of UCDs are [8,9], which include a classification of UCDs based on *inconsistency triggers*. Each inconsistency trigger is a pattern for recognizing possible inconsistencies of the diagram based on the interaction between different modelling constraints. [8,9] introduce various algorithms for checking full satisfiability of UCDs with different expressive power, together with an analysis of their computational complexity (i.e., upper bounds are provided). In particular, checking full satisfiability in the following scenarios is showed to be in:

1. ExpTime, if the standard constructs are used;
2. NP, if ISA between associations and multiple and overwriting inheritance of attributes is dropped—i.e., each attribute has a fixed type;
3. P, if diagrams are further restricted by forbidding completeness constraints;
4. PSpace (instead ofExpTime), if standard constructs are uses (as in scenario 1) but types for attributes associated to sub-classes are sub-types of types for the respective attributes associated to super-classes;
5. NP and P in the scenarios 2 and 3, respectively, if we further allow for attributes with types restricted as in 4.

The main contributions of this paper can be summarised as follows:

– We show tight complexity results for checking full satisfiability proving that the problem is ExpTime-complete in the standard scenario 1, NP-complete in the scenario 2 and NLogSpace-complete (instead of P) in the scenario 3;
– We prove that full satisfiability in the scenario 4 is ExpTime-hard, thus showing that the PSpace algorithm presented in [8,9] must be incomplete.

[2] The results in [6] are formulated in terms of the Entity-Relationship model, but they also carry directly over to UML class diagrams.

Our results build on the formalization of UCDs in terms of DLs given in [5,6]. In fact, our upper bounds for full satisfiability are an almost direct consequence of the corresponding upper bounds of the DL formalization. On the other hand, the obtained lower bounds for full satisfiability are more involved, and in some cases require a careful analysis of the corresponding proof for class satisfiability.

The rest of the paper is organized as follows. In Section 2, we briefly introduce the DL $\mathcal{ALC}$, on which we base our results, and show that full satisfiability in $\mathcal{ALC}$ is ExpTime-complete. In Section 3, we recall the FOL semantics of UCDs. In Sections 4 and 5, we provide our results on full satisfiability for various variants of UCDs. Finally, in Section 6, we draw some conclusions.

2 Full Satisfiability in the Description Logic $\mathcal{ALC}$

We start by studying *full satisfiability* for the DL $\mathcal{ALC}$, one of the basic variants of DLs [10]. The basic elements of $\mathcal{ALC}$ are atomic concepts and roles, denoted by A and P, respectively. Complex concepts C, D are defined as follows:

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

The semantics of $\mathcal{ALC}$, as usual in DLs, is specified in terms of *interpretations*. An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, with a non-empty *domain* $\Delta^\mathcal{I}$ and an *interpretation function* $\cdot^\mathcal{I}$, assigns to each concept C a subset $C^\mathcal{I}$ of $\Delta^\mathcal{I}$, and to each role name P a binary relation $P^\mathcal{I}$ in $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$ such that the following conditions are satisfied:

$$\begin{array}{rl} A^\mathcal{I} \subseteq \Delta^\mathcal{I}, & (C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}, \\ (\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}, & (\exists P.C)^\mathcal{I} = \{a \in \Delta^\mathcal{I} \mid \exists b.\, (a,b) \in P^\mathcal{I} \wedge b \in C^\mathcal{I}\}\,. \end{array}$$

We use the standard abbreviations $C_1 \sqcup C_2 := \neg(\neg C_1 \sqcap \neg C_2)$, and $\forall P.C := \neg\exists P.\neg C$, with the corresponding semantics.

An $\mathcal{ALC}$ *terminological box* (TBox) $\mathcal{T}$ is a finite set of (concept inclusion) *assertions* of the form $C \sqsubseteq D$. An interpretation $\mathcal{I}$ *satisfies* an assertion of the form $C \sqsubseteq D$ if and only if $C^\mathcal{I} \subseteq D^\mathcal{I}$. A TBox $\mathcal{T}$ is *satisfiable* if there is an interpretation $\mathcal{I}$, called a *model* of $\mathcal{T}$, that satisfies every assertion in $\mathcal{T}$. A concept C is *satisfiable w.r.t. a TBox* $\mathcal{T}$ if there is a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^\mathcal{I} \neq \emptyset$. It can be shown that TBox satisfiability and concept satisfiability w.r.t. a TBox are reducible to each other in polynomial time. Moreover, reasoning w.r.t. $\mathcal{ALC}$ TBoxes is ExpTime-complete (see e.g., [10]).

We now define the notion of *full satisfiability* of a TBox and show that for $\mathcal{ALC}$ it has the same complexity as classical satisfiability.

Definition 1 (TBox Full Satisfiability). *An $\mathcal{ALC}$ TBox $\mathcal{T}$ is said to be* fully satisfiable *if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $A^\mathcal{I} \neq \emptyset$, for every atomic concept A in $\mathcal{T}$. We say that $\mathcal{I}$ is a* full model *of $\mathcal{T}$.*

We first prove that full satisfiability in $\mathcal{ALC}$ is ExpTime-hard.

Lemma 2. *Concept satisfiability w.r.t. $\mathcal{ALC}$ TBoxes can be linearly reduced to full satisfiability of $\mathcal{ALC}$ TBoxes.*

Proof. Let $\mathcal{T}$ be an $\mathcal{ALC}$ TBox and C an $\mathcal{ALC}$ concept. As pointed out in [18], C is satisfiable w.r.t. $\mathcal{T}$ if and only if $C \sqcap A_{\mathcal{T}}$ is satisfiable w.r.t. the TBox $\mathcal{T}_1$ consisting of the single assertion, $A_{\mathcal{T}} \sqsubseteq \bigsqcap_{C_1 \sqsubseteq C_2 \in \mathcal{T}} (\neg C_1 \sqcup C_2) \sqcap \bigsqcap_{1 \leq i \leq n} \forall P_i.A_{\mathcal{T}}$, where $A_{\mathcal{T}}$ is a fresh atomic concept and $P_1, \ldots, P_n$ are all the atomic roles in $\mathcal{T}$ and C. In order to reduce the latter problem to full satisfiability, we extend $\mathcal{T}_1$ to $\mathcal{T}_2 = \mathcal{T}_1 \cup \{A_C \sqsubseteq C \sqcap A_{\mathcal{T}}\}$, with A_C a fresh atomic concept, and prove that: $C \sqcap A_{\mathcal{T}}$ is satisfiable w.r.t. $\mathcal{T}_1$ if and only if $\mathcal{T}_2$ is fully satisfiable.
"$\Rightarrow$" Let $\mathcal{I}$ be a model of $\mathcal{T}_1$ such that $(C \sqcap A_{\mathcal{T}})^{\mathcal{I}} \neq \emptyset$. We construct an interpretation of $\mathcal{T}_2$, $\mathcal{J} = (\Delta^{\mathcal{I}} \cup \{d^{top}\}, \cdot^{\mathcal{J}})$, with $d^{top} \notin \Delta^{\mathcal{I}}$, such that:

$A_{\mathcal{T}}^{\mathcal{J}} = A_{\mathcal{T}}^{\mathcal{I}}, \qquad A_C^{\mathcal{J}} = (C \sqcap A_{\mathcal{T}})^{\mathcal{I}},$

$A^{\mathcal{J}} = A^{\mathcal{I}} \cup \{d^{top}\},$ for each atomic concept A in $\mathcal{T}$ and C,

$P^{\mathcal{J}} = P^{\mathcal{I}},$ for each atomic role P in $\mathcal{T}$ and C.

Obviously, the extension of every atomic concept is non-empty in $\mathcal{J}$. Next, we show that $\mathcal{J}$ is a model of $\mathcal{T}_2$, by relying on the fact (easily proved by structural induction) that $D^{\mathcal{I}} \subseteq D^{\mathcal{J}}$, for each subconcept D of concepts in $\mathcal{T}_1$ or of C. Then, it is easy to show that $\mathcal{J}$ satisfies the two assertion in $\mathcal{T}_2$.

"$\Leftarrow$" Conversely, every *full model* $\mathcal{J}$ of $\mathcal{T}_2$ is also a model of $\mathcal{T}_1$ with $(C \sqcap A_{\mathcal{T}})^{\mathcal{J}} \neq \emptyset$, as $A_C^{\mathcal{J}} \subseteq (C \sqcap A_{\mathcal{T}})^{\mathcal{J}}$. □

Theorem 3. *Full satisfiability of $\mathcal{ALC}$ TBoxes is* ExpTime*-complete.*

Proof. The ExpTime membership is straightforward since full satisfiability of an $\mathcal{ALC}$ TBox $\mathcal{T}$ can be reduced to satisfiability of the TBox $\mathcal{T} \cup \bigcup_{1 \leq i \leq n} \{\top \sqsubseteq \exists P'.A_i\}$, where $A_1, \ldots, A_n$ are all the atomic concepts in $\mathcal{T}$, and P' is a fresh atomic role. The ExpTime-hardness follows from Lemma 2. □

We now modify the reduction of Lemma 2 so that it applies also to *primitive* $\mathcal{ALC}^-$ TBoxes, i.e., TBoxes that contain only assertions of the form:

$$A \sqsubseteq B, \qquad A \sqsubseteq \neg B, \qquad A \sqsubseteq B \sqcup B', \qquad A \sqsubseteq \forall P.B, \qquad A \sqsubseteq \exists P.B,$$

where A, B, B' are atomic concepts, and P is an atomic role.

Theorem 4. *Full satisfiability of primitive $\mathcal{ALC}^-$ TBoxes is* ExpTime*-complete.*

Proof. The ExpTime membership follows from Theorem 3. For proving the ExpTime-hardness, we use a result in [5] showing that concept satisfiability in $\mathcal{ALC}$ can be reduced to atomic concept satisfiability w.r.t. primitive $\mathcal{ALC}^-$ TBoxes. Let $\mathcal{T}^- = \{A_j \sqsubseteq D_j \mid 1 \leq j \leq m\}$ be a primitive $\mathcal{ALC}^-$ TBox, and A_0 an atomic concept. By the proof of Lemma 2, we have that A_0 is satisfiable w.r.t. $\mathcal{T}^-$ if and only if the TBox $\mathcal{T}_2'$ consisting of the assertions

$$A_{\mathcal{T}^-} \sqsubseteq \bigsqcap_{A_j \sqsubseteq D_j \in \mathcal{T}^-} (\neg A_j \sqcup D_j) \sqcap \bigsqcap_{1 \leq i \leq n} \forall P_i.A_{\mathcal{T}^-}, \qquad A_0' \sqsubseteq A_0 \sqcap A_{\mathcal{T}^-},$$

is fully satisfiable, with $A_{\mathcal{T}^-}$, A'_0 fresh atomic concepts.

$\mathcal{T}'_2$ is not a primitive $\mathcal{ALC}^-$ TBox, but it is equivalent to the TBox containing the assertions:

$$\begin{array}{lll} A'_0 \sqsubseteq A_{\mathcal{T}^-} & A_{\mathcal{T}^-} \sqsubseteq \neg A_1 \sqcup D_1 & A_{\mathcal{T}^-} \sqsubseteq \forall P_1 . A_{\mathcal{T}^-} \\ & \vdots & \vdots \\ A'_0 \sqsubseteq A_0 & A_{\mathcal{T}^-} \sqsubseteq \neg A_m \sqcup D_m & A_{\mathcal{T}^-} \sqsubseteq \forall P_n . A_{\mathcal{T}^-}, \end{array}$$

Finally, to get a primitive $\mathcal{ALC}^-$ TBox, $\mathcal{T}_2^-$, we replace each assertion of the form $A_{\mathcal{T}^-} \sqsubseteq \neg A_j \sqcup D_j$ by $A_{\mathcal{T}^-} \sqsubseteq B_j^1 \sqcup B_j^2$, $B_j^1 \sqsubseteq \neg A_j$, and $B_j^2 \sqsubseteq D_j$, with B_j^1 and B_j^2 fresh atomic concepts, for $j \in \{1, \ldots, m\}$.

We show now that $\mathcal{T}'_2$ is fully satisfiable iff $\mathcal{T}_2^-$ is fully satisfiable:

($\Rightarrow$) Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a full model of $\mathcal{T}'_2$. We extend $\mathcal{I}$ to an interpretation $\mathcal{J}$ of $\mathcal{T}_2^-$. Let $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}} \cup \{d^+, d^-\}$, with $\{d^+, d^-\} \cap \Delta^{\mathcal{I}} = \emptyset$, and define $\cdot^{\mathcal{J}}$ as follows:

$$\begin{aligned} A_{\mathcal{T}^-}^{\mathcal{J}} &= A_{\mathcal{T}^-}^{\mathcal{I}}, \quad {A'_0}^{\mathcal{J}} = {A'_0}^{\mathcal{I}}, \\ A^{\mathcal{J}} &= A^{\mathcal{I}} \cup \{d^+\}, \quad \text{for every other atomic concept } A \text{ in } \mathcal{T}'_2, \\ {B_j^1}^{\mathcal{J}} &= (\neg A_j)^{\mathcal{J}} \text{ and } {B_j^2}^{\mathcal{J}} = D_j^{\mathcal{J}}, \quad \text{for each } A_{\mathcal{T}^-} \sqsubseteq B_j^1 \sqcup B_j^2 \in \mathcal{T}_2^-, \\ P^{\mathcal{J}} &= P^{\mathcal{I}} \cup \{(d^+, d^+)\}, \quad \text{for each atomic role } P \text{ in } \mathcal{T}_2^-. \end{aligned}$$

It is easy to see that $\mathcal{J}$ is a full model of $\mathcal{T}_2^-$.

($\Leftarrow$) Trivial, since every model of $\mathcal{T}_2^-$ is a model of $\mathcal{T}'_2$. □

3 Formalizing UML Class Diagrams

In this section, we briefly describe UCDs and provide their semantics in terms of First Order Logic (the formalization adopted here is based on previous presentations in [5,15]).

A *class* in UCDs denotes a set of objects with common features. Formally, a class C corresponds to a unary predicate C. An n-ary *association* (also called *relationship*) in UCDs represents a relation between instances $n \geq 2$ classes. Names of associations (as names of classes) are unique in a UCD. A binary association between two classes C_1 and C_2 is graphically rendered as in Fig. 1. The *multiplicity* constraint $n_l..n_u$ (also called participation constraint) written on one end of the binary association specifies that each instance of the class C_1 participates at least n_l times and at most n_u times in the association R, and the multiplicity constraint $m_l..m_u$ specifies an analogous constraint for each instance of the class C_2. When a multiplicity constraint is omitted, it is intended to be $0..*$. Formally, an association R between the classes C_1 and C_2 is captured by a binary predicate R that satisfies the FOL axiom $\forall x_1, x_2 . (R(x_1, x_2) \rightarrow C_1(x_1) \wedge C_2(x_2))$, while multiplicities are formalized by the following FOL assertions:

$$\begin{aligned} &\forall x . (C_1(x) \rightarrow \exists_{\geq n_l} y . R(x, y) \wedge \exists_{\leq n_u} y . R(x, y)) \\ &\forall y . (C_2(y) \rightarrow \exists_{\geq m_l} x . R(x, y) \wedge \exists_{\leq m_u} x . R(x, y)), \end{aligned}$$

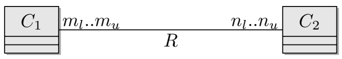

Fig. 1. Binary association

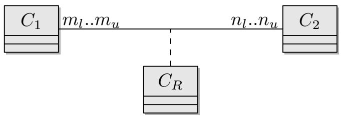

Fig. 2. Binary association with related class

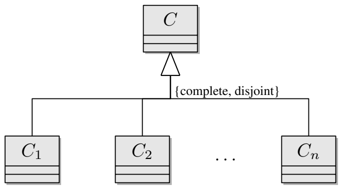

Fig. 3. A class hierarchy in UML

where we use counting quantifiers to abbreviate the FOL formula encoding the multiplicity constraints.

A more general form of multiplicity is the so called *refinement* of multiplicity constraints for sub-classes participating in associations. With such a construct we are able to change (and thus refine) the multiplicity constraints for sub-classes. Refinement involving a binary association, R, between classes C_1 and C_2, and a sub-class of C_1, say C_1', can be formalized with the following FOL axioms:

$$\forall x.\, (C_1'(x) \rightarrow C_1(x)), \qquad \forall x.\, (C_1'(x) \rightarrow \exists_{\geq n_l'} y.\, R(x,y) \wedge \exists_{\leq n_u'} y.\, R(x,y)).$$

An *association class* describes properties of the association, such as attributes, operations, etc. (see Fig. 2). A binary association between classes C_1 and C_2 with a related association class C_R is formalized in FOL by reifying the association into a unary predicate C_R with two binary predicates P_1, P_2, one for each component of the association. We enforce the following semantics, for $i \in \{1,2\}$:

$$\begin{array}{r}\forall x.(C_R(x) \rightarrow \exists y.\, P_i(x,y)),\\ \forall x,y.(C_R(x) \wedge P_i(x,y) \rightarrow C_i(y)),\\ \forall x,y,y'.(C_R(x) \wedge P_i(x,y) \wedge P_i(x,y') \rightarrow y = y'),\\ \forall y_1,y_2,x,x'.(C_R(x) \wedge C_R(x') \wedge (\bigwedge_{i\in\{1,2\}} P_i(x,y_i) \wedge P_i(x',y_i)) \rightarrow x = x').\end{array}$$

For associations with a related class, the multiplicity constraints are formalized by the following FOL assertions:

$$\begin{array}{l}\forall y_1.(C_1(y_1) \rightarrow \exists_{\geq n_l} x.\, (C_R(x) \wedge P_1(x,y_1)) \wedge \exists_{\leq n_u} x.\, (C_R(x) \wedge P_1(x,y_1)))\,,\\ \forall y_2.(C_2(y_2) \rightarrow \exists_{\geq m_l} x.\, (C_R(x) \wedge P_2(x,y_2)) \wedge \exists_{\leq m_u} x.\, (C_R(x) \wedge P_2(x,y_2)))\,.\end{array}$$

Classes can have *attributes*, formalized similarly to binary associations, relating the class with values of a given type. As for associations, we can specify multiplicity constraints over attributes.

A *generalization* (called also ISA constraint) between two classes C_1 and C, formalized as $\forall x.\, C_1(x) \rightarrow C(x)$, specifies that each instance of C_1 is also an

Table 1. Complexity of Full Satisfiability in UML (sub)languages

Language	**Constraints**						Complexity of Full Satisfiability
	Classes			Associations/Attributes			
	ISA	disjoint	complete	ISA	multiplicity	refinement	
UCD$_{full}$	✓	✓	✓	✓	✓	✓	ExpTime [Th.7]
UCD$_{bool}$	✓	✓	✓	✗	✓	✓	NP [Th.9]
UCD$_{ref}$	✓	✓	✗	✗	✓	✓	NLogSpace [Th.11]

instance of C. Several generalizations can be grouped together to form a class *hierarchy*, as shown in Fig. 3. Such a hierarchy is formally captured by means of the FOL axioms $\forall x. C_i(x) \rightarrow C(x)$ for $i \in \{1, \ldots, n\}$. *Disjointness* and *completeness* constraints can also be enforced on a class hierarchy, by adding suitable labels to the diagram. *Disjointness* among the classes $C_1, \ldots, C_n$ is expressed by $\forall x. C_i(x) \rightarrow \bigwedge_{j=i+1}^{n} \neg C_j(x)$, for $i \in \{1, \ldots, n-1\}$. The *completeness* constraint, expressing that each instance of C is an instance of at least one of $C_1, \ldots, C_n$, is captured by $\forall x. C(x) \rightarrow \bigvee_{i=1}^{n} C_i(x)$. We can also have generalization, disjointness and completeness constraints between associations and between association classes with the obvious semantics.

In this paper, we denote with UCD$_{full}$ the class diagram language that comprises all the standard constructs as discussed above (i.e., what we called scenario 1 in Section 1). With UCD$_{bool}$ we denote the language without generalization between associations (i.e., scenario 2 in Section 1), and with UCD$_{ref}$ we further drop completeness constraints over classes (i.e., scenario 3 in Section 1). The constructors allowed in these languages are summarized in Table 1, together with the tight complexity results obtained in this paper.

4 Full Satisfiability of UML Class Diagrams

Three notions of UCD satisfiability have been proposed in the literature [19,5,6,20,9]. First, *diagram satisfiability* refers to the existence of a *model*, i.e., a FOL interpretation that satisfies all the FOL assertions associated to the diagram and where at least one class has a nonempty extension. Second, *class satisfiability* refers to the existence of a model of the diagram where the given class has a nonempty extension. Third, we can check whether there is a model of an UML diagram that satisfies *all* classes and *all* relationships in a diagram. This last notion of satisfiability, referred here as *full satisfiability* and introduced in [8,9] is thus stronger than diagram satisfiability, since a model of a diagram that satisfies all classes is, by definition, also a model of that diagram.

Definition 5 (UML Full Satisfiability). *A UCD, $\mathcal{D}$, is* fully satisfiable *if there is a FOL interpretation, $\mathcal{I}$, that satisfies all the constraints expressed in $\mathcal{D}$ and such that $C^{\mathcal{I}} \neq \emptyset$ for every class C in $\mathcal{D}$, and $R^{\mathcal{I}} \neq \emptyset$ for every association R in $\mathcal{D}$. We say that $\mathcal{I}$ is a* full model *of $\mathcal{D}$.*

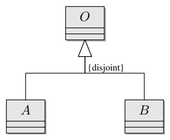

Fig. 4. Encoding of $A \sqsubseteq \neg B$

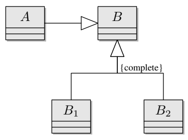

Fig. 5. Encoding of $A \sqsubseteq B_1 \sqcup B_2$

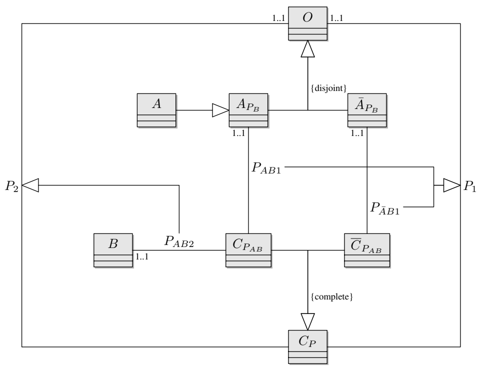

Fig. 6. Encoding of $A \sqsubseteq \forall P.B$

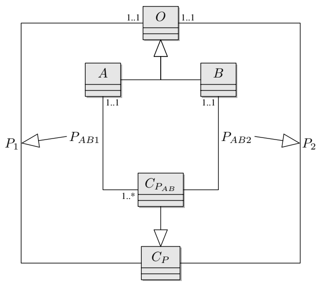

Fig. 7. Encoding of $A \sqsubseteq \exists P.B$

We now address the complexity of full satisfiability for UCDs with the standard set of constructs, i.e., UCD$_{full}$. For the lower bounds, we use the results presented in Section 2 and reduce full satisfiability of primitive $\mathcal{ALC}^-$ TBoxes to full satisfiability of UCD$_{full}$. This reduction is based on the ones used in [5,6] for the lower complexity bound of schema satisfiability in the extended Entity-Relationship model, but the proof of their correctness is more involved here.

Given a primitive $\mathcal{ALC}^-$ TBox $\mathcal{T}$, construct a UCD$_{full}$ diagram $\Sigma(\mathcal{T})$ as follows: for each atomic concept A in $\mathcal{T}$, introduce a class A in $\Sigma(\mathcal{T})$. Additionally, introduce a class O that generalizes (possibly indirectly) all the classes in $\Sigma(\mathcal{T})$ that encode an atomic concept in $\mathcal{T}$. For each atomic role P, introduce a class C_P, which reifies the binary relation P. Further, introduce two functional associations P_1, and P_2 that represent, respectively, the first and second component of P. The assertions in $\mathcal{T}$ are encoded as follows:

- For each assertion of the form $A \sqsubseteq B$, introduce a generalization between the classes A and B.
- For each assertion of the form $A \sqsubseteq \neg B$, construct the hierarchy in Fig. 4.
- For each assertion of the form $A \sqsubseteq B_1 \sqcup B_2$, introduce an *auxiliary* class B, and construct the diagram shown in in Fig. 5.
- For each assertion of the form $A \sqsubseteq \forall P.B$, add the auxiliary classes $C_{P_{AB}}$, $\overline{C}_{P_{AB}}$, A_{P_B}, and $\bar{A}_{P_B}$, and the associations P_{AB1}, $P_{\bar{A}B1}$, and P_{AB2}, and construct the diagram shown in Fig. 6.

– For each assertion of the form $A \sqsubseteq \exists P.B$, add the auxiliary class $C_{P_{AB}}$ and the associations P_{AB1} and P_{AB2}, and construct the diagram shown in Fig. 7.

Notice that $\Sigma(\mathcal{T})$ is a UCD in UCD$_{full}$.

Lemma 6. *A primitive $\mathcal{ALC}^-$ TBox $\mathcal{T}$ is fully satisfiable if and only if the UCD $\Sigma(\mathcal{T})$, constructed as above, is fully satisfiable.*

Proof. "$\Leftarrow$" Let $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ be a full model of $\Sigma(\mathcal{T})$. We construct a full model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{T}$ by taking $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$. Further, for every concept name A and for every atomic role P in $\mathcal{T}$, we define respectively $A^{\mathcal{I}} = A^{\mathcal{J}}$ and $P^{\mathcal{I}} = (P_1^-)^{\mathcal{J}} \circ P_2^{\mathcal{J}}$ ($r_1 \circ r_2$ denotes the composition of two binary relations r_1 and r_2). Let us show that $\mathcal{I}$ satisfies every assertion in $\mathcal{T}$.

– For assertions of the form $A \sqsubseteq B$, $A \sqsubseteq \neg B$, and $A \sqsubseteq B_1 \sqcup B_2$, the statement easily follows from the construction of $\mathcal{I}$.
– For assertions of the form $A \sqsubseteq \forall P.B$ and $A \sqsubseteq \exists P.B$, the proof uses arguments similar to those in the proof of Lemma 1 in [6].

"$\Rightarrow$" Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be a full model of $\mathcal{T}$, and let *role*$(\mathcal{T})$ be the set of role names in $\mathcal{T}$. We extend $\mathcal{I}$ to an instantiation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ of $\Sigma(\mathcal{T})$, by assigning suitable extensions to the auxiliary classes and associations in $\Sigma(\mathcal{T})$. Let $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}} \cup \Gamma \cup \Lambda$, where: $\Lambda = \biguplus_{A \sqsubseteq \forall P.B \in \mathcal{T}} \{a_{A_{P_B}}, a_{\bar{A}_{P_B}}\}$, such that $\Delta^{\mathcal{I}} \cap \Lambda = \emptyset$, and $\Gamma = \biguplus_{P \in role(\mathcal{T})} \Delta_P$, with $\Delta_P = P^{\mathcal{I}} \cup \bigcup_{A \sqsubseteq \forall P.B \in \mathcal{T}} \{(a_{A_{P_B}}, b), (a_{\bar{A}_{P_B}}, \bar{o})\}$ where b is an arbitrary instance of B, and $\bar{o}$ an arbitrary element of $\Delta^{\mathcal{I}}$. We set $O^{\mathcal{J}} = \Delta^{\mathcal{I}} \cup \Lambda$, $A^{\mathcal{J}} = A^{\mathcal{I}}$ for each class A corresponding to an atomic concept in $\mathcal{T}$, and $C_P^{\mathcal{J}} = \Delta_P$ for each $P \in$ *role*$(\mathcal{T})$. Additionally, the extensions of the associations P_1 and P_2 are defined as follows: $P_1^{\mathcal{J}} = \{((o_1, o_2), o_1) \mid (o_1, o_2) \in C_P^{\mathcal{J}}\}$, $P_2^{\mathcal{J}} = \{((o_1, o_2), o_2) \mid (o_1, o_2) \in C_P^{\mathcal{J}}\}$. We now show that $\mathcal{J}$ is a full model of $\Sigma(\mathcal{T})$.

– For the portions of $\Sigma(\mathcal{T})$ due to TBox assertions of the form $A \sqsubseteq B$, $A \sqsubseteq \neg B$, and $A \sqsubseteq B_1 \sqcup B_2$, the statement follows from the construction of $\mathcal{J}$.
– For each TBox assertion in $\mathcal{T}$ of the form $A \sqsubseteq \forall P.B$, let us define the extensions for the *auxiliary* classes and associations as follows:

$$\begin{array}{ll}
A_{P_B}^{\mathcal{J}} = A^{\mathcal{I}} \cup \{a_{A_{P_B}}\}, & \bar{A}_{P_B}^{\mathcal{J}} = O^{\mathcal{J}} \setminus A_{P_B}^{\mathcal{J}}, \\
C_{P_{AB}}^{\mathcal{J}} = \{(o, o') \in C_P^{\mathcal{J}} \mid o \in A_{P_B}^{\mathcal{J}}\}, & \overline{C}_{P_{AB}}^{\mathcal{J}} = \{(o, o') \in C_P^{\mathcal{J}} \mid o \in \bar{A}_{P_B}^{\mathcal{J}}\}, \\
P_{AB1}^{\mathcal{J}} = \{((o, o'), o) \in P_1^{\mathcal{J}} \mid o \in A_{P_B}^{\mathcal{J}}\}, & P_{\bar{A}B1}^{\mathcal{J}} = \{((o, o'), o) \in P_1^{\mathcal{J}} \mid o \in \bar{A}_{P_B}^{\mathcal{J}}\}, \\
P_{AB2}^{\mathcal{J}} = \{((o, o'), o') \in P_2^{\mathcal{J}} \mid o \in A_{P_B}^{\mathcal{J}}\}. &
\end{array}$$

It is not difficult to see that $\mathcal{J}$ satisfies the fragment of $\Sigma(\mathcal{T})$ as shown in Fig. 6. It remains to show that each class and each association has a non-empty extension. This is clearly the case for classes that encode atomic concepts in $\mathcal{T}$. For the classes A_{P_B}, $\bar{A}_{P_B}$, $C_{P_{AB}}$, and $\overline{C}_{P_{AB}}$ we have that

$$a_{A_{P_B}} \in A_{P_B}^{\mathcal{J}}, \quad a_{\bar{A}_{P_B}} \in \bar{A}_{P_B}^{\mathcal{J}}, \quad (a_{A_{P_B}}, b) \in C_{P_{AB}}^{\mathcal{J}}, \quad (a_{\bar{A}_{P_B}}, \bar{o}) \in \overline{C}_{P_{AB}}^{\mathcal{J}}.$$

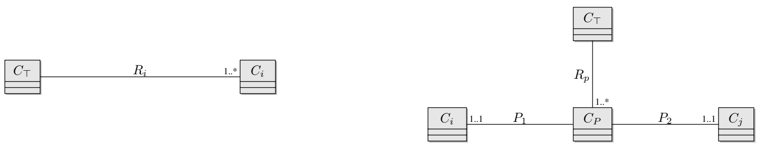

Fig. 8. Reducing UML full satisfiability to class satisfiability

For the associations P_1, P_2, P_{AB1}, P_{AB2}, and $P_{\bar{A}B1}$ we have that $((a_{A_{P_B}}, b), a_{A_{P_B}}) \in P^{\mathcal{J}}_{AB1} \subseteq P^{\mathcal{J}}_1$, $((a_{\bar{A}_{P_B}}, \bar{o}), a_{\bar{A}_{P_B}}) \in P^{\mathcal{J}}_{\bar{A}B1}$, $((a_{A_{P_B}}, b), b) \in P^{\mathcal{J}}_{AB2} \subseteq P^{\mathcal{J}}_2$.

– For each TBox assertion in $\mathcal{T}$ of the form $A \sqsubseteq \exists P.B$, let us define:

$$C^{\mathcal{J}}_{P_{AB}} = \{(o, o') \in C^{\mathcal{J}}_P \mid o \in A^{\mathcal{I}} \text{ and } o' \in B^{\mathcal{I}}\},$$
$$P^{\mathcal{J}}_{AB1} = \{((o, o'), o) \in P^{\mathcal{J}}_1 \mid (o, o') \in C^{\mathcal{J}}_{P_{AB}}\},$$
$$P^{\mathcal{J}}_{AB2} = \{((o, o'), o') \in P^{\mathcal{J}}_2 \mid (o, o') \in C^{\mathcal{J}}_{P_{AB}}\}.$$

We have that $C^{\mathcal{J}}_{P_{AB}} \neq \emptyset$ as there exists a pair $(a, b) \in \Delta_P$ with $a \in A^{\mathcal{I}}$, and $b \in B^{\mathcal{I}}$. Since $C^{\mathcal{J}}_{P_{AB}} \neq \emptyset$, we have that $P^{\mathcal{J}}_{AB1} \neq \emptyset$ and $P^{\mathcal{J}}_{AB2} \neq \emptyset$. □

Theorem 7. *Full satisfiability of UCD$_{full}$ diagrams is* ExpTime*-complete.*

Proof. We establish the upper bound by a reduction to class satisfiability in UCDs, which is known to be ExpTime-complete [5]. Given a UCD $\mathcal{D}$, with classes $C_1, \ldots, C_n$, we construct a UCD $\mathcal{D}'$ by adding to $\mathcal{D}$ a new class $C_\top$ and new associations R_i, for $i \in \{1, \ldots, n\}$, as shown in the left part of Fig. 8. Furthermore, to check that every association is populated we use reification, i.e., we replace each association P in the diagram $\mathcal{D}$ between the classes C_i and C_j (such that neither C_i nor C_j is constrained to participate at least once to P) with a class C_P and two functional associations P_1 and P_2 to represent each component of P. Finally, we add the constraints shown in the right part of Fig. 8. Intuitively, we have that if there is a model $\mathcal{I}$ of the extended diagram $\mathcal{D}'$ in which $C^{\mathcal{I}}_\top \neq \emptyset$, then the multiplicity constraint $1..*$ on the association R_P forces the existence of at least one instance o of C_P. By the functionality of P_1 and P_2 there are at least two elements o_i and o_j, such that $o_i \in C^{\mathcal{I}}_i$, $o_j \in C^{\mathcal{I}}_j$, $(o, o_i) \in P^{\mathcal{I}}_1$ and $(o, o_j) \in P^{\mathcal{I}}_2$. Then, one instance of P can be the pair (o_i, o_j). Conversely, if there is a full model $\mathcal{J}$ of $\mathcal{D}$, it is easy to extend it to a model $\mathcal{I}$ of $\mathcal{D}'$ that satisfies $C_\top$.

The ExpTime-hardness follows from Lemma 6 and Theorem 4. □

Note that, the proof of the above theorem does not involve attributes. Thus, the ExpTime complexity result is valid for both scenarios 1 and 4 in Section 1.

5 Full Satisfiability of Restricted UML Class Diagrams

In this section, we investigate the complexity of the full satisfiability problem for the two sub-languages UCD$_{bool}$ and UCD$_{ref}$ defined in Section 3. By building on the techniques used for the satisfiability proofs in [6], we show that also in this case checking for full satisfiability does not change the complexity of the problem.

We consider first UCD$_{bool}$ diagrams, by showing that deciding full satisfiability is NP-complete. For the lower bound, we provide a polynomial reduction of the 3SAT problem (which is known to be NP-complete) to full satisfiability of UCD$_{bool}$ CDs.

Let an instance of 3SAT be given by a set $\phi = \{c_1, \ldots, c_m\}$ of 3-clauses over a finite set Π of propositional variables. Each clause is such that $c_i = \ell_i^1 \vee \ell_i^2 \vee \ell_i^3$, for $i \in \{1, \ldots, m\}$, where each ℓ_j^k is a literal, i.e., a variable or its negation. We construct an UCD$_{bool}$ diagram $\mathcal{D}_\phi$ as follows: $\mathcal{D}_\phi$ contains the classes $C_\phi, C_\top$, one class C_i for each clause $c_i \in \phi$, and two classes C_p and $C_{\neg p}$ for each variable $p \in \Pi$. To describe the constraints imposed by $\mathcal{D}_\phi$, we provide the corresponding DL inclusion assertions, since they are more compact to write than an UCD. For every $i \in \{1, \ldots, m\}$, $j \in \{1, 2, 3\}$, and $p \in \Pi$, we have the assertions

$$\begin{array}{rcc}
C_\phi \sqsubseteq C_\top & C_i \sqsubseteq C_\top & C_{l_i^j} \sqsubseteq C_i \\
C_p \sqsubseteq C_\top & C_\phi \sqsubseteq C_i & C_i \sqsubseteq C_{\ell_i^1} \sqcup C_{\ell_i^2} \sqcup C_{\ell_i^3} \\
C_{\neg p} \sqsubseteq C_\top & C_\top \sqsubseteq C_p \sqcup C_{\neg p} & C_{\neg p} \sqsubseteq \neg C_p
\end{array}$$

Clearly, the size of $\mathcal{D}_\phi$ is polynomial in the size of ϕ.

Lemma 8. *A set ϕ of 3-clauses is satisfiable if and only if the UCD$_{bool}$ class diagram $\mathcal{D}_\phi$, constructed as above, is fully satisfiable.*

Proof. "$\Rightarrow$" Let $\mathcal{J} \models \phi$. Define an interpretation $\mathcal{I} = (\{0,1\}, \cdot^\mathcal{I})$, with

$$C_\top^\mathcal{I} = \{0,1\} \qquad C_i^\mathcal{I} = C_{\ell_i^1}^\mathcal{I} \cup C_{\ell_i^2}^\mathcal{I} \cup C_{\ell_i^3}^\mathcal{I}, \quad \text{for } c_i = \ell_i^1 \vee \ell_i^2 \vee \ell_i^3$$

$$C_\ell^\mathcal{I} = \begin{cases} \{1\}, & \text{if } \mathcal{J} \models \ell \\ \{0\}, & \text{otherwise} \end{cases} \qquad C_\phi^\mathcal{I} = C_1^\mathcal{I} \cap \cdots \cap C_m^\mathcal{I}.$$

Clearly, $C^\mathcal{I} \neq \emptyset$ for every class C representing a clause or a literal, and for $C = C_\top$. Moreover, as at least one literal ℓ_i^j in each clause is such that $\mathcal{J} \models \ell_i^j$, then $1 \in C_i^\mathcal{I}$ for every $i \in \{1, \ldots, m\}$, and therefore $1 \in C_\phi^\mathcal{I}$. It is straightforward to check that $\mathcal{I}$ satisfies $\mathcal{T}$.

"$\Leftarrow$" Let $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be a full model of $\mathcal{D}_\phi$. We construct a model $\mathcal{J}$ of ϕ by taking an element $o \in C_\phi^\mathcal{I}$, and setting, for every variable $p \in \Pi$, $\mathcal{J} \models p$ if and only if $o \in C_p^\mathcal{I}$. Let us show that $\mathcal{J} \models \phi$. Indeed, for each $i \in \{1, \ldots, m\}$, since $o \in C_\phi^\mathcal{I}$ and by the generalization $C_\phi \sqsubseteq C_i$, we have that $o \in C_i^\mathcal{I}$, and by the completeness constraint $C_i \sqsubseteq C_{\ell_i^1} \sqcup C_{\ell_i^2} \sqcup C_{\ell_i^3}$, there is some $j_i \in \{1, 2, 3\}$ such that $o \in C_{\ell_i^{j_i}}$. If $\ell_i^{j_i}$ is a variable, then $\mathcal{J} \models \ell_i^{j_i}$ by construction, and thus $\mathcal{J} \models c_i$. Otherwise, if $\ell_i^{j_i} = \neg p$ for some variable p, then, by the disjointness constraint $C_{\neg p} \sqsubseteq \neg C_p$, we have that $o \notin C_p^\mathcal{I}$. Thus, $\mathcal{J} \models \neg p$, and therefore, $\mathcal{J} \models c_i$. $\square$

Theorem 9. *Full satisfiability of UCD$_{bool}$ is* NP*-complete*

Proof. To prove the NP upper bound, we reduce full satisfiability to class satisfiability, which, for the case of UCD$_{bool}$, is known to be in NP [6]. We use an encoding similar to the one used in the proof of Theorem 7 (see Fig. 8).

The NP-hardness follows from Lemma 8. □

We turn now to UCD$_{ref}$ class diagrams and show that full satisfiability in this case is NLogSpace-complete. We provide a reduction of the reachability problem on (acyclic) directed graphs, which is known to be NLogSpace-complete (see e.g., [21]) to the complement of full satisfiability of UCD$_{ref}$ CDs.

Let $G = (V, E, s, t)$ be an instance of reachability, where V is a set of vertices, $E \subseteq V \times V$ is a set of directed edges, s is the start vertex, and t the terminal vertex. We construct an UCD$_{ref}$ diagram $\mathcal{D}_G$ from G as follows:

- $\mathcal{D}_G$ has two classes C_v^1 and C_v^2, for each vertex $v \in V \setminus \{s\}$, and one class C_s corresponding to the start vertex s.
- For each edge $(u, v) \in E$ with $u \neq s$ and $v \neq s$, $\mathcal{D}_G$ contains the following constraints (again expressed as DL inclusion assertions): $C_u^1 \sqsubseteq C_v^1$, $C_u^2 \sqsubseteq C_v^2$.
- For each edge $(s, v) \in E$, $\mathcal{D}_G$ contains the following constraints: $C_s \sqsubseteq C_v^1$, $C_s \sqsubseteq C_v^2$.
- For each edge $(u, s) \in E$, $\mathcal{D}_G$ contains the following constraints: $C_u^1 \sqsubseteq C_s$, $C_u^2 \sqsubseteq C_s$.
- The classes C_t^1 and C_t^2 are constrained to be disjoint in $\mathcal{D}$, expressed by: $C_t^1 \sqsubseteq \neg C_t^2$.

The following lemma establishes the correctness of the reduction.

Lemma 10. *t is reachable from s in G iff $\mathcal{D}_G$ is not fully satisfiable.*

Proof. "⇒" Let $\pi = v_1, \ldots, v_n$ be a path in G with $v_1 = s$ and $v_n = t$. We claim that the class C_s in the constructed diagram $\mathcal{D}_G$ is unsatisfiable. Suppose otherwise that there is a model $\mathcal{I}$ of $\mathcal{D}_G$ with $o \in C_s^{\mathcal{I}}$, for some $o \in \Delta^{\mathcal{I}}$. From π, a number of generalization constraints hold in $\mathcal{D}_G$, i.e., $C_s^{\mathcal{I}} \subseteq {C_t^1}^{\mathcal{I}}$ and $C_s^{\mathcal{I}} \subseteq {C_t^2}^{\mathcal{I}}$. Thus, we obtain that $o \in (C_t^1)^{\mathcal{I}}$ and $o \in (C_t^2)^{\mathcal{I}}$, which violates the disjointness between the classes C_t^1 and C_t^2, in contradiction to $\mathcal{I}$ being a model of $\mathcal{D}_G$. Hence, C_s is unsatisfiable, and therefore $\mathcal{D}_G$ is not fully satisfiable.

"⇐" Let us consider the contrapositive. Assume that t is not reachable from s in G. We construct a full model $\mathcal{I}$ of $\mathcal{D}_G$. Let $\Delta^{\mathcal{I}} = \{d_s\} \cup \bigcup_{v \in V \setminus \{s\}} \{d_v^1, d_v^2\}$. Define inductively a sequence of interpretations as follows:

$\mathcal{I}^0 = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}^0})$, such that: $C_s^{\mathcal{I}^0} = \{d_s\}$, ${C_v^i}^{\mathcal{I}^0} = \{d_v^i\}, \forall i \in \{1, 2\}, v \in V \setminus \{s\}$, $\mathcal{I}^{n+1} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}^{n+1}})$, such that: $C_s^{\mathcal{I}^{n+1}} = C_s^{\mathcal{I}^n} \cup \bigcup_{(u,s) \in E} ({C_u^1}^{\mathcal{I}^n} \cup {C_u^2}^{\mathcal{I}^n})$, ${C_v^i}^{\mathcal{I}^{n+1}} = {C_v^i}^{\mathcal{I}^n} \cup \bigcup_{(u,v) \in E,\, u \neq s} {C_u^i}^{\mathcal{I}^n} \cup \bigcup_{(s,v) \in E} C_s^{\mathcal{I}^n}$.

The definition induces a monotone operator over a complete lattice, and hence it has a fixed point. Let $\mathcal{I}$ be defined by such a fixed point. It is easy to check that $\mathcal{I}$ is such that for all $i \in \{1, 2\}$, and $u, v \in V \setminus \{s\}$ the following holds:

1. For each class C_v^i, we have that $d_v^i \in C_v^{i\mathcal{I}}$.
2. $d_s \in C_s^{\mathcal{I}}$.
3. For all $d \in \Delta^{\mathcal{I}}$, $d \in C_u^{i\mathcal{I}}$ implies $d \in C_v^{i\mathcal{I}}$ iff v is reachable from u in G.
4. For all $d_u^i \in \Delta^{\mathcal{I}}$, $d_u^i \in C_v^{j\mathcal{I}}$ for $i \neq j$ iff s is reachable from u in G, and v is reachable from s in G.
5. $d_s \in C_v^{i\mathcal{I}}$ iff v is reachable from s in G.

From (1) and (2) we have that all classes in $\mathcal{D}_G$ are populated in $\mathcal{I}$. It remains to show that $\mathcal{I}$ satisfies $\mathcal{D}_G$. A generalization between the classes C_u^i and C_v^i corresponds to the edge $(u, v) \in E$. This means that v is reachable from u in G, and therefore, by (3) we have that $C_u^{i\mathcal{I}} \subseteq C_v^{i\mathcal{I}}$. A similar argument holds for generalizations involving the class C_s. Furthermore, the classes C_t^1 and C_t^2 are disjoint under $\mathcal{I}$. To show this, suppose that there is an element $d \in \Delta^{\mathcal{I}}$ such that $d \in C_t^{1\mathcal{I}} \cap C_t^{2\mathcal{I}}$. Then by (5), $d \neq d_s$, as t is not reachable from s. Moreover, $d \neq d_v^i$ for all $i \in \{1, 2\}$ and $v \in V \setminus \{s\}$. Indeed, suppose w.l.o.g. that $i = 1$. Then, by (4), $d_v^1 \in C_t^{2\mathcal{I}}$ iff s is reachable from v, and t is reachable from s, which leads to a contradiction. Hence, $C_t^{1\mathcal{I}} \cap C_t^{2\mathcal{I}} = \emptyset$. □

Theorem 11. *Full-satisfiability of UCD$_{ref}$ class diagrams is* NLOGSPACE-*complete.*

Proof. The NLOGSPACE membership follows from the NLOGSPACE membership of class satisfiability [6], and a reduction similar to the one used in Theorem 9. Since NLOGSPACE = CONLOGSPACE (by the Immerman-Szelepcsényi theorem; see, e.g., [21]), and as the above reduction is logspace bounded, it follows that full consistency of UCD$_{ref}$ class diagrams is NLOGSPACE-hard. □

6 Conclusions

This paper investigates the problem of *full satisfiability* in the context of UML class diagrams, i.e., whether there is at least one model of the diagram where each class and association is non-empty. Our results (reported in Table 1) show that the complexity of checking full satisfiability is EXPTIME-complete both in the full scenario (UCD$_{full}$) and in the case where attributes are dropped, NP-complete if we drop ISA between relationships (UCD$_{bool}$), and NLOGSPACE-complete if we further drop covering over classes (UCD$_{ref}$), thus matching the complexity of the classical class diagram satisfiability check. These complexity bounds extend the ones presented in [6] for class/schema satisfiability to full satisfiability. We show a similar result also for the problem of checking the full satisfiability of a TBox expressed in the description logic $\mathcal{ALC}$. As a future work, we intend to investigate the problem under the finite model assumption.

Acknowledgements. This research has been partially supported by the FP7 ICT projects ACSI, contract n. 257593, and OntoRule, contract n. 231875.

References

1. Artale, A., Calvanese, D., Ibanez-Garcia, A.: Full satisfiability of UML class diagrams (extended abstract). Technical Report 127, Roskilde University Computer Science Research Reports. In: Proc. of the 2009 Int. Workshop on Logic in Databases (LID 2009) (2009)
2. Clark, T., Evans, A.S.: Foundations of the Unified Modeling Language. In: Duke, D., Evans, A. (eds.) Proc. of the 2nd Northern Formal Methods Workshop, Springer, Heidelberg (1997)
3. Evans, A., France, R., Lano, K., Rumpe, B.: Meta-modelling semantics of UML. In: Kilov, H. (ed.) Behavioural Specifications for Businesses and Systems. Kluwer Academic Publishers, Dordrecht (1999)
4. Harel, D., Rumpe, B.: Modeling languages: Syntax, semantics and all that stuff. Technical Report MCS00-16, The Weizmann Institute of Science, Rehovot, Israel (2000)
5. Berardi, D., Calvanese, D., De Giacomo, G.: Reasoning on UML class diagrams. Artificial Intelligence 168(1-2), 70–118 (2005)
6. Artale, A., Calvanese, D., Kontchakov, R., Ryzhikov, V., Zakharyaschev, M.: Reasoning over extended ER models. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, pp. 277–292. Springer, Heidelberg (2007)
7. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The *DL-Lite* family and relations. J. of Artificial Intelligence Research 36, 1–69 (2009)
8. Kaneiwa, K., Satoh, K.: Consistency checking algorithms for restricted UML class diagrams. In: Dix, J., Hegner, S.J. (eds.) FoIKS 2006. LNCS, vol. 3861, pp. 219–239. Springer, Heidelberg (2006)
9. Kaneiwa, K., Satoh, K.: On the complexities of consistency checking for restricted UML class diagrams. Theoretical Computer Science 411(2), 301–323 (2010)
10. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, Cambridge (2003)
11. Bergamaschi, S., Sartori, C.: On taxonomic reasoning in conceptual design. ACM Trans. on Database Systems 17(3), 385–422 (1992)
12. Borgida, A.: Description logics in data management. IEEE Trans. on Knowledge and Data Engineering 7(5), 671–682 (1995)
13. Artale, A., Cesarini, F., Soda, G.: Describing database objects in a concept language environment. IEEE Trans. on Knowledge and Data Engineering 8(2), 345–351 (1996)
14. Calvanese, D., Lenzerini, M., Nardi, D.: Description logics for conceptual data modeling. In: Chomicki, J., Saake, G. (eds.) Logics for Databases and Information Systems, pp. 229–264. Kluwer Academic Publishers, Dordrecht (1998)
15. Calvanese, D., Lenzerini, M., Nardi, D.: Unifying class-based representation formalisms. J. of Artificial Intelligence Research 11, 199–240 (1999)
16. Borgida, A., Brachman, R.J.: Conceptual modeling with description logics. In: [10], ch. 10, pp. 349–372

17. Möller, R., Haarslev, V.: Description logic systems. In: [10], ch. 8, pp. 282–305
18. Buchheit, M., Donini, F.M., Schaerf, A.: Decidable reasoning in terminological knowledge representation systems. J. of Artificial Intelligence Research 1, 109–138 (1993)
19. Lenzerini, M., Nobili, P.: On the satisfiability of dependency constraints in entity-relationship schemata. Information Systems 15(4), 453–461 (1990)
20. Jarrar, M., Heymans, S.: Towards pattern-based reasoning for friendly ontology debugging. Int. J. on Artificial Intelligence Tools 17(4), 607–634 (2008)
21. Papadimitriou, C.H.: Computational Complexity. Addison Wesley Publ. Co., Reading (1994)