

# Adding ABoxes to a Description Logic with Uniqueness Constraints via Path Agreements

Vitaliy L. Khizder<sup>†</sup>, David Toman<sup>‡</sup> and Grant Weddell<sup>‡</sup>

<sup>†</sup>Oracle Corporation

<sup>‡</sup>David R. Cheriton School of Computer Science  
University of Waterloo, Canada

**Abstract.** We now know that the addition of a concept constructor called a Path Functional Dependency (PFD) to the Boolean-complete description logic  $\mathcal{DLF}$  leads to undecidability of ABox consistency for  $\mathcal{DLF}$ . Consequently, we define a boundary condition for PFDs that enables the recovery of earlier  $\mathcal{DLF}$  complexity bounds for this problem. This is accomplished indirectly by adding an additional concept constructor for rooted path equality, which has the added benefit of increasing the utility of  $\mathcal{DLF}$  for reasoning about query containment problems in query optimization, in plan generation and in automated synthesis of web services.

## 1 Introduction

The introduction of web service environments with query and ontology languages such as SWRL [8] and OWL [9] make it necessary for agents to reason about query plans and about how to communicate the results of queries between agents. It is consequently imperative to allow identification constraints to be incorporated in ontologies to enable an agent to determine, e.g., that *there is at most one way of performing a currency conversion*, or that *items from a particular company are reliably identified by that company's item code*. Indeed, this situation is anticipated by extensive experience with SQL and the relational model of data in which keys and functional dependencies have been used for reasoning about properties of query plans related, e.g., to tuple identification.

In earlier work, we have explored how a general form of uniqueness constraint called a *Path Functional Dependency* (PFD) can be added to a Boolean-complete description logic called  $\mathcal{DLF}$ , a fragment of the OWL ontology language [14]. The resulting logic is called  $\mathcal{DLFD}$  and can be used to express, e.g., that *customers who have consulted with a manager can be reliably identified by their social insurance number*. In particular, this can be captured by the following inclusion dependency in  $\mathcal{DLFD}$ .

$$\text{Customer} \sqcap \forall \text{Consults}.\text{Manager} \sqsubseteq \text{Person} : \text{SIN} \rightarrow \text{Id}$$

To paraphrase: *If a customer has consulted with a manager, then no other person will share his or her social insurance number*. Later on, we show how this constraint can help with an important reformulation of a query.

Although  $\mathcal{DLFD}$  is particularly suited to capturing such meta-data, we have recently discovered that its ABox consistency problem is undecidable [22]. This is bad news since this greatly limits how the logic can then be used for reasoning about queries and services, e.g., in query reformulation. To remedy this, we define a *boundary* syntactic condition for PFDs that enables the recovery of earlier  $\mathcal{DLF}$  complexity bounds for

reasoning about ABox consistency. This is effectively accomplished by adding an additional concept constructor for rooted path equality along the lines pioneered by the description logic CLASSIC [3]. The addition of this constructor thereby obtains the logic  $\mathcal{DLFDE}$  which serves as the primary focus of this paper.

Our main contributions are as follows.

- We establish the equivalence of the ABox consistency problem for  $\mathcal{DLFD}$  and the logical implication problem for  $\mathcal{DLFDE}$ . Thus, and this was surprising to us, the latter is also undecidable. We also show that this equivalence continues to hold in the absence of any occurrences of PFDs.
- We define the logic  $\mathcal{DLFDE}^-$  which refines  $\mathcal{DLFDE}$  by introducing a syntactic boundary condition for PFDs, and show that both its ABox consistency problem and its logical implication problem are decidable and complete for EXPTIME.

## 1.1 Background and Related Work

In addition to the web ontology language OWL, description logics have been used extensively as a formal way of understanding a large variety of languages for specifying meta-data, including ER diagrams, UML class and object diagrams, relational database schema, etc. [18].

PFDs were first introduced and studied in the context of object-oriented data models [12, 24] as a way of capturing functional constraints such as keys and functional dependencies. An FD concept constructor was subsequently proposed and incorporated in CLASSIC [4], an early DL with a PTIME reasoning procedure, without changing the complexity of its implication problem. This was particularly noteworthy since CLASSIC also included a concept constructor for rooted path (in)equalities. The generalization of the FD constructor to PFDs alone leads to EXPTIME completeness of the implication problem [14]; this complexity remains unchanged in the presence of additional concept constructors common in rich DLs [23]. More recent work has shown the need to limit where PFDs may occur in concepts to avoid undecidability, in particular outside the scope of non-monotonic concept constructors such as negation, and that ABox consistency is undecidable regardless [22].

Calvanese and others have considered a DL with functional dependencies and a general form of keys added as additional varieties of dependencies, called a *key box* [6]. They show that their dialect is undecidable for DLs with inverse roles, but becomes decidable when unary functional dependencies are disallowed. This line of investigation is continued in the context of PFDs and inverse attributes, with analogous results [21]. We therefore disallow inverse attributes in this paper to exclude an already known cause for undecidability.

A form of key dependency with left hand side feature paths has been considered for a DL coupled with various concrete domains [15, 16]. In this case, the authors explore how the complexity of satisfaction is influenced by the selection of a concrete domain together with various syntactic restrictions on the key dependencies themselves.

PFDs have been used in a number of applications: in object-oriented schema diagnosis and synthesis [1, 2], in query optimization [13] and in the selection of indexing for a database [19]. Description logics have also been used for reasoning about query containment in the presence of rich database schema [5, 10].

SYNTAX	SEMANTICS: DEFN OF “ $(\cdot)^{\mathcal{I}}$ ”
$C ::= A$	$(A)^{\mathcal{I}} \subseteq \Delta$
$C_1 \sqcap C_2$	$(C_1)^{\mathcal{I}} \cap (C_2)^{\mathcal{I}}$
$\neg C$	$\Delta \setminus (C)^{\mathcal{I}}$
$\forall f.C$	$\{x : (f)^{\mathcal{I}}(x) \in (C)^{\mathcal{I}}\}$
$D ::= C$	
$C : \text{Pf}_1, \dots, \text{Pf}_k \rightarrow \text{Pf}$	$\{x : \forall y \in (C)^{\mathcal{I}}. \bigwedge_{i=1}^k (\text{Pf}_i)^{\mathcal{I}}(x) = (\text{Pf}_i)^{\mathcal{I}}(y) \Rightarrow (\text{Pf})^{\mathcal{I}}(x) = (\text{Pf})^{\mathcal{I}}(y)\}$
$E ::= C$	
$E_1 \sqcap E_2$	$(E_1)^{\mathcal{I}} \cap (E_2)^{\mathcal{I}}$
$\neg E$	$\Delta \setminus (E)^{\mathcal{I}}$
$\forall f.E$	$\{x : (f)^{\mathcal{I}}(x) \in (E)^{\mathcal{I}}\}$
$(\text{Pf}_1 = \text{Pf}_2)$	$\{x : (\text{Pf}_1)^{\mathcal{I}}(x) = (\text{Pf}_2)^{\mathcal{I}}(x)\}$

**Fig. 1.** SYNTAX AND SEMANTICS OF  $\mathcal{DLFDE}$ .

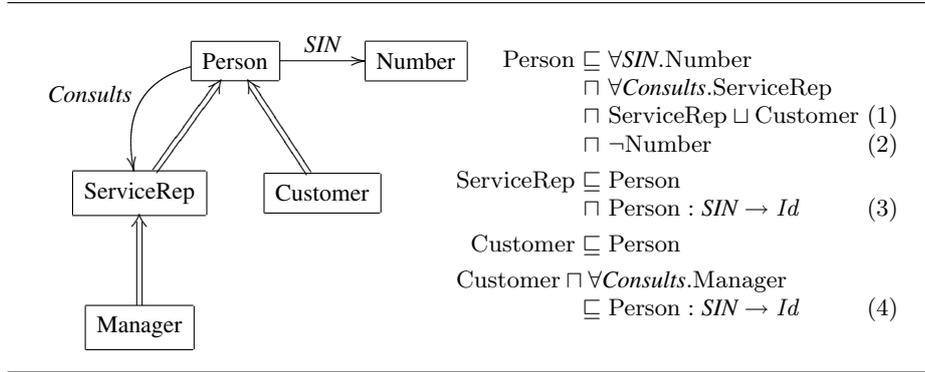
The remainder of the paper is organized as follows. The definition of  $\mathcal{DLFDE}$ , a Boolean complete DL based on attributes that includes the PFD concept constructor, immediately follows. In Section 3, we establish the first of our main results: the equivalence of the ABox consistency problem for  $\mathcal{DLFD}$  and the logical implication problem for  $\mathcal{DLFDE}$ , and that this equivalence continues to hold in the absence of any occurrences of PFDs. Section 4 introduces the logic  $\mathcal{DLFDE}^-$  which refines  $\mathcal{DLFDE}$  by introducing the above-mentioned syntactic boundary condition for PFDs, and shows that both the ABox consistency problem and the logical implication problem for  $\mathcal{DLFDE}^-$  are decidable and complete for EXPTIME. A more in depth example of using  $\mathcal{DLFDE}^-$  for reasoning about a query property follows in Section 5. Our summary comments then follow in Section 6.

## 2 Definitions

To simplify the presentation, the description logic  $\mathcal{DLFDE}$  defined below is based on *attributes* (also called *features*) instead of the more common case of roles. With regard to expressiveness, note that  $\mathcal{ALCN}$  with a suitable PFD construct can simulate our dialect. Conversely,  $\mathcal{DLFD}$  can simulate  $\mathcal{ALCQI}$  [20].

**Definition 1 (Description Logic  $\mathcal{DLFDE}$ )** *Let  $F$ ,  $A$  and  $N$  be sets of (names of) attributes, primitive concepts and individuals, respectively. A path expression is defined by the grammar “ $\text{Pf} ::= f. \text{Pf} \mid \text{Id}$ ” for  $f \in F$ . We define derived concept descriptions by the grammar on the left-hand-side of Figure 1. A concept description obtained by using the fourth production of this grammar is called an attribute value restriction. A concept description obtained by using the sixth production is called a path functional dependency (PFD).*

*An inclusion dependency  $\mathcal{C}$  is an expression of the form  $C \sqsubseteq D$ . A posed question  $\mathcal{Q}$  is an expression of the form  $E_1 \sqsubseteq E_2$ . A terminology (TBox)  $\mathcal{T}$  consists of a finite set of inclusion dependencies.*



**Fig. 2.** THE SERVICE SCHEMA AND TERMINOLOGY.

An ABox  $\mathcal{A}$  consists of a finite set of assertions of the form  $C(a)$  or  $f(a) = b$ , for  $C$  a concept description,  $f \in \mathbf{F}$  and  $\{a, b\} \subseteq \mathbf{N}$ .

The semantics of expressions is defined with respect to a structure  $(\Delta, \cdot^{\mathcal{I}})$ , where  $\Delta$  is a domain of “objects” and  $(\cdot)^{\mathcal{I}}$  an interpretation function that fixes the interpretation of primitive concepts  $A$  to be subsets of  $\Delta$ , primitive attributes  $f$  to be total functions  $(f)^{\mathcal{I}} : \Delta \rightarrow \Delta$  and individuals  $a$  to be elements of  $\Delta$ . The interpretation is extended to path expressions,  $(Id)^{\mathcal{I}} = \lambda x.x$ ,  $(f.Pf)^{\mathcal{I}} = (Pf)^{\mathcal{I}} \circ (f)^{\mathcal{I}}$  and derived concept descriptions  $C$ ,  $D$  and  $E$  as defined on the right-hand-side of Figure 1.

An interpretation satisfies an inclusion dependency  $C \sqsubseteq D$  (resp. a posed question  $E_1 \sqsubseteq E_2$ ) if  $(C)^{\mathcal{I}} \subseteq (D)^{\mathcal{I}}$  (resp.  $(E_1)^{\mathcal{I}} \subseteq (E_2)^{\mathcal{I}}$ ). An interpretation satisfies an ABox assertion  $C(a)$  (resp.  $f(a) = b$ ) if  $(a)^{\mathcal{I}} \in (C)^{\mathcal{I}}$  (resp.  $(f)^{\mathcal{I}}((a)^{\mathcal{I}}) = (b)^{\mathcal{I}}$ ).

The logical implication problem asks if  $\mathcal{T} \models \mathcal{Q}$  holds; that is, for a posed question  $\mathcal{Q}$ , if  $\mathcal{Q}$  is satisfied by any interpretation that satisfies all inclusion dependencies in  $\mathcal{T}$ . The ABox consistency problem asks if  $\mathcal{T} \cup \mathcal{A}$  is consistent; that is, if there exists an interpretation that satisfies all inclusion dependencies in  $\mathcal{T}$  and all assertions in  $\mathcal{A}$ .

To improve readability in the remainder of the paper, we follow the simple protocol of removing “. *Id*” from the end of path expressions that consist of at least one attribute. We also allow the use of standard abbreviations, e.g.,  $\perp$  for  $A \sqcap \neg A$ ,  $(Pf_1 \neq Pf_2)$  for  $\neg(Pf_1 = Pf_2)$ , etc. Finally, we write  $Pf(a) = b$  as shorthand for the equivalent set of primitive ABox assertions with “single use” intermediate individuals.

**Example 2** Figure 2 illustrates an information schema for a hypothetical online customer SERVICE system where, e.g., service representatives are special cases of people who in turn have social insurance numbers and consult with service representatives, and so on. The information can be captured as a SERVICE TBox with the inclusion dependencies in the right part of Figure 2. The four marked dependencies more thoroughly exercise the capabilities of  $\mathcal{DLFDE}$ , asserting that:

1. A person is either a service representative or a customer;
2. Nothing is both a person and a number;
3. Any service representative is uniquely identified by a social insurance number; and
4. (from our introductory comments) Any customer who has consulted with a manager is also uniquely identified by a social insurance number.

### 3 Equations and ABoxes

We first explore the relationship between ABox consistency problems and allowing path agreements in posed questions. It turns out that either capacity alone is sufficient: each is able to effectively simulate the other.

Intuitively, path equations (and inequations) can *enforce* that an arbitrary finite graph (with feature-labeled edges and concept description-labeled nodes) is a part of any model that satisfies the equations. Such a graph can equivalently be enforced by an ABox. Hence we have:

**Theorem 3** *Let  $\mathcal{T}$  be a  $\mathcal{DLFD}$  terminology and  $\mathcal{A}$  an ABox. Then there is a concept  $E$  such that  $\mathcal{T} \cup \mathcal{A}$  is not consistent if and only if  $\mathcal{T} \models E \sqsubseteq \perp$ .*

Conversely, it is also possible to show that ABox reasoning can be used for reasoning about equational constraints in the posed questions. However, as the equational concepts are closed under boolean constructors, a single equational problem may need to map to several ABox consistency problems.

**Theorem 4** *Let  $\mathcal{T}$  be a  $\mathcal{DLFD}$  terminology and  $E$  an equational concept. Then there is a finite set of ABoxes  $\{\mathcal{A}_i : 0 < i \leq k\}$  such that*

$$\mathcal{T} \models E \sqsubseteq \perp \text{ iff } \mathcal{T} \cup \mathcal{A}_i \text{ is not consistent for all } 0 < i \leq k.$$

Theorems 3 and 4 hold even when the terminology  $\mathcal{T}$  is restricted to the  $\mathcal{DLF}$  fragment (i.e., does not contain any occurrences of the PFD concept constructor).

### 4 Adding PFDs

The correspondence between ABox reasoning and equational concepts in the posed questions provides us with the necessary means to understanding the impact of PFDs in the presence of an ABox or path agreements. Indeed, our  $\mathcal{DLFDE}$  grammar in Figure 1 does not explicitly allow posed questions to contain PFDs (the PFD constructor is confined by the Grammar to the TBox). However, this restriction can be easily circumvented using the following lemma:

**Lemma 5** *Let  $\mathcal{T}$  be a  $\mathcal{DLFD}$  terminology and  $E_1 \sqsubseteq E_2 : \text{Pf}_1, \dots, \text{Pf}_k \rightarrow \text{Pf}$  a posed question. Then there is a equational concept  $E$  such that  $\mathcal{T} \models E \sqsubseteq \perp$  iff*

$$\mathcal{T} \models E_1 \sqsubseteq E_2 : \text{Pf}_1, \dots, \text{Pf}_k \rightarrow \text{Pf}.$$

#### 4.1 Undecidability

While allowing general reasoning about path agreements in terminologies leads immediately to undecidability (by virtue of a straightforward reduction of the uniform word problem [17]), the following two restricted cases have decidable decision problems:

- Allowing arbitrary PFDs in terminologies; and
- Allowing path agreements in the posed question.

Unfortunately, the combination of the two cases again leads to undecidability:

**Theorem 6** *Let  $\mathcal{T}$  be a  $\mathcal{DLFD}$  terminology and  $E$  an equational concept. Then the problem  $\mathcal{T} \models E \sqsubseteq \perp$  is undecidable.*

## 4.2 Decidability and a Boundary Condition

To regain decidability, we restrict the PFD constructor to the following two forms:

- $C : Pf_1, \dots, Pf_i, \dots, Pf_k \rightarrow Pf$ ; and
- $C : Pf_1, \dots, Pf_i, \dots, Pf_k \rightarrow Pf.f$ , for some primitive feature  $f$ .

We call the resulting fragment  $\mathcal{DLFDE}^-$ . To abstract syntax a bit for the sake of readability, the condition distinguishes, e.g., the PFDs  $f \rightarrow Id$  and  $f \rightarrow g$  from the PFD  $f \rightarrow g.f$ . Intuitively, a simple saturation procedure that “fires” PFDs on a hypothetical database is now guaranteed to terminate as a consequence.

Notice that the boundary condition still admits PFDs that express arbitrary keys or functional dependencies in the sense of the relational model, including those occurring in all our examples. Thus, we believe that restricting PFDs in this manner does not sacrifice any real world modeling utility.

**Theorem 7** *Let  $\mathcal{T}$  and  $\mathcal{T}'$  be a  $\mathcal{DLF}$  and  $\mathcal{DLFDE}^-$  terminologies, respectively, and  $E$  an equational concept. Then there is a concept  $E'$  such that*

$$\mathcal{T} \cup \mathcal{T}' \models E \sqsubseteq \perp \text{ iff } \mathcal{T} \models (E \sqcap E') \sqsubseteq \perp.$$

Moreover,  $E'$  can be constructed from  $\mathcal{T}'$  and  $E$  effectively and is polynomial in  $|\mathcal{T}'|$ .

The *boundary* condition on PFDs is essential for the above theorem to hold. If unrestricted PFDs are combined with either equations or an ABox, there is no limit on the *length* of paths participating in path agreements when measured from an initial object  $o \in E \sqcap E'$  in the associated satisfiability problem. Moreover, any minimal relaxation of this condition, i.e., allowing any PFDs of the form  $C : f \rightarrow g.h$ , already leads to undecidability [22].

**Corollary 8**  *$\mathcal{DLFDE}^-$  logical implication and ABox consistency problems are decidable and complete for EXPTIME.*

## 5 Applications to Query Optimization

Assuming set semantics for query results and the presence of a database schema, [5] and [10] have shown how conjunctive and positive query containment can be reduced to ABox consistency problems for the description logics  $\mathcal{DLR}$  [7] and  $\mathcal{SHIQ}$  [11], respectively. Analogous reductions can also be made to ABox consistency problems for  $\mathcal{DLFD}$ . In addition, more direct and transparent reductions that use path agreements are now possible with  $\mathcal{DLFDE}^-$  (c.f. the concept description  $E$  in the following example).

$\mathcal{DLFDE}^-$  can also be used in query reformulation when allowing duplicate semantics. However, in this case, PFDs serve an essential role as we now illustrate.

**Example 9** Consider the following SQL-like query on our example SERVICE schema that *finds distinct social insurance numbers for all persons who have consulted with a manager*.

```

select distinct N
from   Person as P, Number as N,
where  exists ( select *
                from   Manager as M
                where  P.Consults = M )
and    P.SIN = N

```

Clearly, there is a considerable incentive on the grounds of query performance to reason about the possibility of avoiding expensive duplication elimination operations, i.e., if the above query can be reformulated *without* the `distinct` keyword. Indeed, this is possible iff

$$\text{SERVICE} \models E \sqsubseteq E : N \rightarrow P$$

where  $E$  is the concept description

$$\begin{aligned} &(\forall P.\text{Person}) \sqcap (\forall N.\text{Number}) \sqcap (\forall M.\text{Manager}) \\ &\qquad \sqcap (P.\text{Consults} = M) \sqcap (P.\text{SIN} = N) \end{aligned}$$

that encodes the above query. Note that this clear and direct formulation relies on Lemma 5.

The link between the above formulation and earlier ABox consistency approaches to query containment is explained by Theorems 4 and 3.

## 6 Conclusions

Earlier research has led to the development of a Boolean-complete description logic called  $\mathcal{DLFD}$  that incorporated a powerful concept constructor for expressing uniqueness constraints called PFDs. Unfortunately, recent negative results have shown that the ABox consistency problem for  $\mathcal{DLFD}$  is not decidable. In this paper, we have proposed a boundary condition for PFDs that re-obtains decidability and tight complexity bounds for ABox consistency and logical implication problems. We have also shown how ABox consistency checking relates to logical implication problems when path agreement is allowed in posed questions. This connection is essential to the design of the decision procedure for  $\mathcal{DLFD}$  in the presence of an ABox.

There are several directions for future research, in particular exploring alternative fragments of  $\mathcal{DLFDE}$  with decidable reasoning problems; finding further restrictions on  $\mathcal{DLFDE}$  that lead to polynomial time reasoning algorithms; and incorporating more general ordering dependencies that generalize equality based reasoning that underlies PFDs.

## References

1. Joachim Biskup and Torsten Polle. Decomposition of Database Classes under Path Functional Dependencies and Onto Constraints. In *Foundations of Information and Knowledge Systems*, pages 31–49, 2000.
2. Joachim Biskup and Torsten Polle. Adding inclusion dependencies to an object-oriented data model with uniqueness constraints. *Acta Informatica*, 39:391–449, 2003.
3. Alex Borgida and Peter F. Patel-Schneider. A Semantics and Complete Algorithm for Subsumption in the CLASSIC Description Logic. *J. of Artificial Intelligence Research*, 1:277–308, 1994.
4. Alexander Borgida and Grant E. Weddell. Adding Uniqueness Constraints to Description Logics (Preliminary Report). In *International Conference on Deductive and Object-Oriented Databases*, pages 85–102, 1997.
5. Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. On the Decidability of Query Containment under Constraints. In *ACM Symposium on Principles of Database Systems*, pages 149–158, 1998.

6. Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Identification Constraints and Functional Dependencies in Description Logics. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 155–160, 2001.
7. Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. Unifying Class-Based Representation Formalisms. *J. Artificial Intelligence Research*, 11:199–240, 1999.
8. Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosf, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Technical report, W3C, 2004.
9. Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From SHIQ and RDF to OWL: the making of a Web Ontology Language. *J. Web Sem.*, 1(1):7–26, 2003.
10. Ian Horrocks, Ulrike Sattler, Sergio Tessaris, and Stephan Tobies. How to Decide Query Containment under Constraints using a Description Logic. In *LPAR Int. Conf. on Logic for Programming and Automated Reasoning*, pages 326–343. LNCS 1955, 2000.
11. Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical Reasoning for Expressive Description Logics. In *Logic Programming and Automated Reasoning, LPAR'99*, pages 161–180, 1999.
12. Minoru Ito and Grant E. Weddell. Implication Problems for Functional Constraints on Databases Supporting Complex Objects. *Journal of Computer and System Sciences*, 49(3):726–768, 1994.
13. Vitaliy L. Khizder, David Toman, and Grant E. Weddell. Reasoning about Duplicate Elimination with Description Logic. In *Rules and Objects in Databases (DOOD, part of CL'00)*, pages 1017–1032, 2000.
14. Vitaliy L. Khizder, David Toman, and Grant E. Weddell. On Decidability and Complexity of Description Logics with Uniqueness Constraints. In *International Conference on Database Theory ICDT'01*, pages 54–67, 2001.
15. C. Lutz and M. Milicic. Description Logics with Concrete Domains and Functional Dependencies. In *Proc. European Conference on Artificial Intelligence (ECAI)*, pages 378–382, 2004.
16. Carsten Lutz, Carlos Areces, Ian Horrocks, and Ulrike Sattler. Keys, Nominals, and Concrete Domains. In *Proc. of the 18th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 349–354, 2003.
17. Michael Machtley and Paul Young. *An Introduction to the General Theory of Algorithms*. North-Holland Amsterdam, 1978.
18. U. Sattler, D. Calvanese, and R. Molitor. Relationships with other formalisms. In *The Description Logic Handbook: Theory, Implementation, and Applications*, chapter 4, pages 137–177. Cambridge University Press, 2003.
19. Lubomir Stanchev and Grant E. Weddell. Index Selection for Embedded Control Applications using Description Logics. In *Description Logics 2003*, pages 9–18. CEUR-WS vol.81, 2003.
20. David Toman and Grant Weddell. On Reasoning about Structural Equality in XML: A Description Logic Approach. *Theoretical Computer Science*, 336(1):181–203, 2005.
21. David Toman and Grant Weddell. On the Interaction between Inverse Features and Path-functional Dependencies in Description Logics. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 603–608, 2005.
22. David Toman and Grant Weddell. On Keys and Functional Dependencies as First-Class Citizens in Description Logics. In *Proc. of the 3rd Int. Joint Conf. on Automated Reasoning (IJCAR)*, pages 647–661, 2006.
23. David Toman and Grant E. Weddell. On Attributes, Roles, and Dependencies in Description Logics and the Ackermann Case of the Decision Problem. In *Description Logics 2001*, pages 76–85. CEUR-WS vol.49, 2001.
24. Grant Weddell. A Theory of Functional Dependencies for Object Oriented Data Models. In *International Conference on Deductive and Object-Oriented Databases*, pages 165–184, 1989.