$\mathcal{CFD}nc$: A PTIME Description Logic with Functional Constraints and Disjointness

David Toman and Grant Weddell

Cheriton School of Computer Science
University of Waterloo, Canada
{david,gweddell}@cs.uwaterloo.ca

Abstract. We consider $\mathcal{CFD}nc$, an alternative to the description logic $\mathcal{CFD}$ that retains the latter's ability to support PTIME reasoning in the presence of terminological cycles with universal restrictions over functional roles and also in the presence of functional constraints over functional role paths. In contrast, $\mathcal{CFD}nc$ replaces the ability to have conjunction on left-hand-sides of inclusion dependencies with the ability to have primitive negation on right-hand-sides. This makes it possible to say that primitive concepts must denote disjoint sets of individuals, a common requirement with many information sources.

1 Introduction

Scalability issues in reasoning over the semantic web have led the W3C to adopt two description logic (DL) fragments of OWL 2 that are designed to ensure PTIME complexity in the size of respective knowledge bases for a number of important reasoning problems. Called *profiles*, the DLs are $\mathcal{EL}$++ [2] and DL-Lite [1, 6, 7]. Medical ontologies were an important motivation for the former, whereas the latter was heavily influenced by a need to access information residing in data sources conforming to relational schema, particularly in cases where the schema has been derived via ER modelling.

Toman and Weddell proposed an alternative to DL-Lite called $\mathcal{CFD}$ that was designed to provide better support for data sources based on relational schema that include more extensive collections of dependencies such as primary and foreign keys [22]. The paper showed that the problem of deciding *concept subsumption* in $\mathcal{CFD}$ had PTIME complexity, and therefore might qualify as a useful additional option for an OWL 2 profile. However, there are two problems with $\mathcal{CFD}$ that make it less attractive in this role: (1) unlike DL-Lite, it is not possible to say that two primitive concepts must denote disjoint sets of individuals or entities, a common requirement with many information sources, and (2) computing the certain answers to conjunctive queries is PSPACE-complete, even for queries of the form $\exists x.\mathrm{A}(x)$, for A a primitive concept.

In this paper we introduce $\mathcal{CFD}nc$, an alternative to $\mathcal{CFD}$ that retains the latter's key abilities: supporting terminological cycles with universal restrictions over functional roles, and supporting a rich variety of functional constraints over functional role paths. In particular, $\mathcal{CFD}nc$ replaces the ability in $\mathcal{CFD}$ to have conjunction on left-hand-sides of inclusion dependencies with a new ability to have primitive negation on right-hand-sides (the same is also true for the original version of DL-Lite). This removes both problems with $\mathcal{CFD}$. In particular, we show that the following fundamental reasoning problems are in PTIME.

SYNTAX	SEMANTICS: "$(\cdot)^{\mathcal{I}}$"
$\mathrm{C} ::= \mathrm{A}$	$\mathrm{A}^{\mathcal{I}} \subseteq \triangle$
$\mid \neg\mathrm{A}$	$\triangle \setminus \mathrm{A}^{\mathcal{I}}$
$\mid \mathrm{C}_1 \sqcap \mathrm{C}_2$	$\mathrm{C}_1^{\mathcal{I}} \cap \mathrm{C}_2^{\mathcal{I}}$
$\mid \forall\, \mathsf{Pf}\,.\mathrm{C}$	$\{x : \mathsf{Pf}^{\mathcal{I}}(x) \in \mathrm{C}^{\mathcal{I}}\}$
$\mid \mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$	$\{x : \forall\, y \in \mathrm{A}^{\mathcal{I}}. \bigwedge_{i=1}^{k} \mathsf{Pf}_i^{\mathcal{I}}(x) = \mathsf{Pf}_i^{\mathcal{I}}(y) \Rightarrow \mathsf{Pf}^{\mathcal{I}}(x) = \mathsf{Pf}^{\mathcal{I}}(y)\}$

Fig. 1. $\mathcal{CFD}nc$ CONCEPTS.

1. *Knowledge base consistency*: determining if at least one model exists for a given knowledge base;
2. *Logical implication*: determining if a given inclusion dependency is logically entailed by the terminological component of a given knowledge base; and
3. *Instance checking*: determining if a given concept assertion is entailed by a given knowledge base.

We also show that the problem of computing certain answers for arbitrary conjunctive queries over a $\mathcal{CFD}nc$ knowledge base $\mathcal{K}$ is in PTIME in the size of $\mathcal{K}$ and is PSPACE-complete for combined complexity, that is, when the size of a query is included.

Reasoning in DL-Lite, $\mathcal{EL}$, and their variants often relies on the existence of polynomially-sized *canonical models* (or *canonical structures* that closely resemble such models) to address the above reasoning tasks [14, 16]. It is worth noting that $\mathcal{CFD}nc$ does not share this property: an equivalent of a canonical model for a $\mathcal{CFD}nc$ knowledge base is necessarily exponential in the size of the knowledge base.

We begin in the next section by introducing the syntax and semantics of $\mathcal{CFD}nc$ and talk about some of its key features and limitations. The problems above are the focus of Section 4 in which we appeal to an automata-based method for their resolution. This method is introduced in Section 3 where we consider the simpler problem of *concept satisfiability*. Computing certain answers for conjunctive queries is considered in Section 5, and a review of related work and summary comments then follow in Sections 6 and 7, respectively.

2 The Description Logic $\mathcal{CFD}nc$

A formal definition of $\mathcal{CFD}nc$ knowledge bases and the above reasoning problems now follows. Observe that the logic is based on *attributes* or *features* instead of the more common case of *roles* which can denote arbitrary binary relations. However, this is not really a issue. Indeed, $\mathcal{CFD}nc$ is ideal for expressing reification for predicates of arbitrary arity [19].

Definition 1 ($\mathcal{CFD}nc$ Knowledge Bases) Let F, PC and IN be disjoint sets of (names of) attributes, primitive concepts and individuals, respectively. A *path function* Pf is a word in F^* with the usual convention that the empty word is denoted by id and concatenation by ".". *Concept descriptions* are defined by the grammar on the left-hand-side of

Figure 1 in which occurrences of "A" denote primitive concepts. A concept produced by the "$\mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}$" production of this grammar is called a *path functional dependency* (PFD). In addition, any occurrence of a PFD must adhere to one of the following two forms:

$$\begin{array}{l} 1.\ \mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf} \,.\, \mathsf{Pf}_i, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf} \ \text{ or} \\ 2.\ \mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf} \,.\, \mathsf{Pf}_i, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf} \,.f \end{array} \tag{1}$$

Metadata and data in a $\mathcal{CFD}nc$ knowledge base $\mathcal{K}$ are respectively defined by a *TBox* $\mathcal{T}_\mathcal{K}$ consisting of a finite set of *inclusion dependencies* of the form $\mathrm{A} \sqsubseteq \mathrm{C}$, and by an *ABox* $\mathcal{A}_\mathcal{K}$ consisting of a finite set of *concept assertions* of the form $\mathrm{A}(a)$ and *path function assertions* of the form $\mathsf{Pf}_1(a) = \mathsf{Pf}_2(b)$, where A is a primitive concept, C an arbitrary concept, $\{\mathsf{Pf}_1, \mathsf{Pf}_2\} \subseteq \mathsf{F}^*$ and where $\{a, b\} \subseteq \mathsf{IN}$.

Semantics is defined in the standard way with respect to a structure $(\triangle, (\cdot)^\mathcal{I})$, where $\triangle$ is a domain of "objects" and $(\cdot)^\mathcal{I}$ an interpretation function that fixes the interpretation of primitive concepts A to be subsets of $\triangle$, attributes f to be total functions on $\triangle$, and individuals a to be elements of $\triangle$. The interpretation is extended to path expressions by interpreting id, the empty word, as the identity function $\lambda x.x$, concatenation as function composition, and to derived concept descriptions C as defined on the right-hand-side for the remaining concept constructors.

An interpretation satisfies an inclusion dependency $\mathrm{A} \sqsubseteq \mathrm{C}$ if $\mathrm{A}^\mathcal{I} \subseteq \mathrm{C}^\mathcal{I}$, a concept assertion $\mathrm{A}(a)$ if $a^\mathcal{I} \in \mathrm{A}^\mathcal{I}$ and a path function assertion $\mathsf{Pf}_1(a) = \mathsf{Pf}_2(b)$ if $\mathsf{Pf}_1^\mathcal{I}(a^\mathcal{I}) = \mathsf{Pf}_2^\mathcal{I}(b^\mathcal{I})$. An interpretation satisfies a knowledge base $\mathcal{K}$ if it satisfies each inclusion dependency and assertion in $\mathcal{K}$. [1]

There are several reasoning problems for $\mathcal{CFD}nc$ that shall concern us. *Logical implication* asks if $\mathcal{T} \models \mathrm{A} \sqsubseteq \mathrm{C}$ holds; that is, if $\mathrm{A} \sqsubseteq \mathrm{C}$ is satisfied by any interpretation satisfying $\mathcal{T}$. *Knowledge base consistency* asks if there exists at least one interpretation for a give knowledge base $\mathcal{K}$, and *instance checking* asks if $\mathcal{K} \models \mathrm{A}(a)$ holds; that is, if $\mathrm{A}(a)$ is satisfied by any interpretation that satisfies $\mathcal{K}$. □

(*aside on notation*) We write $\mathsf{F}_\mathcal{K}$ and $\mathsf{PC}_\mathcal{K}$ to denote all attributes and primitive concepts occurring in $\mathcal{K}$, respectively, and write: (1) $\bot$ as shorthand for $\mathrm{A} \sqcap \neg\mathrm{A}$, (2) $a = b$ as shorthand for $id(a) = id(b)$, and (3) $f(a) = b$ and shorthand for $f(a) = id(b)$. Also we elide any mention of subscripts in our notation when their presence is clear from context. (*end of aside*)

The conditions imposed on PFDs in (1) distinguish, for example, PFDs of the form $\mathrm{C} : f \rightarrow id$ and $\mathrm{C} : f \rightarrow g$ from PFDs of the form $\mathrm{C} : f \rightarrow g.h$. This is necessary to ensure PTIME complexity for reasoning problems in $\mathcal{CFD}nc$ [13] and does not impact the modelling utility of $\mathcal{CFD}nc$ for formatted legacy data sources. It remains possible, for example, to capture arbitrary keys or functional dependencies in a relational schema.

Observe that only atomic concepts can appear on the left-hand-side or as part of a PFD. Indeed, relaxing this assumption in some cases will lead to a loss of PTIME

[1] Note that $\mathcal{CFD}nc$ does not assume the unique name assumption, but that its ability to express disjointness enables mutual inequality between each pair of n individuals to be captured by introducing $O(n)$ new atomic concepts, concepts assertions and inclusion dependencies in a straightforward way.

complexity for at least one of the reasoning problems for $\mathcal{CFD}nc$, and remains an open issue for others. (See related work and our conclusions below.)

3 TBox and Concept Satisfiability

It is easy to see that every $\mathcal{CFD}nc$ TBox $\mathcal{T}$ is consistent (by setting all primitive concepts to be interpreted as the empty set). However, for other reasoning tasks such as concept satisfiability and knowledge base consistency, it is convenient to assume by default, and without loss of generality, that $\mathcal{CFD}nc$ knowledge bases are given in a normal form.

Lemma 2 (TBox and ABox Normal Forms) For every $\mathcal{CFD}nc$ TBox $\mathcal{T}$, there exists an equivalent TBox $\mathcal{T}'$ that adheres to the following (more limited) grammar for $\mathcal{CFD}nc$ concept descriptions.

$$\mathrm{C} \ ::= \ \mathrm{A} \mid \neg\mathrm{A} \mid \forall f.\mathrm{A} \mid \mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}$$

Also, for every ABox $\mathcal{A}$, there exists an equivalent ABox $\mathcal{A}'$ containing only assertions of the form $f(a) = b$ and $a = b$. □

Obtaining $\mathcal{T}'$ and $\mathcal{A}'$ from an arbitrary knowledge base $\mathcal{K}$ is achieved by a straightforward introduction of auxiliary names for intermediate concept descriptions and individuals (e.g., see defn. of *simple concepts* in [21]).

Definition 3 (A Transition Relation for $\mathcal{T}$) Let $\mathcal{T}$ be a $\mathcal{CFD}nc$ TBox in normal form. We define a transition relation $\delta(\mathcal{T})$ over the set of states $S = \mathsf{PC} \cup \{\neg\mathrm{A} \mid \mathrm{A} \in \mathsf{PC}\}$ and the alphabet F as follows:

$$\mathrm{A}_1 \xrightarrow{\epsilon} \mathrm{A}_2 \in \delta(\mathcal{T}) \text{ if } \mathrm{A}_1 \sqsubseteq \mathrm{A}_2 \in \mathcal{T}$$
$$\mathrm{A}_1 \xrightarrow{\epsilon} \neg\mathrm{A}_2 \in \delta(\mathcal{T}) \text{ if } \mathrm{A}_1 \sqsubseteq \neg\mathrm{A}_2 \in \mathcal{T}$$
$$\mathrm{A}_1 \xrightarrow{f} \mathrm{A}_2 \in \delta(\mathcal{T}) \text{ if } \mathrm{A}_1 \sqsubseteq \forall f.\mathrm{A}_2 \in \mathcal{T}$$

where ϵ is the empty letter transition. □

The transition relation will allow us to construct *non-deterministic finite automata* (NFA) that can be used for various reasoning problems that relate to a $\mathcal{CFD}nc$ TBox $\mathcal{T}$. Note that we also follow common practice in automata theory and use ϵ for the empty letter in transition relations.[2]

Lemma 4 Let $M = (S, \{\mathrm{A}\}, \{\mathrm{B}\}, \delta(\mathcal{T}))$ be an NFA with the set of states S, start state A, final state B, and transition relation $\delta(\mathcal{T})$. Then $\mathcal{T} \models \mathrm{A} \sqsubseteq \forall\, \mathsf{Pf}\, .\mathrm{B}$ whenever $\mathsf{Pf} \in \mathcal{L}(M)$.

Proof (sketch) For $\mathsf{Pf} \in \mathcal{L}(M)$ there must be a run

$$\mathrm{A} = \mathrm{A}_0 \xrightarrow{l_1} \mathrm{A}_1 \xrightarrow{l_2} \mathrm{A}_2 \cdots \mathrm{A}_{k-1} \xrightarrow{l_k} \mathrm{A}_k = \mathrm{B}$$

[2] Another option would have been to use id for this purpose, but we thought, on balance, that this would hinder readability.

in M where $l_i \in \mathsf{F} \cup \{\epsilon\}$ and such that $\mathsf{Pf} = l_1.l_2.\cdots.l_k$. It follows from the definition of $\delta(\mathcal{T})$ that $\mathrm{A}_{i-1} \xrightarrow{l_i} \mathrm{A}_i$ exists if $\mathrm{A}_{i-1} \sqsubseteq \mathrm{A}_i$, for $l_i = \epsilon$, or $\mathrm{A}_{i-1} \sqsubseteq \forall l_i.\mathrm{A}_i$, for $l_i \in \mathsf{F}$ (and hence these dependencies are trivially implied by $\mathcal{T}$). The claim then follows by simple transitive reasoning, all necessary cases derive from the fact that

$$\{\mathrm{B}_1 \sqsubseteq \forall\, \mathsf{Pf}\,.\mathrm{B}_2, \mathrm{B}_2 \sqsubseteq \forall\, \mathsf{Pf}'\,.\mathrm{B}_3\} \models \mathrm{B}_1 \sqsubseteq \forall\, \mathsf{Pf}\,.\,\mathsf{Pf}'\,.\mathrm{B}_3,$$

and by induction on the length of the run. □

Note that the converse implication in this lemma may not hold, such as when A is inconsistent with respect to $\mathcal{T}$.

The problem of *concept satisfiability* asks, for a given concept C and TBox $\mathcal{T}$, if there exists an interpretation $\mathcal{I}$ for $\mathcal{T}$ in which $\mathrm{C}^{\mathcal{I}}$ is non-empty. Such problems can be reduced to the case where C is a primitive concept A by simply augmenting $\mathcal{T}$ with $\{\mathrm{A} \sqsubseteq \mathrm{C}\}$, where A is a fresh primitive concept.

Given a primitive concept A and TBox $\mathcal{T}$, one can test for primitive concept satisfiability by using the following NFA, denoted $\mathsf{nfa}^a_{\mathrm{B}}(\mathcal{T}, \{\mathrm{A}(a)\})$:

$$(S \cup \{a\}, \{a\}, \{\mathrm{B}\}, \delta(\mathcal{T}) \cup \{a \xrightarrow{\epsilon} A\}),$$

with states given by primitive concepts, their negations, and a distinguished node a, with start state a, with final state $\mathrm{B} \in S$, and with transition relation $\delta(\mathcal{T}) \cup \{a \xrightarrow{\epsilon} A\}$.

Theorem 5 (Concept Satisfiability) A is satisfiable with respect to the TBox $\mathcal{T}$ if and only if

$$\mathcal{L}(\mathsf{nfa}^a_{\mathrm{B}}(\mathcal{T}, \{\mathrm{A}(a)\})) \cap \mathcal{L}(\mathsf{nfa}^a_{\neg\mathrm{B}}(\mathcal{T}, \{\mathrm{A}(a)\})) = \emptyset$$

for every $\mathrm{B} \in \mathsf{PC}$.

Proof (sketch) For a primitive concept $\mathrm{B} \in \mathsf{PC}$, a word Pf in the intersection language of the two automata above is a witness of the fact that $\mathsf{Pf}^{\mathcal{I}}(a^{\mathcal{I}}) \in \mathrm{B}^{\mathcal{I}}$ and $\mathsf{Pf}^{\mathcal{I}}(a^{\mathcal{I}}) \in \neg\mathrm{B}^{\mathcal{I}}$ must hold in every model of $\mathcal{T}$, for reasons analogous to the proof of Lemma 4, which leads to a contradiction since Pf is a (total) function.

Conversely, if no such word exists then one can construct a *deterministic* finite automaton from $\mathsf{nfa}^a_{\mathrm{B}}(\mathcal{T}, \{\mathrm{A}(a)\})$, using the standard subset construction, in which no state containing both B and $\neg$B is reachable from the start state $\{a\}$. Unfolding the transition relation of this automaton, starting from the state $\{a\}$, labelling nodes by the concepts associated with the automaton's states, and adding missing features to complete trees in which no primitive concept is true for any node, yields a tree interpretation that satisfies $\mathcal{T}$ (in particular in which all PFD constraints are satisfied vacuously) and whose root a provides a witness for consistency of A. □

Since all the automata operations run in PTIME we immediately get the following result.

Corollary 6 Concept satisfiability with respect to $\mathcal{CFD}nc$ TBoxes is in PTIME.

Note it is not possible to *precompute* all inconsistent classes for an arbitrary C since that would require consideration of all possible *types* over PC (i.e., finite subsets of primitive concepts), a process essentially equivalent to constructing the deterministic automaton used in the proof of Theorem 5, and in turn make the procedure exponential.

4 ABox Reasoning

The automata-based approach to *concept satisfiability* can be extended to the more general problem of knowledge base consistency. Intuitively, each ABox individual a must be linked to the TBox automaton in a fashion similar to how the "prototypical object" a was linked in Section 3. This idea leads to the following definition:

Definition 7 (A Transition Relation for $\mathcal{A}$) Let $\mathcal{A}$ be a $\mathcal{CFD}nc$ ABox in normal form. We create a transition relation $\delta(\mathcal{A})$ for an nfa over the set of states $S = \mathsf{PC} \cup \{a \mid a \text{ in } \mathcal{A}\}$ and the alphabet F as follows:

$$\begin{aligned} a \xrightarrow{\epsilon} a \in \delta(\mathcal{A}) \text{ if } a \text{ appears in } \mathcal{A}, \\ a \xrightarrow{\epsilon} \mathrm{A} \in \delta(\mathcal{A}) \text{ if } \mathrm{A}(a) \in \mathcal{A}, \\ a \xrightarrow{f} b \in \delta(\mathcal{A}) \text{ if } f(a) = b \in \mathcal{A} \text{ and} \\ a \xrightarrow{\epsilon} b, b \xrightarrow{\epsilon} a \in \delta(\mathcal{A}) \text{ if } a = b \in \mathcal{A}. \end{aligned}$$

where ϵ is the empty letter transition. □

Observe that we have used ϵ transitions to simulate equality assertions in $\mathcal{A}$. This is justified, e.g., by considering the ABox individuals to be nominals.

(*aside on notation*) Hereon, we write "$n \overset{\mathsf{Pf}}{\rightsquigarrow} m$ in δ" if $\mathsf{Pf} \in \mathcal{L}(\mathsf{nfa}(S, \{m\}, \{n\}, \delta))$, where S will be some set of states (that will be clear from the context), where m and n will be two states in S, and where δ will denote a NFA transition relation over S (that will also be clear from context). (*end of aside*)

Unfortunately, taking $\delta(\mathcal{T}) \cup \delta(\mathcal{A})$ alone as the transition relation of an NFA and then testing for consistency of every ABox individual (as in Theorem 5) is not sufficient as the following cases illustrate. The problems raised by each case will be addressed by defining rules that impose conditions on a transition relation.

To begin, we need to ensure that ABox assertions $f(a) = b$ are functional:

Example 8 (Path Function Assertions) Consider the ABox $\mathcal{A} = \{f(a) = b, f(a) = c\}$. Clearly $b^{\mathcal{I}}$ must equal $c^{\mathcal{I}}$ in any model $\mathcal{I}$ of a knowledge base that includes $\mathcal{A}$. □

To remedy this, we define a *functionality rule* for the transition relation $\delta(\mathcal{T}, \mathcal{A})$ as follows:

if $a \overset{f}{\rightsquigarrow} b$ and $a \overset{f}{\rightsquigarrow} c$ in $\delta(\mathcal{T}, \mathcal{A})$ **then** $\{b \xrightarrow{\epsilon} c, c \xrightarrow{\epsilon} b\} \subseteq \delta(\mathcal{T}, \mathcal{A})$.

Next, we need to ensure that ABox assertions of the form $f(a) = b$ are coherent with TBox assertions $\mathrm{A} \sqsubseteq \forall f.\mathrm{B}$ with respect to concept memberships of a and b:

Example 9 (ABox and Value Restrictions) Consider the TBox $\mathcal{T} = \{\mathrm{A} \sqsubseteq \forall f.\mathrm{B}\}$ and an ABox $\mathcal{A} = \{f(a) = b, \mathrm{A}(a)\}$. Clearly, in any model $\mathcal{I}$ of the knowledge base $(\mathcal{T}, \mathcal{A})$, $b^{\mathcal{I}}$ must be an element of $\mathrm{B}^{\mathcal{I}}$. However, B *cannot* be reached from b in $\delta(\mathcal{T}) \cup \delta(\mathcal{A})$, and therefore an automaton based on this transition relation alone cannot reflect the correct concept membership of b. □

We define a *coherence rule* for the transition relation $\delta(\mathcal{T}, \mathcal{A})$ to remedy this as follows:

if $a \overset{f}{\rightsquigarrow} b$, $a \overset{\epsilon}{\rightsquigarrow} A$, and $A \overset{f}{\rightsquigarrow} B$ in $\delta(\mathcal{T}, \mathcal{A})$ **then** $b \overset{\epsilon}{\rightarrow} B \in \delta(\mathcal{T}, \mathcal{A})$.

And finally, consider that tree interpretations, such as the one we used to show concept consistency in Theorem 5, vacuously satisfy all PFDs in $\mathcal{T}$, but that this is not necessarily the case for a given ABox $\mathcal{A}$.

Example 10 (ABox and PFDs) Consider $\mathcal{A} = \{A(a), B(b), f(a) = c, f(b) = c\}$.

- A TBox $\mathcal{T} = \{\mathrm{A} \sqsubseteq \mathrm{B} : f \rightarrow id\}$ implies that the individuals a and b must denote the same domain element.
- A TBox $\mathcal{T} = \{\mathrm{A} \sqsubseteq \mathrm{B} : f \rightarrow g\}$ implies that there must be an additional (anonymous) individual d such that $g(a) = d$ and $g(b) = d$.

Note that the PFD $\mathrm{A} \sqsubseteq \mathrm{B} : f.g \rightarrow id$ is *also* violated by the pair of individuals a and b, this despite the fact that neither of these two individuals is the origin of an *explicit* $f.g$ path in $\mathcal{A}$: since features are interpreted as total functions, individual c must have an "outgoing" g feature, and therefore a and b must agree on $f.g$. □

A remedy for these cases is obtained by defining a *PFD closure rule* for the transition relation $\delta(\mathcal{T}, \mathcal{A})$ for each PFD $\mathrm{A} \sqsubseteq \mathrm{B} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf} \in \mathcal{T}$. The rule will refer to the following auxiliary functions.

$\mathsf{match}(a, b, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A}))$**:** Returns *true* if there is a (possibly empty) prefix Pf' of Pf such that $a \overset{\mathsf{Pf}'}{\rightsquigarrow} c$ and $b \overset{\mathsf{Pf}'}{\rightsquigarrow} c$ in $\delta(\mathcal{T}, \mathcal{A})$ for some individual c; it returns *false* otherwise.

$\mathsf{expf}(a, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A}))$**:** Returns the minimal set of transitions (by creating new individuals) such that $a \overset{\mathsf{Pf}}{\rightsquigarrow} c$ in $\delta(\mathcal{T}, \mathcal{A})$ holds for some c.

$\mathsf{mkeq}(a, b, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A}))$**:** Returns $\{c \overset{\epsilon}{\rightarrow} d, d \overset{\epsilon}{\rightarrow} c\}$ where, for some individuals c and d, we have $a \overset{\mathsf{Pf}}{\rightsquigarrow} c$ and $b \overset{\mathsf{Pf}}{\rightsquigarrow} d$ in $\delta(\mathcal{T}, \mathcal{A})$.

The PFD closure rule is then defined as follows:

if $\{a \overset{\epsilon}{\rightsquigarrow} A, b \overset{\epsilon}{\rightsquigarrow} B\} \subseteq \delta(\mathcal{T}, \mathcal{A})$ **and**
$\mathsf{match}(a, b, \mathsf{Pf}_i, \delta(\mathcal{T}, \mathcal{A}))$, for $0 < i \leq k$, **and** not $\mathsf{match}(a, b, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A}))$
then $\mathsf{expf}(a, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A})) \subseteq \delta(\mathcal{T}, \mathcal{A})$, $\mathsf{expf}(b, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A})) \subseteq \delta(\mathcal{T}, \mathcal{A})$, and
$\mathsf{mkeq}(a, b, \mathsf{Pf}, \delta(\mathcal{T}, \mathcal{A})) \subseteq \delta(\mathcal{T}, \mathcal{A})$

The rules enable one to define a transition relation for an NFA that captures reasoning in the knowledge base $(\mathcal{T}, \mathcal{A})$ as follows.

Definition 11 (Transition Relation $\delta(\mathcal{T}, \mathcal{A})$) Let $\delta(\mathcal{T}, \mathcal{A})$ be the smallest transition relation containing $\delta(\mathcal{T})$ and $\delta(\mathcal{A})$ that is closed under the functionality, coherence, and the PFD closure rules. □

Note that $\delta(\mathcal{T}, \mathcal{A})$ is constructed by applying the closure rules to $\delta(\mathcal{T}) \cup \delta(\mathcal{A})$. Since this process is monotonic, it is sound to check for the preconditions of the rules in the partially completed $\delta(\mathcal{T}) \cup \delta(\mathcal{A})$. We use $\delta(\mathcal{T}, \mathcal{A})$ as the transition function for the NFA $\mathsf{nfa}^a_{\mathrm{B}}(\mathcal{T}, \mathcal{A})$ with the start state $\{a\}$ and final state B (similarly to Section 3).

Theorem 12 (Knowledge Base Consistency) A knowledge base $(\mathcal{T}, \mathcal{A})$ is consistent if and only if

$$\mathcal{L}(\mathsf{nfa}^a_{\mathrm{B}}(\mathcal{T}, \mathcal{A})) \cap \mathcal{L}(\mathsf{nfa}^a_{\neg \mathrm{B}}(\mathcal{T}, \mathcal{A}))$$

is empty for all primitive concepts $B \in \mathsf{PC}$ and all ABox individuals a in $\mathcal{A}$.

Proof (sketch) Assume $\mathsf{Pf} \in \mathcal{L}(\mathsf{nfa}^a_{\mathrm{B}}(\mathcal{T}, \mathcal{A})) \cap \mathcal{L}(\mathsf{nfa}^a_{\neg \mathrm{B}}(\mathcal{T}, \mathcal{A}))$ for some path function Pf, individual a and primitive concept B, and that $\mathcal{I} \models (\mathcal{T}, \mathcal{A})$. Composing all the assertions corresponding to the transitions in $\delta(\mathcal{T}, \mathcal{A})$ along the runs corresponding to Pf in the two automata, however, implies that $\mathsf{Pf}^{\mathcal{I}}(a^{\mathcal{I}}) \in \mathrm{B}^{\mathcal{I}}$ and $\mathsf{Pf}^{\mathcal{I}}(a^{\mathcal{I}}) \in \neg \mathrm{B}^{\mathcal{I}}$ (similarly to Lemma 4); a contradiction as interpretations of path functions are functional.

For the other direction we define an interpretation $\mathcal{I}$ as follows: let $\lceil a \rceil$ be an representative of the equivalence class $\{a \mid a \overset{\epsilon}{\rightsquigarrow} b, b \overset{\epsilon}{\rightsquigarrow} a \text{ in } \delta(\mathcal{T}, \mathcal{A})\}$ and let $\mathrm{PF}(a)$ denote

$$\{f.\,\mathsf{Pf} \mid a \overset{f}{\rightsquigarrow} b \text{ not in } \delta(\mathcal{T}, \mathcal{A})\} \text{ for any individual } b\}.$$

Then set

- $\triangle^{\mathcal{I}} = \bigcup_{a \text{ in } \mathcal{A}} \{\lceil a \rceil.\, id\} \cup \{\lceil a \rceil.\, \mathsf{Pf} \mid \mathsf{Pf} \in \mathrm{PF}(a)\}$;
- $a^{\mathcal{I}} = \lceil a \rceil.\, id$;
- $\mathrm{A}^{\mathcal{I}} = \{\lceil a \rceil.\, \mathsf{Pf} \mid a \overset{\mathsf{Pf}}{\rightsquigarrow} \mathrm{A} \text{ in } \delta(\mathcal{T}, \mathcal{A})\}$; and
- $f^{\mathcal{I}} = \{(\lceil a \rceil.\, id, \lceil b \rceil.\, id) \mid a \overset{f}{\rightsquigarrow} b \text{ in } \delta(\mathcal{T}, \mathcal{A})\} \cup$
  $\{(\lceil a \rceil.\, \mathsf{Pf}, \lceil b \rceil.\, \mathsf{Pf}.f) \mid \lceil a \rceil.\, \mathsf{Pf}, \lceil a \rceil.\, \mathsf{Pf}.f \in \triangle^{\mathcal{I}}\}$.

It is immediate that $\mathcal{I} \models \mathcal{A}$ since $\delta(\mathcal{A}) \subseteq \delta(\mathcal{T}, \mathcal{A})$ and we corrected for all violations of PFDs. By inspecting inclusion dependencies in $\mathcal{T}$ it is also easy to see that $\mathcal{I} \models \mathcal{T}$. □

Note that the core of this construction is again the subset construction for NFA determinization (cf. Theorem 5) where the TBox-ABox interactions are facilitated by the closure rules. What remains is to show that knowledge base consistency can be checked in PTIME.

Lemma 13 $|\delta(\mathcal{T}, \mathcal{A})|$ is polynomial in $|\mathcal{T}| + |\mathcal{A}|$.

Proof (sketch) The number of individuals in $\delta(\mathcal{T}, \mathcal{A})$ is bounded by $|\mathcal{A}| + 2|\mathcal{T}||\mathcal{A}|^2$ since the PFD closure rule can add at most two new individuals per pair of individuals in $\mathcal{A}$ and PFD in $\mathcal{T}$. Thus, since the number of states is polynomial in $|\mathcal{T}| + |\mathcal{A}|$, the number of transitions in $\delta(\mathcal{T}, \mathcal{A})$ is also at most polynomial in $|\mathcal{T}| + |\mathcal{A}|$. □

Taken together with the argument we made for concept consistency with respect to a TBox yields PTIME algorithm for KB consistency. Since we do not assume the unique name assumption, the problem is also PTIME-hard (we have Horn-SAT embedded in reasoning with the PFDs alone).

Corollary 14 Knowledge base consistency for $\mathcal{CFD}nc$ is PTIME-complete. □

4.1 Logical Implication

Now we consider the questions of logical implication of the form $(\mathcal{T}, \mathcal{A}) \models \mathrm{C}(a)$, $(\mathcal{T}, \mathcal{A}) \models \mathsf{Pf}_1(a) = \mathsf{Pf}_2(b)$, and ultimately $\mathcal{T} \models \mathrm{A} \sqsubseteq \mathrm{C}$. Since C can be a complex concept and $\mathcal{CFD}nc$ is not closed under negation, logical implication must be resolved by asking several separate questions by exhaustively applying the following simplification rules:

$$\mathsf{Simp}(\mathrm{C}) \rightarrow \{\mathrm{C}\}$$
$$\mathsf{Simp}(\forall\, \mathsf{Pf}\, .\mathrm{C}_1 \sqcap \mathrm{C}_2) \rightarrow \mathsf{Simp}(\forall\, \mathsf{Pf}\, .\mathrm{C}_1) \cup \mathsf{Simp}(\forall\, \mathsf{Pf}\, .\mathrm{C}_2)$$
$$\mathsf{Simp}(\forall\, \mathsf{Pf}\, .\forall\, \mathsf{Pf}'\, .\mathrm{C}_1) \rightarrow \mathsf{Simp}(\forall\, \mathsf{Pf}\, .\, \mathsf{Pf}'\, .\mathrm{C}_1)$$

where C is one of the irreducible concepts of the forms $\forall\, \mathsf{Pf}\, .\mathrm{A}$, $\forall\, \mathsf{Pf}\, .\neg\mathrm{A}$, and $\forall\, \mathsf{Pf}\, .\mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}'$. We call the irreducible concepts obtained by these rules the *simplifications* of the given concept.

Lemma 15 $(\mathcal{T}, \mathcal{A}) \models \mathrm{C}(a)$ ($\mathcal{T} \models \mathrm{A} \sqsubseteq \mathrm{C}$) if and only if $(\mathcal{T}, \mathcal{A}) \models \mathrm{D}(a)$ ($\mathcal{T} \models \mathrm{A} \sqsubseteq \mathrm{D}$, respectively) for all $\mathrm{D} \in \mathsf{Simp}(\mathrm{C})$.

Proof (sketch) By observing that the each step of simplifications preserves logical implication. □

The simplified logical implication questions can now be reduced in a natural way to $\mathcal{CFD}nc$ knowledge base satisfiability as follows:

Theorem 16 (Instance Checking)

1. $(\mathcal{T}, \mathcal{A}) \models \forall\, \mathsf{Pf}\, .\mathrm{A}(a)$ iff $(\mathcal{T}, \mathcal{A} \cup \{\forall\, \mathsf{Pf}\, .\neg\mathrm{A}(a)\})$ is not satisfiable.
2. $(\mathcal{T}, \mathcal{A}) \models \forall\, \mathsf{Pf}\, .\neg\mathrm{A}(a)$ iff $(\mathcal{T}, \mathcal{A} \cup \{\forall\, \mathsf{Pf}\, .\mathrm{A}(a)\})$ is not satisfiable.
3. $(\mathcal{T}, \mathcal{A}) \models (\forall\, \mathsf{Pf}\, .\mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}')(a)$ iff
$$(\mathcal{T}, \mathcal{A} \cup \{\mathsf{Pf}(a) = b, \mathrm{A}(c), \mathrm{D}(\mathsf{Pf}'(b)), \neg\mathrm{D}(\mathsf{Pf}'(c)\} \cup \bigcup_{0<i\leq k} \{\mathsf{Pf}_i(b) = \mathsf{Pf}_i(c)\})$$
is not satisfiable, where b and c are fresh individual names and D is a fresh primitive concept.
4. $(\mathcal{T}, \mathcal{A}) \models (\mathsf{Pf}_1(a) = \mathsf{Pf}_2(b))$ iff $(\mathcal{T}, \mathcal{A} \cup \{\mathrm{D}(\mathsf{Pf}_1(a)), \neg\mathrm{D}(\mathsf{Pf}_2(b))\}$ is not satisfiable, where D a fresh primitive concept. □

For *logical implication* questions of the form $\mathcal{T} \models \mathrm{A} \sqsubseteq \mathrm{C}$, where C is irreducible, simply replace the ABox $\mathcal{A}$ in the above by $\{\mathrm{A}(a)\}$. The results then follow by virtue of the first three cases in the preceding theorem. Overall, we have the following:

Corollary 17 Both instance checking and logical implication for $\mathcal{CFD}nc$ are in PTIME.

5 Conjunctive Queries

We assume the standard definition of conjunctive queries, and begin by considering queries of the form

$$\exists x.(A_1(x) \wedge \ldots \wedge A_k(x)).$$

It turns out that such queries are the overriding source of complexity in computing certain answers over $\mathcal{CFD}nc$ knowledge bases.

Lemma 18 Let $(\mathcal{T}, \mathcal{A})$ be a consistent $\mathcal{CFD}nc$ knowledge base and q a conjunctive query of the form $\exists x.(A_1(x) \wedge \ldots \wedge A_k(x))$, where A_i are primitive concept names. The question $(\mathcal{T}, \mathcal{A}) \models q$ is PSPACE-hard (in combined complexity) and in PTIME in $|\mathcal{T}| + |\mathcal{A}|$.

Proof (sketch) It is sufficient to show that in every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$ there is a object $o \in \triangle^{\mathcal{I}}$ such that $o \in A_i^{\mathcal{I}}$ for all $0 < i \leq k$. We set

$$\mathcal{A}' = \mathcal{A} \cup \{f_i(s) = a_i \mid a_i \text{ an individual in } \mathcal{A}\}$$

where s and f_i do not appear in $\mathcal{T}$ and $\mathcal{A}$ and an NFA

$$M = \mathsf{nfa}^s_{A_1}(\mathcal{T}, \mathcal{A}') \times \ldots \times \mathsf{nfa}^s_{A_k}(\mathcal{T}, \mathcal{A}').$$

The remainder of the argument is similar to the concept consistency proof (Theorem 5), namely $\mathcal{L}(M) \neq \emptyset$ if and only if $(\mathcal{T}, \mathcal{A}) \models q$:

- If $(\mathcal{T}, \mathcal{A}) \models q$ then, in every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$, there must be an ABox individual a_i and a path function Pf such that $\mathsf{Pf}^{\mathcal{I}}(a_i^{\mathcal{I}}) \in \mathrm{A}_j^{\mathcal{I}}$ for all $0 < j \leq k$. Then, however, it is easy to verify that $f_i.\mathsf{Pf} \in \mathcal{L}(M)$ by analysis of the definition of M;
- Conversely, if $f_i.\mathsf{Pf} \in \mathcal{L}(M)$, then, in every model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$ and every $0 < j \leq k$, it follows that $\mathsf{Pf}^{\mathcal{I}}(a_i^{\mathcal{I}}) \in \mathrm{A}_j^{\mathcal{I}}$ since $f_i.\mathsf{Pf} \in \mathcal{L}(\mathsf{nfa}^s_{A_j}(\mathcal{T}, \mathcal{A}'))$, and, by the definition of M, that $\mathsf{Pf} \in \mathcal{L}(\mathsf{nfa}^{a_i}_{A_j}(\mathcal{T}, \mathcal{A}))$.

Note that every word in $\mathcal{L}(M)$ must start with one of the f_i features, thus ensuring that a common individual is used.

The complexity bounds then follow from well known results in automata theory (the DFA intersection problem, [15]); for the lower bound we simply use a knowledge base in which the query is not directly satisfied by any ABox individual. □

This result can be extended to all rooted and tree shaped conjunctive queries by appropriately modifying the final states in the individual automata in the above construction. For general conjunctive queries, it becomes necessary to analyze the query in order to search for ABox matches for non-tree components, matches close to the ABox for tree-shaped parts connected to non-tree components, and use the above technique for tree-shaped disconnected components. A full elaboration of this is straightforward but requires much more space than is available. This yields a PTIME query answering algorithm in the size of the knowledge base, $|\mathcal{T}| + |\mathcal{A}|$, and shows PSPACE-completeness for combined complexity.

6 Related Work

PFDs in $\mathcal{CFD}nc$ were first introduced and studied in the context of graph-oriented data models such as RDF and its refinements [11, 23]. Subsequently, an FD concept constructor was proposed and incorporated in Classic [5], an early DL with PTIME reasoning capabilities, without changing the complexity of its implication problem. We mentioned earlier that removing the conditions imposed on PFDs in (1) for $\mathcal{CFD}nc$ makes all of its reasoning problems EXPTIME-complete [13]. This remains unchanged in the absence of primitive negation or in the presence of additional concept constructors

common in very rich DLs such as (general) concept negation, roles, qualified number restrictions, and so on [17, 18]. Relating to applications, PFDs have been incorporated in graph-based data models to address problems in schema diagnosis and synthesis [3, 4] and in query optimization [10, 12].

We also mentioned earlier that relaxing the syntactic restrictions for left-hand sides of inclusion dependencies often causes the loss of PTIME complexity for some of the reasoning problems of $\mathcal{CFD}nc$. Here are three cases worth noting.

- Allowing conjunction "$\sqcap$" yields the logic $\mathcal{CFD}^{\sqcap}$ and therefore makes logical implication PSPACE-complete [22].
- Allowing conjunction and value restriction "$\forall$" makes logical implication EXPTIME-complete [13].

In [9], the authors consider a DL with functional dependencies and a general form of keys added as additional varieties of dependencies, called a *key box*. 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 features, with analogous results [20]. Subsequently, Calvanese et al. have shown how DL-Lite can be extended with a path-based variety of identification constraints analogous to PFDs without affecting the complexity of reasoning problems [8].

7 Summary

We have presented the DL logic $\mathcal{CFD}nc$, a variation on the logic $\mathcal{CFD}$ with the following notable properties.

- $\mathcal{CFD}nc$ retains what we believe are the most important features of $\mathcal{CFD}$: its ability to capture terminological cycles with universal restrictions over functional roles and its ability to capture a rich variety of functional constraints over functional role paths.
- In contrast to $\mathcal{CFD}$, the logic adds an ability to express disjointness of atomic concepts.
- Also in contract to $\mathcal{CFD}$, the logic supports important reasoning services in PTIME: determining knowledge base consistency, deciding logical implication and instance checking.

There are a number of open issues and directions for continued research. The consequences of allowing $\mathcal{CFD}nc$ concept constructors other that conjunction on the left-hand-side of inclusion dependencies is, to the best of our knowledge, open. In particular, this includes values restrictions "$\forall$", negated primitive concepts "$\neg$A" and PFDs.

One enhancement to $\mathcal{CFD}nc$ that we believe is straightforward, and that would considerably enhance its utility for modelling RDF data sources, would be to allow roles and role inclusion axioms of either the form "$f \sqsubseteq R$" or the form "$R_1 \sqsubseteq R_2$" to be included in $\mathcal{CFD}nc$ TBoxes, and then to allow roles to be mentioned in conjunctive queries. We conjecture that allowing $\mathcal{EL}$ role constructors on right-hand-sides of inclusion dependencies in $\mathcal{CFD}nc$ would also be possible without damage to its PTIME capabilities.

References

1. Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The DL-Lite family and relations. *J. Artif. Intell. Res. (JAIR)*, 36:1–69, 2009.
2. Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ Envelope. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 364–369, 2005.
3. 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.
4. Joachim Biskup and Torsten Polle. Adding inclusion dependencies to an object-oriented data model with uniqueness constraints. *Acta Informatica*, 39:391–449, 2003.
5. Alexander Borgida and Grant Weddell. Adding Uniqueness Constraints to Description Logics (Preliminary Report). In *International Conference on Deductive and Object-Oriented Databases*, pages 85–102, 1997.
6. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. *DL-Lite*: Tractable description logics for ontologies. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 602–607, 2005.
7. Diego Calvanese, Giuseppe de Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.
8. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Path-Based Identification Constraints in Description Logics. In *Proc. of the 11th Int. Joint Conf. on Principles of Knowledge Representation and Reasoning (KR)*, pages 231–241, 2008.
9. Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Identification Constraints and Functional Dependencies in Description Logics. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 155–160, 2001.
10. David DeHaan, David Toman, and Grant Weddell. Rewriting Aggregate Queries using Description Logics. In *Description Logics 2003*, pages 103–112. CEUR-WS vol.81, 2003.
11. Minoru Ito and Grant Weddell. Implication Problems for Functional Constraints on Databases Supporting Complex Objects. *Journal of Computer and System Sciences*, 49(3):726–768, 1994.
12. Vitaliy L. Khizder, David Toman, and Grant Weddell. Reasoning about Duplicate Elimination with Description Logic. In *Rules and Objects in Databases (DOOD, part of CL'00)*, pages 1017–1032, 2000.
13. Vitaliy L. Khizder, David Toman, and Grant Weddell. On Decidability and Complexity of Description Logics with Uniqueness Constraints. In *Int. Conf. on Database Theory ICDT'01*, pages 54–67, 2001.
14. Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to query answering in DL-Lite. In *KR*, 2010.
15. Dexter Kozen. Lower bounds for natural proof systems. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 254–266. IEEE Computer Society, 1977.
16. Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic EL using a relational database system. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2070–2075, 2009.
17. David Toman and Grant 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.
18. David Toman and Grant Weddell. Attribute Inversion in Description Logics with Path Functional Dependencies. In *Description Logics 2004*, pages 178–187. CEUR-WS vol.104, 2004.

19. David Toman and Grant Weddell. On Reasoning about Structural Equality in XML: A Description Logic Approach. *Theoretical Computer Science*, 336(1):181–203, 2005.
20. David Toman and Grant Weddell. On the Interaction between Inverse Features and Path-functional Dependencies in Description Logics. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 603–608, 2005.
21. David Toman and Grant Weddell. On Keys and Functional Dependencies as First-Class Citizens in Description Logics. In *Proc. of Int. Joint Conf. on Automated Reasoning (IJCAR)*, pages 647–661, 2006.
22. David Toman and Grant E. Weddell. Applications and extensions of ptime description logics with functional constraints. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 948–954, 2009.
23. 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.