# Query Answering over $\mathcal{CFD}_{nc}^{\forall}$ Knowledge Bases

David Toman and Grant Weddell

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

**Abstract.** We consider the problem of answering conjunctive queries in the description logic $\mathcal{CFD}_{nc}^{\forall}$, a generalization of the logic $\mathcal{CFD}_{nc}$ in which universal restrictions are now permitted on left-hand-sides of inclusion dependencies. We show this problem retains PTIME data complexity and exhibit a procedure in the spirit of OBDA in which a relational engine can be usefully employed to address scalability issues for an ABox. The procedure relies on a combination of the strategies that underlie both the perfect rewriting and combined approaches to OBDA. In particular, a knowledge base comprised of a TBox and ABox is first preprocessed to obtain a new ABox. An arbitrary conjunctive query together with the TBox can then be translated to a union of conjunctive queries that can be evaluated over the new ABox viewed as a relational database.

## 1 Introduction

*Ontology based data access* (OBDA) is concerned with access to data in a setting where the data sources may be incomplete with respect to a given logical schema or ontology, and where simple model checking no longer suffices to compute answers to queries. To address scalability issues relating to the volume of data, current approaches to OBDA focus on ontology and conjunctive query languages based on DL dialects for which computing certain answers is at worst PTIME-complete with respect to data complexity, and aim to achieve circumstances in which query answering can be efficiently reduced to SQL query evaluation over a relational encoding of data. In particular, scalable OBDA is possible for a DL dialect if, for any conjunctive query (CQ) $Q$, ontology $\mathcal{T}$, and an ABox $\mathcal{A}$ in that dialect, there is an ABox $\mathcal{A}'$, a *completion of* $\mathcal{A}$, and query $Q'$ that satisfy the following three requirements:

**(1)** The certain answers of $Q$ over the knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ can be obtained by evaluating $Q'$ viewed as a SQL query over $\mathcal{A}'$ viewed as a relational database;
**(2)** $\mathcal{A}'$ can be computed from $\mathcal{K}$ by a procedure with PTIME complexity with respect to $\mathcal{A}$ (i.e., with PTIME *data complexity* with respect to $\mathcal{K}$); and
**(3)** $Q'$ can be computed from $Q$ and $\mathcal{T}$ alone.

Indeed, scalable OBDA has been shown possible for various dialects of the DL-Lite and $\mathcal{EL}$ families of DLs [2, 6–9].

Recently, Toman and Weddell proposed $\mathcal{CFD}_{nc}$ [14], a dialect of their $\mathcal{CFD}$ family of DLs [4, 13] that has PTIME complexity for the fundamental reasoning tasks of determining knowledge base consistency and of computing the certain answers to conjunctive queries, and conjectured that scalable OBDA would also be possible for this logic. In this paper, we show that this is indeed true, and remains true for $\mathcal{CFD}_{nc}^{\forall}$, a slight generalization of $\mathcal{CFD}_{nc}$ in which value restrictions are also permitted on left-hand-sides of inclusion dependencies. In particular, our contributions, in the order presented, are as follows:

– We show that concept satisfiability with respect to a $\mathcal{CFD}_{nc}^{\forall}$ TBox $\mathcal{T}$ is complete for NLOGSPACE by appeal to an automata that derives from $\mathcal{T}$;
– We exhibit an ABox completion procedure with PTIME data complexity to compute $\mathcal{A}'$ above for a given knowledge base $\mathcal{K}$, and show how this can be coupled with our automata for checking concept satisfiability to yield a procedure for checking knowledge base consistency, also with PTIME data complexity (thus satisfying condition (**2**) above);
– We define a mapping that produces a union of conjunctive queries $Q'$ from a given conjunctive query $Q$ and $\mathcal{T}$, and show that evaluating $Q'$ as a SQL query over $\mathcal{A}'$, viewed as a relational database, computes the certain answers of $Q$ over $\mathcal{K}$ (thus satisfying conditions (**1**) and (**3**) above).

Altogether, this shows that scalable OBDA for $\mathcal{CFD}_{nc}^{\forall}$ is also possible, and in particular that computing the certain answers to conjunctive queries over a $\mathcal{CFD}_{nc}^{\forall}$ knowledge base can be achieved by a *combination* of perfect rewriting [2] and combined approaches [6–9]. In the process, we also show that this is *not* possible for either of the two approaches alone, and that the potential for an exponential blowup of query rewriting cannot be avoided. All this follows since the data complexity for conjunctive query answering in $\mathcal{CFD}_{nc}^{\forall}$ is PTIME-complete and since the combined complexity of this problem is PSPACE-complete.

We begin in the next section by introducing the syntax and semantics of $\mathcal{CFD}_{nc}^{\forall}$, including a normal form that is assumed in the remainder of the paper. The problem of concept satisfiability for $\mathcal{CFD}_{nc}^{\forall}$ TBoxes is then considered in Section 3. Our second contribution is given in Section 4 in which we present our ABox completion procedure, and show how consistency of $\mathcal{CFD}_{nc}^{\forall}$ knowledge bases can be determined in PTIME. Our final contribution relating to the above mentioned mapping of conjunctive queries is the topic of Section 5, and a discussion of related work and summary comments then follow in Sections 6 and 7.

## 2 The Description Logic $\mathcal{CFD}_{nc}^{\forall}$

$\mathcal{CFD}_{nc}^{\forall}$ is a member of the $\mathcal{CFD}$ family of DLs, all of which are essentially fragments of FO with underlying signatures based on disjoint sets of unary predicate symbols called *primitive concepts*, constant symbols called *individuals* and unary

| Syntax | Semantics: "$(\cdot)^{\mathcal{I}}$" |
|---|---|
| $\mathrm{C} ::= \mathrm{A}$ | $\mathrm{A}^{\mathcal{I}} \subseteq \triangle$ |
| $\mid\ \forall\, \mathsf{Pf}\,.\mathrm{C}$ | $\{x : \mathsf{Pf}^{\mathcal{I}}(x) \in \mathrm{C}^{\mathcal{I}}\}$ |
| $\mathrm{D} ::= \mathrm{A}$ | $\mathrm{A}^{\mathcal{I}} \subseteq \triangle$ |
| $\mid\ \neg\mathrm{A}$ | $\triangle \setminus \mathrm{A}^{\mathcal{I}}$ |
| $\mid\ \forall\, \mathsf{Pf}\,.\mathrm{D}$ | $\{x : \mathsf{Pf}^{\mathcal{I}}(x) \in \mathrm{D}^{\mathcal{I}}\}$ |
| $\mid\ \mathrm{C} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$ | $\{x : \forall\, y \in \mathrm{C}^{\mathcal{I}}.\ \bigwedge_{i=1}^{k} \mathsf{Pf}_i^{\mathcal{I}}(x) = \mathsf{Pf}_i^{\mathcal{I}}(y) \Rightarrow \mathsf{Pf}^{\mathcal{I}}(x) = \mathsf{Pf}^{\mathcal{I}}(y)\}$ |

**Fig. 1.** $\mathcal{CFD}_{nc}^{\forall}$ concepts.

function symbols called *attributes*. Note that incorporating attributes deviates from normal practice to use binary predicate symbols called *roles*. However, attributes make is easier to incorporate concept constructors suited to the capture of relational data sources and constraints such as keys and functional dependencies by a straightforward reification of $n$-ary predicates. Thus, e.g., a role $R$ in $\mathcal{ALC}$ would correspond to a primitive concept $R_C$ and two attributes *domR* and *ranR* in $\mathcal{CFD}_{nc}^{\forall}$, and an $\mathcal{ALC}$ inclusion dependency $\mathrm{A} \sqsubseteq \forall R.\mathrm{B}$ would be captured as the $\mathcal{CFD}_{nc}^{\forall}$ inclusion dependency $\forall domR.\mathrm{A} \sqsubseteq \forall ranR.\mathrm{B}$.

**Definition 1** **($\mathcal{CFD}_{nc}^{\forall}$ Knowledge Bases)** Let $\mathsf{F}$, $\mathsf{PC}$ and $\mathsf{IN}$ be disjoint sets of (names of) attributes, primitive concepts and individuals, respectively. A *path function* $\mathsf{Pf}$ is a word in $\mathsf{F}^*$ with the usual convention that the empty word is denoted by *id* and concatenation by ".". *Concepts* C and D are defined by the grammars on the left-hand-side of Figure 1 in which occurrences of "A" denote primitive concepts. A concept "$\mathrm{C} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$" produced by the last production of the grammar for D is called a *path functional dependency* (PFD). To avoid undecidability [12], any occurrence of a PFD must also satisfy a *regularity* condition by adhering to one of the following two forms:

$$\begin{array}{l} (a)\ \mathrm{C} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}\,.\,\mathsf{Pf}_i, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}\ \text{ or} \\ (b)\ \mathrm{C} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}\,.\,\mathsf{Pf}_i, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}\,.f \end{array} \qquad (1)$$

A PFD is a *key* if it adheres to the first of these forms.

Metadata and data in a $\mathcal{CFD}_{nc}^{\forall}$ knowledge base $\mathcal{K}$ are respectively defined by a *TBox* $\mathcal{T}$ and an *ABox* $\mathcal{A}$. Assume $\mathrm{A} \in \mathsf{PC}$, C and D are arbitrary concepts given by the grammars in Figure 1, $\{\mathsf{Pf}_1, \mathsf{Pf}_2\} \subseteq \mathsf{F}^*$ and that $\{a, b\} \subseteq \mathsf{IN}$. Then $\mathcal{T}$ consists of a finite set of *inclusion dependencies* of the form $\mathrm{C} \sqsubseteq \mathrm{D}$, and $\mathcal{A}$ consists of a finite set of facts in form of *concept assertions* $\mathrm{A}(a)$, *basic function assertions* $f(a) = b$ and *path function assertions* $\mathsf{Pf}_1(a) = \mathsf{Pf}_2(b)$. $\mathcal{A}$ is called a *primitive* ABox if it consists only of concept and basic function assertions.

Semantics is defined in the standard way with respect to an interpretation $\mathcal{I} = (\triangle, (\cdot)^{\mathcal{I}})$, 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 function 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 or D as defined in Figure 1. An interpretation $\mathcal{I}$ satisfies an inclusion dependency $\mathrm{C} \sqsubseteq \mathrm{D}$ if $\mathrm{C}^{\mathcal{I}} \subseteq \mathrm{D}^{\mathcal{I}}$, a concept assertion $\mathrm{A}(a)$ if $a^{\mathcal{I}} \in \mathrm{A}^{\mathcal{I}}$, a basic function assertion $f(a) = b$ if $f^{\mathcal{I}}(a^{\mathcal{I}}) = b^{\mathcal{I}}$ and a path function assertion $\mathsf{Pf}_1(a) = \mathsf{Pf}_2(b)$ if $\mathsf{Pf}_1^{\mathcal{I}}(a^{\mathcal{I}}) = \mathsf{Pf}_2^{\mathcal{I}}(b^{\mathcal{I}})$. $\mathcal{I}$ satisfies a knowledge base $\mathcal{K}$ if it satisfies each inclusion dependency and assertion in $\mathcal{K}$, and also satisfies UNA if, for any individuals $a$ and $b$ occurring in $\mathcal{K}$, $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. $\square$

As usual, allowing conjunction (resp. disjunction) on the right-hand (resp. left-hand) sides of inclusion dependencies is a simple syntactic sugar.

The conditions imposed on PFDs in (1) are necessary to retain PTIME complexity for the reasoning problems [5, 12] and does not impact the modeling utility of $\mathcal{CFD}_{nc}^{\forall}$ for formatted legacy data sources such as relational databases. It remains possible, for example, to capture arbitrary keys or functional dependencies in a relational schema.

For reasoning tasks, such as TBox and more general KB consistency, it is convenient to assume by default, and without loss of generality, that $\mathcal{CFD}_{nc}^{\forall}$ knowledge bases are given in a normal form.

**Lemma 2 (TBox and ABox Normal Forms)**
For every $\mathcal{CFD}_{nc}^{\forall}$ TBox $\mathcal{T}$, there exists a conservative extension $\mathcal{T}'$ that adheres to the following (more limited) grammar for $\mathcal{CFD}_{nc}^{\forall}$ concept descriptions.

$$\begin{array}{rcl} \mathrm{C} & ::= & \mathrm{A} \mid \forall f.\mathrm{A} \\ \mathrm{D} & ::= & \mathrm{A} \mid \neg\mathrm{A} \mid \forall f.\mathrm{A} \mid \mathrm{A} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf} \end{array}$$

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

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 [12]); the normalized TBox and ABox are linear in the size of the inputs.

## 3 TBox and Concept Satisfiability

It is easy to see that every $\mathcal{CFD}_{nc}^{\forall}$ TBox $\mathcal{T}$ is consistent (by setting all primitive concepts to be interpreted as the empty set).

**Definition 3 (Transition Relation for $\mathcal{T}$)** Let $\mathcal{T}$ be a $\mathcal{CFD}_{nc}^{\forall}$ TBox in normal form. We define a transition relation $\delta(\mathcal{T})$ over the set of states

$$S = \mathsf{PC} \cup \{\neg\mathrm{A} \mid \mathrm{A} \in \mathsf{PC}\} \cup \{\forall f.\mathrm{A} \mid \mathrm{A} \in \mathsf{PC}, f \in \mathsf{F}\}$$

and the alphabet $\mathsf{F}$ as follows:

$$\begin{array}{rl} \mathrm{C} \xrightarrow{\epsilon} \mathrm{D} \in \delta(\mathcal{T}) & \text{if } \ \mathrm{C} \sqsubseteq \mathrm{D} \in \mathcal{T} \\ \forall f.\mathrm{A} \xrightarrow{f} \mathrm{A} \in \delta(\mathcal{T}) & \end{array}$$

where $\epsilon$ is the empty letter transition, $f \in \mathsf{F}$, $A \in \mathsf{PC}$, and $C, D \in S$. $\square$

The transition relation allows us to construct *non-deterministic finite automata* (NFA) that can be used for various reasoning problems that relate to a $\mathcal{CFD}_{nc}^{\forall}$ TBox $\mathcal{T}$. Note that we follow common practice in automata theory and use $\epsilon$ for the empty letter in transition relations.[1]

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

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

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

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 $l_i \in \mathsf{F}$ (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 the lemma then follows by induction on the length of the run. $\square$

### 3.1 Concept Satisfiability

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. Note the C concept can be a *conjunction* of other concepts as it only appears on the right-hand side of an inclusion dependency.

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

$$(S, \{\mathrm{A}\}, \{\mathrm{B}\}, \delta(\mathcal{T})),$$

with states induced by primitive concepts, their negations, and value restrictions, with start state A, with the set of final states $\{\mathrm{B}\} \subseteq S$, and with transition relation $\delta(\mathcal{T})$.

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

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

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

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

Proof (sketch) Assume A is non-empty and hence there is $a \in \mathrm{A}^{\mathcal{I}}$. For a primitive concept $\mathrm{B} \in \mathsf{PC}$, a word $\mathsf{Pf}$ 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 $\mathsf{Pf}$ is a (total) function.

Conversely, if no such word exists then one can construct a *deterministic* finite automaton from $\mathsf{nfa}_{\mathrm{B}}^{\mathrm{A}}(\mathcal{T})$, using the standard subset construction, in which no state containing both B and ¬B is reachable from the start state A. Unfolding the transition relation of this automaton, starting from the state A, labeling 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 provides a witness for satisfiability of A (as we can simply assert $a \in \mathrm{A}$). □

To test for emptiness of $\mathsf{nfa}_{\mathrm{B}}^{\mathrm{A}}(\mathcal{T})$ we use an graph connectivity algorithm that (nondeterministically) searches for a $(\mathrm{A}, \mathrm{A}) - (\mathrm{B}, \neg\mathrm{B})$ path in the (virtual) poly-sized product automaton [3]; the following result is then immediate.

**Corollary 6** Concept satisfiability with respect to $\mathcal{CFD}_{nc}^{\forall}$ TBoxes is complete for NLOGSPACE.

Note that, as we remarked above, this procedure can be used to test for consistency of *conjunctions of concepts* in $\mathcal{CFD}_{nc}^{\forall}$. It is, however, impossible to *precompute* all such inconsistent concepts since this would require consideration of all possible *types* over $\mathsf{PC}$ (or finite subsets of primitive concepts), a process essentially equivalent to constructing an equivalent deterministic automaton which can require exponential time [3].

## 4 ABox Completion

To test for consistency we follow the path first outlined for the *combined approach* to CQ answering in PTIME-complete DLs [9] by *completing* the explicit data using the TBox. Note however, that in our case do not attempt to generate *auxiliary anonymous individuals* to satisfy totality of features (the counterpart of qualified existential restrictions in $\mathcal{EL}$). We reuse the completion later for CQ answering.

**Definition 7 (ABox Completion)** A *completion of an ABox $\mathcal{A}$ with respect to $\mathcal{T}$*, denoted $\mathsf{completion}_{\mathcal{T}}(\mathcal{A})$, is the least ABox that contains $\mathcal{A}$ and is closed under the rules in Figure 2.

**Lemma 8** Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a $\mathcal{CFD}_{nc}^{\forall}$ knowledge base such that $\mathrm{A}(a) \in \mathsf{completion}_{\mathcal{T}}(\mathcal{A})$. Then $\mathcal{K} \models \mathrm{A}(a)$.

Proof (sketch) The completion rules in Figure 2 only add facts implied by $\mathcal{K}$.

**ABox Equality Rules:**

$$\begin{array}{llll}
\text{if} & a = b, b = c \in \mathcal{A} & \text{then add} & a = c \ \text{ to } \mathcal{A} \\
\text{if} & f(a) = b, b = c \in \mathcal{A} & \text{then add} & f(a) = c \text{ to } \mathcal{A} \\
\text{if} & a = b, f(b) = c \in \mathcal{A} & \text{then add} & f(a) = c \text{ to } \mathcal{A} \\
\text{if} & f(a) = b, f(a) = c \in \mathcal{A} & \text{then add} & b = c \ \text{ to } \mathcal{A} \\
\text{if} & a = b, \mathrm{A}(a) \in \mathcal{A} & \text{then add} & \mathrm{A}(b) \ \text{ to } \mathcal{A}
\end{array}$$

**ABox–$\delta(\mathcal{T})$ Interactions:**

$$\begin{array}{llll}
\text{if } \mathrm{A}(a) \in \mathcal{A} & \text{and } \epsilon \in \mathcal{L}(\mathsf{nfa}_{\mathrm{B}}^{\mathrm{A}}(\delta(\mathcal{T}))) & \text{then add } \mathrm{B}(a) & \text{to } \mathcal{A} \\
\text{if } \mathrm{A}(a), f(a) = b \in \mathcal{A} & \text{and } f \in \mathcal{L}(\mathsf{nfa}_{\mathrm{B}}^{\mathrm{A}}(\delta(\mathcal{T}))) & \text{then add } \mathrm{B}(b) & \text{to } \mathcal{A}
\end{array}$$

**ABox–PFD Interactions:**

if $A(a), B(b) \in \mathcal{A}$, $\mathsf{Pf}'_i(a) = c_i, \mathsf{Pf}'_i(b) = c_i \in \mathcal{A}$ for $0 < i \le k$, and
$\mathrm{A} \sqsubseteq \mathrm{B} : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf} \in \mathcal{T}$ such that $\mathsf{Pf}'_i$ is a prefix of $\mathsf{Pf}_i$ then

1. $\mathsf{Pf}'(a) = c, \mathsf{Pf}'(b) = c \in \mathcal{A}$ for $\mathsf{Pf}'$ a prefix of $\mathsf{Pf}$ and $c$ an $\mathcal{A}$ individual,
2. $\mathsf{Pf}(a) = c, \mathsf{Pf}(b) = d \in \mathcal{A}$ and then we add $c = d$ to $\mathcal{A}$, or
3. $\mathsf{Pf}$ is of the form $\mathsf{Pf}''.f$ and $\mathsf{Pf}''(a) = c, \mathsf{Pf}''(b) = d$ and then
   we add $f(c) = e, f(d) = e$ to $\mathcal{A}$ for a new individual $e$.

**Fig. 2.** ABox Completion Rules.

Note that the converse is contingent on consistency of $\mathcal{K}$.

### 4.1 Knowledge Base Consistency

The automata-based approach to *concept satisfiability* can be used to the more general problem of knowledge base consistency.

**Theorem 9 ($\mathcal{K}$ Consistency)** Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a $\mathcal{CFD}_{nc}^{\forall}$ knowledge base. $\mathcal{K}$ is consistent if and only if $\{\mathrm{A} \mid \mathrm{A}(a) \in \mathsf{completion}_{\mathcal{T}}(\mathcal{A})\}$ is satisfiable with respect to $\mathcal{T}$ for every individual $a$ in $\mathsf{completion}_{\mathcal{T}}(\mathcal{A})$.

Proof (sketch) If $\{\mathrm{A} \mid \mathrm{A}(a) \in \mathsf{completion}_{\mathcal{T}}(\mathcal{A})\}$ is not satisfiable then the knowledge base $(\mathcal{T}, \{\mathrm{A} \mid \mathrm{A}(a) \in \mathsf{completion}_{\mathcal{T}}(\mathcal{A})\})$ is inconsistent and hence $\mathcal{K}$ is also inconsistent due to Lemma 8.

For the other direction we construct an interpretation for $\mathcal{K}$ as follows: We construct an interpretation $I$ by closing $\mathcal{A}$ under the rules in Figure 2 and then extending the closure with anonymous objects by unfolding $\delta(\mathcal{T})$ for every individual that does not satisfy the totality of features requirement. It is easy to show $I \models \mathcal{T}$. $\square$

Note that the ABox individuals are considered *separately* (i.e., without considering the ABox equalities); however, were an inconsistency with respect to an individual forced by traversing a feature within the ABox, this inconsistency will be detected when the target individual is considered in the above theorem.

The construction yields a PTIME algorithm for consistency checking, a lower bound has been already established for $\mathcal{CFD}_{nc}$ [14].

**Corollary 10** Knowledge base consistency for $\mathcal{CFD}_{nc}^{\forall}$ is PTIME-complete. $\Box$

The interpretation constructed in the *only if* part of the proof is called the *canonical model* of $\mathcal{K}$ and is analogous to the minimal models of horn theories.

## 5 Conjunctive Queries over $\mathcal{CFD}_{nc}^{\forall}$ KBs

In the following we show that CQ answering is tractable in data complexity for $\mathcal{CFD}_{nc}^{\forall}$. This development also subsumes the instance retrieval problem.

A *conjunctive query* (CQ) is an expression of the form $\{\bar{x} \mid \exists\bar{y}.\text{BODY}\}$ where BODY is a conjunction of atomic formulas of the form $\mathrm{C}(x)$ and $\mathsf{Pf}(x) = \mathsf{Pf}'(y)$ for C a $\mathcal{CFD}_{nc}^{\forall}$ concept description not containing PFDs, $\mathsf{Pf}, \mathsf{Pf}' \in \mathsf{F}^*$, and $x$ are variables among $\bar{x} \cup \bar{y}$. We often conflate the BODY of the query with the set of its atomic conjuncts. We call the variables $\bar{x}$ the *answer variables*. A *union of CQ* (UCQ) is a set of CQ that denotes a disjunction of the formulas that define the individual CQs. An *answer to a CQ* $\varphi$ w.r.t. a KB $\mathcal{K}$ is a vector of individuals $\bar{a} \subseteq \mathsf{IN}$ such that $\mathcal{K} \models \varphi(\bar{a})$ where $\varphi(\bar{a})$ is a formula obtained from $\varphi$ by substituting $\bar{x}$ by $\bar{a}$. We assume that CQs are connected; otherwise we simply process each component separately.

Without loss of generality we can assume that all CQs are in normal form: the concepts used in the CQs are primitive concepts or their negations and the equational atoms of the form $f(x) = y$. It is easy to see that every CQ can be transformed to a equivalent one by introducing additional variables and existential quantifiers.

To compute answers for a CQ $\varphi$ we use the notion of *CQ folding*. We need the following auxiliary definition:

**Definition 11** Let $\mathcal{T}$ be a $\mathcal{CFD}_{nc}^{\forall}$ TBox and C a primitive concept, negation of one, or value restriction. We say that a primitive concept A is a $\mathsf{Pf}$-precondition of C in $\mathcal{T}$ if $\mathsf{Pf} \in \mathcal{L}(\mathsf{nfa}_C^A(\delta(\mathcal{T})))$.

**Lemma 12** Let $\mathcal{T}$ be a $\mathcal{CFD}_{nc}^{\forall}$ TBox and $\mathrm{A}_1, \ldots, \mathrm{A}_k$ all $\mathsf{Pf}$-preconditions of C in $\mathcal{T}$. Then in every model $I$ of $\mathcal{T}$ we have $\mathrm{C}^{\mathcal{I}} \subseteq \bigcup_{0<i\leq k} \mathrm{A}_i^{\mathcal{I}}$.

Proof (sketch) Follows from immediately Lemma 4.

The above allows us to replace concepts of the form $\neg A$ and $\forall f.A$ by their preconditions w.r.t. $\mathcal{T}$ and this way keeping each of the queries in normal form.

**Definition 13** Let $\varphi$ be a CQ. We define a set $\mathsf{Fold}_{\mathcal{T}}(\varphi)$ with respect to $\mathcal{T}$ to be the least set of CQ that contains $\varphi$ and is closed under the following rules.

1. If $\{\bar{x} \mid \exists\bar{y}.\text{BODY}\} \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$, $\{\neg\mathrm{A}(x)\} \subseteq \text{BODY}$ then $\{\bar{x} \mid \exists\bar{y}.\text{BODY}\} - \{\neg\mathrm{A}(x)\} \cup \{\mathrm{B_i}(x)\} \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$ for all $\mathrm{B_i}$ an $\epsilon$-precondition of $\neg$A;

2. If $\{\bar{x} \mid \exists \bar{y}.\textsc{body}\} \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$, $\{f(x) = y, \mathrm{A}_1(y), \ldots, \mathrm{A}_\mathrm{k}(y)\} \subseteq \textsc{body}$ and $y$ does not appear elsewhere in $\textsc{body}$ nor in $\bar{x}$, then $\{\bar{x} \mid \exists \bar{y}.\textsc{body} - \{f(x) = y, \mathrm{A}_1(y), \ldots, \mathrm{A}_\mathrm{k}(y)\} \cup \{\mathrm{B}_1^{\mathrm{j}_1}(x), \ldots, \mathrm{B}_\mathrm{k}^{\mathrm{j}_\mathrm{k}}(x)\}\} \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$ for all possible combinations of $\mathrm{B}_\mathrm{i}^{\mathrm{j}_\mathrm{i}}$ an $f$-precondition of $\mathrm{A}_\mathrm{i}$ w.r.t. $\mathcal{T}$.
3. If $\{\bar{x} \mid \exists \bar{y}.\textsc{body}\} \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$ and $\{f(x) = y, f(x') = y\} \subseteq \textsc{body}$, then $\{\bar{x} \mid \exists \bar{y}.\textsc{body}\}[x/x'] \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$;

The intuition behind this definition is that, to find query answers, it is now sufficient to *match* the queries in $\mathsf{Fold}(\varphi)$ explicitly against the (extended) ABox (cf. Definition 7) and verify correct concept membership for these nodes as prescribed by the query since possible matches outside of this ABox are reduced to primitive membership checks against $\mathsf{Pf}$-preconditions.

**Lemma 14** Let $\varphi$ be a CQ with at least one answer variable. Then $\bar{a}$ is an answer to $\varphi$ over $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if and only if there is a mapping $\mu : \bar{x} \cup \bar{y} \to \mathsf{IN}$ is the set of ABox individuals in $\mathsf{completion}_{\mathcal{T}}(\mathcal{A})$, such that

1. $\mu(x)$ an individual in $\mathcal{A}$ for $x \in \bar{x}$ an answer variable;
2. $f(\mu(x)) = \mu(y) \in \mathsf{completion}_{\mathcal{T}}(\mathcal{A})$ for all $f(x) = y \in \textsc{body}$; and
3. $\mathrm{A}(\mu(x)) \in \mathsf{completion}_{\mathcal{T}}(\mathcal{A})$ for all $\mathrm{A}(x) \in \textsc{body}$,

for at least one $\{\bar{x} \mid \exists \bar{y}.\textsc{body}\} \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$.

Proof (sketch) Observing that the extended ABox is essentially a part of the minimal model of $\mathcal{K}$ (since $\mathcal{K}$ is Horn) and that every element of $\mathsf{Fold}(\varphi)$ implies $\varphi$, it is easy to see that whenever (1-3) are satisfied, there is a match of $\varphi$ in the minimal model and thus $\bar{a}$ is an answer. Conversely, if a match of $\varphi$ in a minimal model exists yielding $\bar{a}$ as an answer, then part of the match will be realized in the ABox (since at least one variable must be bound to an ABox individual) and the reminder of the match must be forest-like. Hence, one of the queries in $\mathsf{Fold}_{\mathcal{T}}(\varphi)$ matches in the ABox making the remaining conjuncts implied by $\mathcal{T}$ due to Lemma 12. □

For CQ without answer variables, we need an additional step that checks whether the query (when equivalent to a concept) matches in the tree part of the canonical model of $\mathcal{K}$. To achieve this, we determine every primitive type $\{\mathrm{A}_1, \ldots, \mathrm{A}_n\} \subseteq \mathsf{PC}$ (of a potential completed ABox individual) whether C must be realized in the canonical model in which an extended ABox individual belongs to such a primitive type. We use the following construction: Let $\mathcal{T}$ be a $\mathcal{CFD}_{nc}^{\forall}$ TBox, $\{\mathrm{A}_1, \ldots, \mathrm{A}_n\} \subseteq \mathsf{PC}$ a consistent primitive type w.r.t. $\mathcal{T}$, and be a query the form $\psi = \{\emptyset \mid \exists y.\mathrm{B}_1(y) \wedge \ldots \wedge \mathrm{B}_k(y)\}$[2]. We define an automaton

$$M(\psi) = \mathsf{nfa}_{\mathrm{B}_1}^{\mathrm{A}_0}(\delta) \times \ldots \times \mathsf{nfa}_{\mathrm{B}_k}^{\mathrm{A}_0}(\delta)$$

where $\delta = \delta(\mathcal{T}) \cup \{A_0 \xrightarrow{\epsilon} \mathrm{A}_1, \ldots, A_0 \xrightarrow{\epsilon} \mathrm{A}_n\}$ and $\mathrm{A}_0$ is a primitive concept not occurring in $\mathcal{T}$.

[2] Due to the definition of $\mathsf{Fold}_{\mathcal{T}}$ it is sufficient to consider only queries of this form as more complex queries are simplified by the folding process.

**Definition 15** Let $\mathcal{T}$ be a $\mathcal{CFD}_{nc}^{\forall}$ TBox, $\{\mathrm{A}_1,\ldots,\mathrm{A}_n\} \subseteq \mathsf{PC}$ a primitive type, and $\psi \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$ of the form $\{\emptyset \mid \exists y.\mathrm{B}_1(y),\ldots,\mathrm{B}_k(y)\}$. We say that the type $\{\mathrm{A}_1,\ldots,\mathrm{A}_n\}$ forces $\psi$ if $M(\psi)$ is nonempty.

Now whenever such a query $\psi$ appears in $\mathsf{Fold}_{\mathcal{T}}(\varphi)$ we add $\exists x.T(x)$ for all $T$ that force $\psi$ w.r.t. $\mathcal{T}$. These additional queries can be evaluated solely with respect to the completed ABox and guarantee that $\psi$ is realized outside of the ABox whenever a match is found.

**Theorem 1.** *Let $\varphi$ be a CQ with at least one answer variable. Then $\bar{a}$ is an answer to $\varphi$ over $\mathcal{K} = (\mathcal{T},\mathcal{A})$ if and only if $\mathsf{completion}_{\mathcal{T}}(\mathcal{A}) \models \psi(\bar{a})$ for at least one $\psi \in \mathsf{Fold}_{\mathcal{T}}(\varphi)$.*

Proof (sketch) Follows from Lemma 14 and the observation that closed queries that correspond to $\mathcal{CFD}_{nc}^{\forall}$ concepts are handled using the construction in Definition 15.

Analyzing the constructions it is easy to verify that overall data complexity of CQ answering over $\mathcal{CFD}_{nc}^{\forall}$ knowledge bases is in PTIME, as the ABox completion can be realized by a Datalog program (that depends only on $\mathcal{T}$) followed by evaluation of an UCQ (defined by $\mathsf{Fold}_{\mathcal{T}}(\varphi)$) which is in $\mathrm{AC}_0$. This matches the lower bound established for $\mathcal{CFD}_{nc}$ in [14] (and hence precludes using perfect rewriting alone for CQ answering). Note also that $|\mathsf{Fold}_{\mathcal{T}}(\varphi)|$ is worst-case exponential in $|\mathcal{T}| + |\varphi|$; this, however, is unavoidable as the combined complexity of CQ answering is hard for PSPACE even for $\mathcal{CFD}_{nc}$ while merely NP-complete for UCQ. Hence, unless NP=PSPACE, the blowup cannot be avoided. This observation also precludes the various *filtering* approaches used previously for the combined approach [6–9]. Hence the combination of the combined approach and perfect rewriting is necessary for CQ answering over $\mathcal{CFD}_{nc}^{\forall}$.

## 6 Related Work

An early version of the $\mathcal{CFD}$ dialect first appeared in [4]; the name was a contraction of "CLASSIC with FDs". The present form appears in [13], which also explored the consequences of adding additional concept constructors on the complexity of concept subsumption problems. Dialect $\mathcal{CFD}_{nc}$ was recently proposed as a modification of $\mathcal{CFD}$ in [14], mainly to gain PTIME data complexity for conjunctive query answering. However, scalable OBDA in the sense we have outlined was not possible with the query answering approach used in this work.

In [13], the authors outline an alternative approach to computing the certain answers to so-called *attribute connected* conjunctive queries for the logic $\mathcal{CFD}$. The approach reduces such problems to concept subsumption problems in which path agreements are used in posed question concepts to encode an ABox.

Scalable OBDA based on a perfect rewriting of conjunctive queries was shown for DL-Lite in [2]. Note that if perfect rewriting suffices to accomplish scalable OBDA for a DL dialect, requirement (**2**) given in our introductory comments can be strengthened to require that $\mathcal{A}' = \mathcal{A}$. This has considerable advantages in an information integration setting in which rebuilding a new A is not feasible. The combined approach to scalable OBDA was first shown for a member of the $\mathcal{EL}$ family in [9]. In general, if a combined approach suffices, requirement (**3**) can be strengthened (but not requirement (**2**)) to require that $Q'$ can be computed independently of the inclusion dependencies in a given ontology, and can have a size bounded by a polynomial in the size of $Q$, but see [8] to understand how this is possible in the presence of role hierarchies.

## 7 Summary

We have introduced a new member of the $\mathcal{CFD}$ family of DL dialects called $\mathcal{CFD}_{nc}^{\forall}$ with the following notable properties.

- $\mathcal{CFD}_{nc}^{\forall}$ is a generalization of $\mathcal{CFD}_{nc}$. Consequently, it inherits the ability of $\mathcal{CFD}_{nc}$ to capture terminological cycles with universal restrictions over functional roles, to capture a rich variety of functional constraints over functional role paths, and to express disjointness of atomic concepts. In addition, it is now possible in $\mathcal{CFD}_{nc}^{\forall}$ to have universal restrictions occurring on left-hand-sides of inclusion dependencies.
- We have established the foundations for scalable OBDA for $\mathcal{CFD}_{nc}^{\forall}$ knowledge bases, for which both knowledge base consistency and conjunctive query evaluation have PTIME data complexity.

We have also shown that it is *not* possible for scalable OBDA over a $\mathcal{CFD}_{nc}^{\forall}$ knowledge base $\mathcal{K}$ that is based exclusively on either a perfect rewriting approach or a combined approach. Thus, there is a potential for exponential blowup of a conjunctive query in the size of a TBox, and for a non-linear completion $\mathcal{A}'$ in the size of $\mathcal{K}$. However, note that earlier work on (1) reducing the complexity of generated queries in perfect rewriting approaches [11], e.g., removing members of a union of conjunctive queries that are subsumed by other members, and on (2) using interval encodings to compress the expansion of an ABox in combined approaches [1, 10] can also be applied in our setting.

The complexity landscape of most of the variants of $\mathcal{CFD}$ have now been resolved. In particular, see [15][3] in the case of $\mathcal{CFD}_{nc}$ and [13] for the case of $\mathcal{CFD}$ (which allows conjunction on left-hand-sides of inclusion dependencies, but disallows negation on right-hand-sides). However, several additional issues for $\mathcal{CFD}_{nc}^{\forall}$ merit further investigation. For one, it would be very desirable to incorporate at least a limited capacity for expressing existential restrictions relating to *inverse attributes* in $\mathcal{CFD}_{nc}^{\forall}$. This would allow one to fully reify roles in $\mathcal{CFD}_{nc}^{\forall}$ and, in this way, to reduce reasoning with roles to reasoning about concepts. Another more ambitious opportunity lies in incorporating limited forms of equational constraints while preserving tractability of reasoning. Finally, although the modelling utility is unclear, the consequences of allowing PFDs on the left-hand-sides of inclusion dependencies in $\mathcal{CFD}_{nc}^{\forall}$ is still open.

[3] Also submitted to DL14.

## References

1. Rakesh Agrawal, Alexander Borgida, and H. V. Jagadish. Efficient management of transitive relationships in large data and knowledge bases. In James Clifford, Bruce G. Lindsay, and David Maier, editors, *SIGMOD Conference*, pages 253–262. ACM Press, 1989.
2. 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. *J. Autom. Reasoning*, 39(3):385–429, 2007.
3. John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.
4. 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.
5. 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.
6. Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to query answering in DL-Lite. In *Principles of Knowledge Representation and Reasoning*, 2010.
7. Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to ontology-based data access. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2656–2661, 2011.
8. Carsten Lutz, Inanç Seylan, David Toman, and Frank Wolter. The combined approach to OBDA: Taming role hierarchies using filters. In *International Semantic Web Conference (1)*, pages 314–330, 2013.
9. Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2070–2075, 2009.
10. Mariano Rodriguez-Muro and Diego Calvanese. High performance query answering over DL-Lite ontologies. In Gerhard Brewka, Thomas Eiter, and Sheila A. McIlraith, editors, *KR*. AAAI Press, 2012.
11. Riccardo Rosati and Alessandro Almatelli. Improving query answering over DL-Lite ontologies. In Fangzhen Lin, Ulrike Sattler, and Miroslaw Truszczynski, editors, *KR*. AAAI Press, 2010.
12. David Toman and Grant E. Weddell. On keys and functional dependencies as first-class citizens in description logics. *J. Autom. Reasoning*, 40(2-3):117–132, 2008.
13. 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.
14. David Toman and Grant E. Weddell. Conjunctive Query Answering in $\mathcal{CFD}_{nc}$: A PTIME Description Logic with Functional Constraints and Disjointness. In *Australasian Conference on Artificial Intelligence*, pages 350–361, 2013.
15. David Toman and Grant E. Weddell. Pushing the $\mathcal{CFD}_{nc}$ Envelope. Technical report, Cheriton School of Computer Science, University of Waterloo, April 2014. Available at http://cs.uwaterloo.ca/ david/papers-dl14a.pdf.