On Limited Conjunctions and Partial Features in Parameter-Tractable Feature Logics

Stephanie McIntyre,[1] **Alexander Borgida,**[2] **David Toman,**[1] **Grant Weddell**[1]
[1]Cheriton School of Computer Science, University of Waterloo, Canada
[2]Department of Computer Science, Rutgers University, NJ, U.S.A.
{srmcinty,david,gweddell}@uwaterloo.ca, borgida@cs.rutgers.edu

Abstract

Standard reasoning problems are complete for EXPTIME in common feature-based description logics—ones in which all roles are restricted to being functions. We show how to control conjunctions on left-hand-sides of subsumptions and use this restriction to develop a parameter-tractable algorithm for reasoning about knowledge base consistency. We then show how the resulting logic can simulate partial features, and present algorithms for efficient query answering in that setting.

1 Introduction

Ontology-based data access (OBDA) emphasizes the use of ontologies, usually expressed in a *description logic* (DL), as a preferred front end for interacting with (multiple) databases, containing facts about a domain (Calvanese et al. 2007; Kontchakov et al. 2010; Lutz et al. 2013). A desirable DL for OBDA supports (i) more expressive conceptual modelling of the domain, (ii) capturing domain semantics embedded in relational schema, and (iii) effective query answering. The $\mathcal{CFD}$ family of feature-based DLs has been designed primarily to support PTIME reasoning in accessing relational data sources. A distinguishing property of this family is support for expressing complex functional dependencies, which are the most widely used way to capture domain semantics in relational databases, in addition to foreign keys. One dialect, called $\mathcal{CFDI}_{nc}^{\forall-}$ (St. Jacques, Toman, and Weddell 2016), supported OBDA to relational databases, and was able to do so without "loading" the relational database into an ABox. In addition, it was capable of emulating DL-Lite$_{core}^{\mathcal{F}}$.

Our main contribution is a new parameterized member of this family called $\mathcal{CFDI}_{kc}^{\forall-}$, which adds the ability to use conjunctions on the left-hand side of subsumptions in a $\mathcal{CFDI}_{nc}^{\forall-}$ TBox, where the parameter k is a positive integer that constitutes a limit on the number of conjunctions that need to be considered in reasoning services. (Indeed, any $\mathcal{CFDI}_{nc}^{\forall-}$ TBox can be easily mapped to a $\mathcal{CFDI}_{2c}^{\forall-}$ TBox.) This enhances the modelling capacity of $\mathcal{CFDI}_{kc}^{\forall-}$: in a university context, we can now not only specify that a `StudentWorker` is both a `Student` and an `Employee`, but actually *define* `StudentWorker` as anyone who is both, by adding the axiom

$$(\texttt{Student} \sqcap \texttt{Employee}) \sqsubseteq \texttt{StudentWorker}.$$

This distinction between "primitive concepts" ("phones, which happen to all be black") and "defined concepts" ("black phones") was one of the key insights that drove Brachman (Brachman 1977) to the development of KL-ONE, the progenitor of DLs, and was an important missing ingredient in semantic data models (Hull and King 1987), such as Taxis (Mylopoulos, Bernstein, and Wong 1980) and GEM (Zaniolo 1983), as well as UML.

A frequently remarked limitation of the $\mathcal{CFD}$ family, including $\mathcal{CFDI}_{nc}^{\forall-}$, is that features connecting objects are *total* functions. An interesting and important benefit of allowing limited conjunctions on the left-hand side of subsumption will be the ability to encode and support reasoning with features that are instead *partial* functions. This means that we can now explicitly require that a `Building` never has a `salary`. In relational database terms, this means that $\mathcal{CFDI}_{kc}^{\forall-}$ can indirectly represent "null inapplicable", in addition to the usual "null unknown", which comes from the open world assumption of DLs. In particular, we also introduce *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$ variants of $\mathcal{CFDI}_{kc}^{\forall-}$ DLs, which add an ability to capture partial features, and show how any *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$ terminology can always be mapped to a $\mathcal{CFDI}_{(k+1)c}^{\forall-}$ terminology. The remaining technical contributions of this paper are to show that each of the following problems are parameter-tractable in k:

1. (*parameter diagnosis*) given an arbitrary $\mathcal{CFDI}^{\forall}$ TBox $\mathcal{T}$ and integer k, determining if $\mathcal{T}$ is a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox;
2. (*concept satisfiability*) given a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox and concept C, determining if C is satisfiable;
3. (*knowledge base consistency*) determining if a given $\mathcal{CFDI}_{kc}^{\forall-}$ knowledge base is consistent; and
4. (*query answering in OBDA*) computing the certain answers for conjunctive queries over a given $\mathcal{CFDI}_{kc}^{\forall-}$ knowledge base.

Note that, similarly to other approaches to defining (parameter-)tractable fragments of first-order logic, e.g., (Simancik, Motik, and Horrocks 2014; Bienvenu et al. 2017;

SYNTAX	SEMANTICS: "$(\cdot)^{\mathcal{I}}$"
$C ::= A$	$A^{\mathcal{I}} \subseteq \triangle$
$\mid \forall \mathsf{Pf}.C$	$\{x \mid \mathsf{Pf}^{\mathcal{I}}(x) \in C^{\mathcal{I}}\}$
$\mid C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
$D ::= C$	$C^{\mathcal{I}} \subseteq \triangle$
$\mid \bot$	$\emptyset$
$\mid \exists f^{-1}$	$\{x \mid \exists y \in \triangle : f^{\mathcal{I}}(y) = x\}$
$\mid \forall \mathsf{Pf}.D$	$\{x \mid \mathsf{Pf}^{\mathcal{I}}(x) \in D^{\mathcal{I}}\}$
$\mid C : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$	$\{x \mid \forall y \in C^{\mathcal{I}} : (\bigwedge_{i=1}^{k} \mathsf{Pf}_i^{\mathcal{I}}(x) = \mathsf{Pf}_i^{\mathcal{I}}(y)) \Rightarrow \mathsf{Pf}^{\mathcal{I}}(x) = \mathsf{Pf}^{\mathcal{I}}(y)\}$

Figure 1: $\mathcal{CFDI}^{\forall}$ Concepts.

Calì, Gottlob, and Pieris 2012), $\mathcal{CFDI}^{\forall}$ TBoxes need to be globally analyzed to determine whether they satisfy the $\mathcal{CFDI}_{kc}^{\forall -}$ restrictions (see Theorem 6 in Section 3).

2 Background and Definitions

All members of the $\mathcal{CFD}$ family are fragments of FOL with underlying signatures based on disjoint sets of unary predicate symbols PC, called *primitive concepts*, constant symbols IN, called *individuals*, and unary function symbols F, called *features*. A *path function* Pf is a word in F*, with the empty word denoted by id, and concatenation by ".". *Concept descriptions* of two kinds, C and D, are defined by the grammar rules on the left-hand-side of Fig. 1. An instance of the final production is called a *path functional dependency* (PFD). (Note that PFD-like constructs have also been considered in versions of DL-Lite (Calvanese et al. 2008).)

Semantics is defined in the standard way with respect to an interpretation $\mathcal{I} = (\triangle, (\cdot)^{\mathcal{I}})$ that fixes the meaning of symbols in PC, IN, F. Here, features are interpreted as total functions. The interpretation function $\mathcal{I}$ is extended to path expressions by interpreting id as the identity function $\lambda x.x$, concatenation as function composition, and to complex concept descriptions C or D as per Fig. 1. An interpretation $\mathcal{I}$ satisfies a *subsumption* $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, a *concept assertion* $A(a)$ if $a^{\mathcal{I}} \in A^{\mathcal{I}}$, and a *path function assertion* $\mathsf{Pf}_1(a) = \mathsf{Pf}_2(b)$ if $\mathsf{Pf}_1^{\mathcal{I}}(a^{\mathcal{I}}) = \mathsf{Pf}_2^{\mathcal{I}}(b^{\mathcal{I}})$. A *knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ consists of a TBox $\mathcal{T}$ of subsumptions, and an ABox $\mathcal{A}$ of assertions. $\mathcal{I}$ satisfies $\mathcal{K}$ if it satisfies each subsumption and assertion in $\mathcal{K}$.

Example 1 Consider the relational schema for a university database in Fig. 2 where key attributes are underlined for brevity, and SQL keywords are in boldface. In constructing a $\mathcal{CFDI}^{\forall}$ conceptual model for this, we start with primitive concepts for each table (e.g., Building, Room, Employee, Student, and StudentWorker). So called "concrete" features are used to record columns of tables (e.g., bname, roomNr, inBldg, and salary). Functional relationships between instances of concepts are captured using "abstract" features, whose identifiers come from the name of the foreign key constraints (e.g., caretakerRef, office, hasMgrRef). Domain constraints on table columns are captured by value restrictions

```
Building( bname, campus );

Room( roomNr, inBldg, caretaker,
    constraint caretakerRef foreign key
        ( caretaker ) references Employee,
    constraint inBldgRef foreign key
        ( inBldg ) references Building );

Employee( name, salary, roomNr, inBldg,
    constraint office foreign key
        ( roomNr, inBldg ) references Room )

Prof( name, level,
    constraint nameRef foreign key
        ( name ) references Employee );

Student( snum, name, hasMgr, gpa,
    constraint hasMgrRef foreign key
        ( hasMgr ) references Employee );

materialized view StudentWorker as
    select * from Student natural join Employee
```

Figure 2: A University Database Schema.

on concrete features, as in Room $\sqsubseteq \forall$roomNr.Integer. Ordinary key constraints are captured by PFDs, as in the following one for buildings: Building $\sqsubseteq$ Building : bname $\to id$.

The identification of Rooms presents a more interesting case, since they are so-called "weak entities" in the Entity Relationship model: they require a local part for the key (roomNr), and the key of some other entity, Building in this case. This is captured by the more complex PFD Room $\sqsubseteq$ Room : roomNr, inBldgRef.bname $\to id$. Single-attribute foreign keys are captured by value restrictions on abstract features naming the foreign key constraint, as in Room $\sqsubseteq \forall$caretakerRef.Employee.[1] The complex foreign key office for Employee requires the value restriction Employee $\sqsubseteq \forall$office.Room and an additional more complex PFD:

$$\texttt{Employee} \sqsubseteq \texttt{Employee} : \texttt{roomNr}, \texttt{inBldg} \to \texttt{office}.$$

Note that binary relationships that are not functional in either direction (called "N-M relationships" in database circles), as well as n-ary relationships, can be represented using reification: if employees could have multiple offices, and, conversely, an office could house multiple people, then this would be modelled as the concept Occupancy, with features of and by, restrictions Occupancy $\sqsubseteq \forall$of.Room and Occupancy $\sqsubseteq \forall$by.Employee, and a crucial PFD

$$\texttt{Occupancy} \sqsubseteq \texttt{Occupancy} : \texttt{of}, \texttt{by} \to id,$$

ensuring that every (room,employee)-pair is represented by at most one instance of Occupancy.

[1]Explicit naming of foreign key constraints, e.g., "**constraint** office ...", is part of the SQL standard (International Organization for Standardization 2016).

The situation where the primary key of a table is also a foreign key is the usual representation of subclass hierarchies, so we add the axiom $\texttt{Prof} \sqsubseteq \texttt{Employee}$.

The following are examples of domain semantics not captured by traditional conceptual models, such as UML, but attainable using PFDs:

- (StudentWorker*s managed by the same person must work in the same office*) $\texttt{StudentWorker} \sqsubseteq \texttt{StudentWorker} : \texttt{hasMgr} \rightarrow \texttt{office}$
- (where professors are another subclass of Employee, Prof*s must have their own individual offices*) $\texttt{Prof} \sqsubseteq \texttt{Employee} : \texttt{office} \rightarrow \mathit{id}$
- (*all rooms in a building must have the same caretaker*) $\texttt{Room} \sqsubseteq \texttt{Room} : \texttt{inBldg} \rightarrow \texttt{caretaker}$

Once we add to the language conjunctions on the left-hand side of subsumptions, we will also be able to *define* StudentWorker, as discussed in the Introduction, capturing the view definition. Note that since the view is materialized, it should be modelled, since accessing it is faster than performing the join that defines it.

Since features are total functions, at this point each Building has a salary, etc., which is an unpleasant aspect. Once we introduce partial features and limited conjunctions on the lhs, we will be able to restrict the domain of features in the ontology, by stating, for example, $(\texttt{Building} \sqcap \exists \texttt{salary}) \sqsubseteq \bot$.

It is easy to see that for every $\mathcal{CFDI}^{\forall}$ KB $(\mathcal{T}, \mathcal{A})$, there is a simplified *normal form*: a conservative extension $(\mathcal{T}', \mathcal{A}')$ in which subsumptions in $\mathcal{T}'$ adhere to the form $C \sqsubseteq D$ where the structure of concepts C and D are now given by the following:

$$
\begin{array}{lcl}
C & ::= & A \mid \forall f.A \mid A_1 \sqcap A_2 \\
D & ::= & A \mid \bot \mid \forall f.A \mid \exists f^{-1} \mid A : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}
\end{array}
$$

Hereon, we also assume w.l.o.g. that at least one of the concept descriptions C and D is a primitive concept, and that the ABox $\mathcal{A}'$ contains only assertions of the form "$A(a)$", "$a.f = b$", and "$a = b$". Note that such a normalization of ABoxes leads to the introduction of additional constant symbols that, from the point of query answering, should behave the same way anonymous objects do (and thus be excluded from query answers by appeal to straightforward housekeeping checks).

Unfortunately, unrestricted use of the concept constructors in Fig. 1 leads to intractability of checking KB consistency and logical implication (Toman and Weddell 2005). As usual, to ensure PTIME complexity, one looks for additional restrictions on concept constructors. One restriction, which has been investigated (e.g., (Khizder, Toman, and Weddell 2000)), is to limit the PFD constructor to one of the following two forms:

1. $C : \mathsf{Pf}_1, \ldots, \mathsf{Pf}.\mathsf{Pf}_i, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}$ or
2. $C : \mathsf{Pf}_1, \ldots, \mathsf{Pf}.f, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}.g$

Note that this form continues to allow all the examples of PFD concepts introduced above, including keys and ordinary functional dependencies. The second restriction, introduced in this paper, limits the use of conjunction $\sqcap$ with a parameter k as follows:

Definition 2 (Restricted Conjunction) *Let $k > 0$ be a constant. We say that TBox $\mathcal{T}$ is a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox if, whenever $\mathcal{T} \models (A_1 \sqcap \cdots \sqcap A_n) \sqsubseteq B$ for some set of primitive concepts $\{A_1, \ldots, A_n\} \cup \{B\}$, with $n > k$, then $\mathcal{T} \models (A_{i_1} \sqcap \cdots \sqcap A_{i_k}) \sqsubseteq B$ for some k-sized subset $\{A_{i_1}, \ldots, A_{i_k}\}$ of the primitive concepts $\{A_1, \ldots, A_n\}$.* $\square$

Note that this condition is not syntactic, and we will return to the issue of checking it later, showing that this can be done in time exponential in k, but linear in $|\mathcal{T}|$. Adding these restrictions leads to the logic $\mathcal{CFDI}^{\forall-}_{kc}$.

Relation to $\mathcal{CFDI}^{\forall-}_{nc}$. It is relatively easy to see that $\mathcal{CFDI}^{\forall-}_{nc}$ (Toman and Weddell 2014), a logic that *disallows* conjunctions on the left-hand sides of subsumptions altogether but allows for primitive negation on the right-hand sides can be embedded into $\mathcal{CFDI}^{\forall-}_{2c}$ by mapping $A \sqsubseteq \neg B$ to $A \sqcap B \sqsubseteq \bot$ and keeping the remainder of a $\mathcal{CFDI}^{\forall-}_{nc}$ TBox unchanged. It is then easy to verify that the above transformation always yields a $\mathcal{CFDI}^{\forall-}_{2c}$ TBox since the only subsumptions with a conjunction on their left-hand sides are of the form $A \sqcap B \sqsubseteq \bot$. This embedding also shows that the expressive power of $\mathcal{CFDI}^{\forall-}_{nc}$ falls strictly between $\mathcal{CFDI}^{\forall-}_{1c}$ (which is unable to capture disjointness) and $\mathcal{CFDI}^{\forall-}_{2c}$ (which can in addition define certain binary intersections of concepts).

3 Reasoning in $\mathcal{CFDI}^{\forall-}_{kc}$

We now present our main result: showing that the complexity of reasoning in $\mathcal{CFDI}^{\forall-}_{kc}$ is in PTIME for a fixed k. The presentation proceeds in two steps. The first shows how concept consistency w.r.t. a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox can be decided, and the second extends this to full KB consistency.

3.1 TBox and Concept Satisfiability

It is easy to see that every $\mathcal{CFDI}^{\forall-}_{kc}$ TBox $\mathcal{T}$ is consistent (by setting all primitive concepts to be interpreted as the empty set). To test for *primitive concept satisfiability* we use the following construction for the closure of relevant inferred subsumptions:

Definition 3 (TBox Closure) *Let $\hat{E}$, $\hat{F}$, $\hat{G}$ and $\hat{H}$ be sets of primitive concepts (including $\bot$) of size at most k, or sets of value restrictions involving a common feature f over such a set of concepts (written as $\forall f.\hat{E}$, etc.; all the sets representing* conjunctions *of their elements); and $\mathcal{T}$ a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox in normal form. Then define $\mathsf{Clos}(\mathcal{T})$ to be a set of "small" subsumptions entailed by $\mathcal{T}$. In particular, let it be the least set such that*

1. *$\hat{E} \sqsubseteq \hat{E}$ and $\bot \sqsubseteq \hat{E}$ are in $\mathsf{Clos}(\mathcal{T})$ for every set $\hat{E}$;*
2. *If $\hat{E} \sqsubseteq \hat{F}$ and $\hat{F} \sqsubseteq \hat{G}$ are in $\mathsf{Clos}(\mathcal{T})$ then so is $\hat{E} \sqsubseteq \hat{G}$;*
3. *If $\hat{E} \sqsubseteq \hat{F}$ and $\hat{E} \sqsubseteq \hat{G}$ are in $\mathsf{Clos}(\mathcal{T})$ then so is $\hat{E} \sqsubseteq \hat{H}$ for all $\hat{H} \subseteq \hat{F} \cup \hat{G}$ of size between 1 and k;*
4. *If $C \sqsubseteq D \in \mathcal{T}$, for D not of the form $\exists f^{-1}$, and $C \subseteq \hat{E}$, then $\hat{E} \sqsubseteq D \in \mathsf{Clos}(\mathcal{T})$;*
5. *$\forall f.\bot \sqsubseteq \bot$ and $\bot \sqsubseteq \forall f.\bot$ are in $\mathsf{Clos}(\mathcal{T})$;*

6. *If $\hat{E} \sqsubseteq \hat{F}$ is in $\mathsf{Clos}(\mathcal{T})$ then so is $\forall f.\hat{E} \sqsubseteq \forall f.\hat{F}$;*
7. *If $A \sqsubseteq \exists f^{-1} \in \mathcal{T}$ and $\hat{E} \sqsubseteq A, \forall f.\hat{E} \sqsubseteq \forall f.\hat{F} \in \mathsf{Clos}(\mathcal{T})$ then $\hat{E} \sqsubseteq \hat{F} \in \mathsf{Clos}(\mathcal{T})$.*

Note that $\mathsf{Clos}(\mathcal{T})$ is polynomial in $|\mathcal{T}|$ and exponential in k. And clearly each subsumption added to $\mathsf{Clos}(\mathcal{T})$ by rules (1-7) in Definition 3 is logically implied by $\mathcal{T}$. We also note that while the above construction will be sufficient to provide the desired complexity bounds, the resulting $\mathsf{Clos}(\mathcal{T})$ is by no means the smallest such set. For example, one could complicate the rules in order to make all right-hand sides singletons, or by not explicitly representing all weakenings.

Theorem 4 (Primitive Concept Satisfiability) *Let $\mathcal{T}$ be a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox in normal form and A a primitive concept. Then A is satisfiable with respect to $\mathcal{T}$ if and only if $A \sqsubseteq \bot \notin \mathsf{Clos}(\mathcal{T})$.*

Proof (sketch): One direction is immediate: were $A \sqsubseteq \bot$ in $\mathsf{Clos}(\mathcal{T})$ it would be logically implied by $\mathcal{T}$ and hence there couldn't be a model of $\mathcal{T}$ in which A is nonempty. For the other direction, given $\mathsf{Clos}(\mathcal{T})$, an object o, and a primitive concept A, define the following family of subsets of PC indexed by paths of features and their inverses, starting from o, in the following recursive manner:

1. $S_o = \{B \mid A \sqsubseteq B \in \mathsf{Clos}(\mathcal{T})\}$;
2. $S_{f(x)} = \{B \mid \hat{E} \sqsubseteq \forall f.B \in \mathsf{Clos}(\mathcal{T})$ and $\hat{E} \subseteq S_x\}$, when $f \in \mathsf{F}$ and x not of the form "$f^-(y)$"; and
3. $S_{f^-(x)} = \{B \mid \forall f.\hat{E} \sqsubseteq B \in \mathsf{Clos}(\mathcal{T})$ and $\hat{E} \subseteq S_x\}$, when $A' \sqsubseteq \exists f^{-1} \in \mathcal{T}$, $A' \in S_x$, and x not of the form "$f(y)$".

We say that S_x is *defined* if it conforms to one of the three cases above, and that it is *consistent* if $\bot \notin S_x$. Observe that all defined sets S_x are consistent. Otherwise, A must be inconsistent, implying in turn that $A \sqsubseteq \bot \in \mathsf{Clos}(\mathcal{T})$, a contradiction. Hence the defined sets S_x induce a (tree) model of $\mathcal{T}$, in which $o \in A^{\mathcal{I}}$. □

Note that the above model witnessing the satisfiability of A does not contain any identical path agreements, and hence vacuously satisfies all PFDs in $\mathcal{T}$.

Observation 5 *Let $\mathcal{T}$ be a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox in normal form and $A_1, \ldots, A_n$ primitive concepts. Then $A_1 \sqcap \ldots \sqcap A_n$ is satisfiable with respect to $\mathcal{T}$ if and only if A is satisfiable with respect to $\mathcal{T} \cup \{A \sqsubseteq A_1, \ldots, A \sqsubseteq A_n\}$, for a fresh primitive concept A, since $\mathcal{T} \cup \{A \sqsubseteq A_1, \ldots, A \sqsubseteq A_n\}$ is a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox whenever $\mathcal{T}$ is a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox.* □

This observation allows *consistency checks* for arbitrary conjunctions, including cases that may not appear in $\mathcal{CFDI}_{kc}^{\forall-}$ TBoxes (for a particular fixed k).

On determining k: a pay as you go approach. The above development assumes a fixed k known in advance. However, the TBox closure procedure also allows for testing whether a given value of k is sufficient for a $\mathcal{CFDI}^{\forall}$ TBox $\mathcal{T}$:

Theorem 6 (Testing for k) *Let $\mathcal{T}$ be a $\mathcal{CFDI}^{\forall}$ TBox. Then $\mathcal{T}$ is* not *a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox if and only if there are $\hat{E}, \hat{F}, \hat{G}, D$ such that (1) $\hat{E} \sqsubseteq \hat{F} \in \mathsf{Clos}(\mathcal{T})$, (2) $\hat{G} \sqsubseteq D \in \mathsf{Clos}(\mathcal{T})$, (3) $\hat{F} \subseteq \hat{G}$, (4) $|\hat{E} \cup (\hat{G} - \hat{F})| > k$, and (5) for all $\hat{H} \sqsubseteq D \in \mathsf{Clos}(\mathcal{T})$ we have $\hat{H} \not\subseteq \hat{E} \cup (\hat{G} - \hat{F})$.*

Proof (sketch): We have remarked already that all subsumptions in $\mathsf{Clos}(\mathcal{T})$ are entailed by $\mathcal{T}$. Hence $\hat{E} \cup (\hat{G} - \hat{F}) \sqsubseteq D$ is also logically implied by $\mathcal{T}$. Since there is no $\hat{H} \not\subseteq \hat{E} \cup (\hat{G} - \hat{F})$ such that $\hat{H} \sqsubseteq D \in \mathsf{Clos}(\mathcal{T})$ either $\hat{E} \cup (\hat{G} - \hat{F}) \sqsubseteq D$ violates the conditions in Definition 2, or we failed to derive one of the $\hat{H} \sqsubseteq D$ logically implied by $\mathcal{T}$. However, were $\mathcal{T}$ a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox such that $\hat{H} \sqsubseteq D \notin \mathsf{Clos}(\mathcal{T})$, one could then construct a model, similar to the construction in the proof of Theorem 4, in which $\hat{H} \sqsubseteq D$ doesn't hold, a contradiction with the assumption that $\mathcal{T}$ a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox. □

Note that this Theorem leads to an algorithm based on iteratively increasing k and testing whether $\mathcal{T}$ is a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox using the above Theorem. The algorithm terminates when $\mathcal{T}$ is a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox.[2] Note also that successive iterations easily reuse subsumptions from the previous level, and in this way can avoid unnecessary recomputation of all such subsumptions.

3.2 Knowledge Base Consistency

Inverse features affect how PFDs interact with an ABox. In particular, PFDs in which all path functions have a common prefix, i.e., of the form

$$A \sqsubseteq B : f.\,\mathsf{Pf}_1, \ldots, f.\,\mathsf{Pf}_k \to f.\,\mathsf{Pf}$$

may apply to (pairs of) anonymous individuals mandated by the existence of inverse features (f in this case). In general, to enforce PFDs with respect to an ABox *while avoiding any need to explicitly create anonymous predecessor objects*, we add additional logically implied PFDs to a given TBox as follows:

Definition 7 (PFD Enrichment for Inverses) *Let $\mathcal{T}$ be a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox in normal form,*

$$A \sqsubseteq B : f.\,\mathsf{Pf}_1, \ldots, f.\,\mathsf{Pf}_k \to f.\,\mathsf{Pf} \in \mathcal{T}$$

($A \sqsubseteq B : f.\,\mathsf{Pf}_1, \ldots, f.\,\mathsf{Pf}_k \to id \in \mathcal{T}$) with $\mathsf{Pf}_i \neq id$ for $1 \leq i \leq k$. Then we require that $A \sqsubseteq \forall f.A'$, $B \sqsubseteq \forall f.B'$, and $A' \sqsubseteq B' : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$ ($A' \sqsubseteq B' : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to id$), where A' and B' are fresh primitive concepts, also be in $\mathcal{T}$. □

For further details on this, see (St. Jacques, Toman, and Weddell 2016). The first step of deciding KB consistency, ABox completion, is defined by the rules in Fig. 3. In particular, the rules extend a given ABox with all implied concept memberships and feature agreements.

Definition 8 *Let $(\mathcal{T}, \mathcal{A})$ be a $\mathcal{CFDI}_{kc}^{\forall-}$ KB. We define an ABox $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$ to be the least ABox $\mathcal{A}'$ closed under the rules in Fig. 3 such that $\mathcal{A} \subseteq \mathcal{A}'$.* □

[2]Note that the algorithm terminates for an arbitrary $\mathcal{T}$ since every TBox $\mathcal{T}$ is a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox for $k = |\mathcal{T}|$.

If a appears in $\mathcal{A}$ then add $a = a$ to $\mathcal{A}$ If $a = b, \varphi \in \mathcal{A}$ then add $\varphi[b/a]$ to $\mathcal{A}$
If $a = b \in \mathcal{A}$ then add $b = a$ to $\mathcal{A}$ If $a.f = b, a.f = c \in \mathcal{A}$ then add $b = c$ to $\mathcal{A}$

(a) ABox Equality Interactions

If $A_1(a), \ldots, A_k(a) \in \mathcal{A}$ and $\mathcal{T} \models A_1 \sqcap \ldots \sqcap A_k \sqsubseteq B$ then add $B(a)$ to $\mathcal{A}$
If $\{A(a), a.f = b\} \subseteq \mathcal{A}$ and $\mathcal{T} \models A \sqsubseteq \forall f.B$ then add $B(b)$ to $\mathcal{A}$
If $\{A(a), b.f = a\} \subseteq \mathcal{A}$ and $\mathcal{T} \models \forall f.A \sqsubseteq B$ then add $B(b)$ to $\mathcal{A}$

(b) ABox–TBox Interactions

If (i) $A \sqsubseteq B : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf} \in \mathcal{T}$, (ii) $A(a), B(b) \in \mathcal{A}$, and (iii) for $1 \leq i \leq k$, there exists a prefix Pf'_i of Pf_i s.t. $a.\mathsf{Pf}'_i = c_i, b.\mathsf{Pf}'_i = d_i, c_i = d_i \in \mathcal{A}$ then:

1. If $a.\mathsf{Pf} = c, b.\mathsf{Pf} = d \in \mathcal{A}$ and $c = d \notin \mathcal{A}$ then add $c = d$ to $\mathcal{A}$; or
2. If Pf is of the form $\mathsf{Pf}''.f$ and $a.\mathsf{Pf}'' = c, b.\mathsf{Pf}'' = d \in \mathcal{A}$, and $c = d \notin \mathcal{A}$ then (i) if $a.\mathsf{Pf} = c' \in \mathcal{A}$ then add $d.f = c'$ to $\mathcal{A}$, else (ii) if $b.\mathsf{Pf} = d' \in \mathcal{A}$ then add $c.f = d'$ to $\mathcal{A}$, otherwise (iii) add $c.f = e, d.f = e$ to $\mathcal{A}$, where e *is a new individual.*

(c) ABox–PFD Interactions

Figure 3: ABox Completion Rules for $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$

Observe that individuals can only be declared to be members of primitive concepts since $\mathcal{A}$ is in normal form. It is also easy to see that completion terminates since it can add at most $|\mathcal{T}||\mathcal{A}|^2$ new objects, one for every pair of existing objects and a feature name.

What remains to verify is that for every ABox object a the set of primitive concepts $\{A \mid A(a) \in \mathsf{Completion}_{\mathcal{T}}(\mathcal{A})\}$ is satisfiable with respect to $\mathcal{T}$. We use Corollary 5 to test this condition for each object a appearing in $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$.

Theorem 9 (**$\mathcal{CFDI}_{kc}^{\forall-}$ KB Consistency**) *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a $\mathcal{CFDI}_{kc}^{\forall-}$ KB. Then $\mathcal{K}$ is consistent if and only if $\{A \mid A(a) \in \mathsf{Completion}_{\mathcal{T}}(\mathcal{A})\}$ is satisfiable with respect to $\mathcal{T}$ for every "a" appearing in* $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$. □

It is easy to see that the above construction can be implemented to run in $\mathcal{O}(|\mathcal{T}|^k + |\mathcal{T}||\mathcal{A}|^2)$. Other reasoning problems for $\mathcal{CFDI}_{kc}^{\forall-}$, such as *logical implication*, $\mathcal{T} \models C \sqsubseteq D$, are reduced to KB consistency in the standard way.

4 Partial Features

In this section we utilize our newfound ability to accommodate limited conjunctions on the left-hand sides of subsumptions to introduce *partial features* in $\mathcal{CFDI}_{kc}^{\forall-}$. We start with modifying the syntax and semantics for this purpose.

Definition 10 (Partial Features)

1. *Features $f \in \mathsf{F}$ are now interpreted as* partial *functions on $\triangle$ (i.e., the result can be* undefined *for some elements of $\triangle$);*
2. *The semantics of path function* Pf *denotes a partial function resulting from the composition of partial functions.*
3. *The syntax of C in feature-based DLs is extended with an additional concept constructor of the form "$\exists f$", called an* existential restriction *that can then appear on both sides of subsumptions.*
4. *The $\exists f$ concept constructor is interpreted as $\{x \mid \exists y \in \triangle. f^{\mathcal{I}}(x) = y\}$.*
5. *We adopt a* strict *interpretation of set membership and equality. This means that set membership holds only when the value exists; and equality holds only when both sides are defined and denote the same object.* □

There are several observations worth making at this point. First, as a consequence of 2 and 5, the semantics of value restrictions and PFDs coincide with the original semantics when features were interpreted as total functions. Note also that our PFDs agree with the definition of identity constraints in (Calvanese et al. 2008), where $\mathsf{Pf}_0 = id$, which also require path values to exist. To further clarify the impact of this observation note that a *PFD subsumption* of the form "$C_1 \sqsubseteq C_2 : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}_0$" is violated when (a) all path functions $\mathsf{Pf}_0, \ldots, \mathsf{Pf}_k$ are defined for a C_1 object e_1 and a C_2 object e_2, and (b) $\mathsf{Pf}_i^{\mathcal{I}}(e_1) = \mathsf{Pf}_i^{\mathcal{I}}(e_2)$ holds only for $1 \leq i \leq k$. Formally, and more explicitly, this leads to the following interpretation of PFDs in the presence of partial features:

$$
\begin{aligned}
&(C : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}_0)^{\mathcal{I}} = \\
&\{x \mid \forall y. y \in C^{\mathcal{I}} \wedge x \in (\exists \mathsf{Pf}_0)^{\mathcal{I}} \wedge y \in (\exists \mathsf{Pf}_0)^{\mathcal{I}} \wedge \\
&\quad \textstyle\bigwedge_{i=1}^{k} (x \in (\exists \mathsf{Pf}_i)^{\mathcal{I}} \wedge y \in (\exists \mathsf{Pf}_i)^{\mathcal{I}} \wedge \mathsf{Pf}_i^{\mathcal{I}}(x) = \mathsf{Pf}_i^{\mathcal{I}}(y)) \\
&\qquad\qquad \to \mathsf{Pf}_0^{\mathcal{I}}(x) = \mathsf{Pf}_0^{\mathcal{I}}(y) \,\}.
\end{aligned}
$$

Second, as a consequence of item 5, we have the tautology $\forall f.E \sqsubseteq \exists f$ for arbitrary f and E (in other words, $\exists f$ is a top-free notation for $\forall f.\top$).

Finally, since features are still *functional*, so-called "qualified existential restrictions" of the form "$\exists f.C$", with semantics given as follows:

$$(\exists f.C)^{\mathcal{I}} = \{x \mid \exists y \in \triangle. f^{\mathcal{I}}(x) = y \wedge y \in C^{\mathcal{I}}\},$$

are the same as "$\forall f.C$". Hence we will write "$(\exists \mathsf{Pf})$" as shorthand for "$(\exists f_1 \sqcap \forall f_1.(\exists f_2 \sqcap \forall f_2.(\ldots(\exists f_k)\ldots)))$".

Example 11 Now that we have partial functions, we can refine our conceptual model to more properly reflect the domain and range of features:

$$\begin{array}{l}(\texttt{Building} \sqcap \exists \texttt{salary}) \sqsubseteq \bot \\ (\texttt{Building} \sqcap \exists \texttt{inBldg}) \sqsubseteq \bot \\ \ldots \\ \texttt{Building} \sqsubseteq \exists \texttt{name} \sqcap \exists \texttt{campus} \sqcap \ldots\end{array}$$

The new constructor $\exists f$ naturally extends our *normal form* to *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$ TBoxes by allowing $\exists f$ to appear on both sides of subsumptions.

The following definition now shows how $\mathcal{CFDI}_{kc}^{\forall-}$ is able to simulate its extension with partial functions and existential restrictions in a straightforward manner. The idea is to introduce a new atomic concept G that intuitively stands for "objects in the total model that exist in the partial model".

Definition 12 *Let $\mathcal{T}$ be a partial-$\mathcal{CFDI}_{kc}^{\forall-}$ TBox in normal form. We then derive a $\mathcal{CFDI}_{(k+1)c}^{\forall-}$ TBox $\mathcal{T}'$ from $\mathcal{T}$ by applying the following rules:*

$$\begin{array}{rrcl} 1. & A \sqsubseteq \bot & \mapsto & A \sqcap G \sqsubseteq \bot \\ 2. & A \sqsubseteq B & \mapsto & A \sqcap G \sqsubseteq B \\ 3. & A \sqcap B \sqsubseteq C & \mapsto & A \sqcap B \sqcap G \sqsubseteq C \\ 4. & A \sqsubseteq \forall f.B & \mapsto & A \sqcap G \sqsubseteq \forall f.B \sqcap \forall f.G \\ 5. & \forall f.A \sqsubseteq B & \mapsto & \forall f.A \sqcap \forall f.G \sqsubseteq B \\ 6. & A \sqsubseteq \exists f & \mapsto & A \sqcap G \sqsubseteq \forall f.G \\ 7. & \exists f \sqsubseteq A & \mapsto & \forall f.G \sqsubseteq A \end{array}$$

and then by adding the subsumption $\forall f.G \sqsubseteq G$ to $\mathcal{T}'$. □

It is easy to verify that $\mathcal{T}'$, defined above, is a $\mathcal{CFDI}_{(k+1)c}^{\forall-}$ TBox and that:

Theorem 13 *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a partial-$\mathcal{CFDI}_{kc}^{\forall-}$ KB, and let $\mathcal{T}'$ be defined as above. Then $\mathcal{K}$ is consistent if and only if the $\mathcal{CFDI}_{(k+1)c}^{\forall-}$ KB $(\mathcal{T}', \mathcal{A} \cup \{G(a) \mid a$ appears in $\mathcal{A}\})$ is consistent.* □

Note that PFDs do not interact with TBox completion and are *applied* only in the process of ABox closure. Hence the only extension necessary is to verify that, in ABox-PFD interactions, the object e necessarily exists (all the other objects involved are explicitly in the ABox already). This is achieved by modifying the precondition on the objects a and b to $A \sqcap \exists \mathsf{Pf}(a)$ and $B \sqcap \exists \mathsf{Pf}(b)$, respectively.

Hereon, we assume that logical consequence with respect to *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$ TBoxes is reduced to KB unsatisfiability in the standard way.

4.1 On Value Restrictions

In *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$, the value restriction $\forall f.A$ inherits its definition from $\mathcal{CFDI}_{kc}^{\forall-}$, i.e., it is the set of all objects o that *have* a feature f and such that $f(o) \in A^{\mathcal{I}}$. Note that this does not impact the fact that features in *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$ are *partial*. For example, to express that A objects do *not have* feature f, one can say $A \sqcap \exists f \sqsubseteq \bot$. Similarly, to restrict the range of a partial feature *without making it total* for all A objects, we can say $A \sqcap \exists f \sqsubseteq \forall f.B$.

On the other hand, value restrictions in more traditional role-based description logics, such as $\mathcal{ALC}$, also cover the *vacuous cases*, containing objects for which f is undefined (in addition to the above). This definition unfortunately leads to computational difficulties: the *disjunctive* nature of such value restriction, when used on left-hand sides of subsumptions, destroys the canonical model property of the logic. This leads to intractability of query answering as shown by Calvanese *et al.* (Calvanese et al. 2013).

To regain tractability of our logic we would have to restrict the use of value restrictions on the left-hand side of subsumptions. In our normal form, we would have to replace $\forall f.A$ with $\forall f.A \sqcap \exists f$ in the grammar for left-hand side concepts. This would then lead to alternative rules when simulating the partial-feature logic in the total-feature counterpart in Definition 12:

$$\begin{array}{rrcl} 4'. & A \sqsubseteq \forall f.B & \mapsto & A \sqcap G \sqsubseteq \forall f.B \\ 5'. & (\forall f.A \sqcap \exists f) \sqsubseteq B & \mapsto & (\forall f.A \sqcap \forall f.G) \sqsubseteq B \end{array}$$

5 OBDA for *partial-*$\mathcal{CFDI}_{kc}^{\forall-}$

Conjunctive queries are, as usual, formed from atomic queries (or *atoms*) of the form $C(x)$ and $x.\mathsf{Pf}_1 = y.\mathsf{Pf}_2$, where x and y are variables, using conjunction and existential quantification. To simplify notation, we conflate conjunctive queries with the set of its constituent atoms and a set of *answer variables*:

Definition 14 (Conjunctive Query)
Let φ be a set of atoms (representing a conjunction) $C(x_i)$ and $x_{i_1}.\mathsf{Pf}_1 = x_{i_2}.\mathsf{Pf}_2$ (where C is a concept description), Pf_i path functions, and $\bar{x}$ a tuple of variables. We call the expression $\{\bar{x} \mid \varphi\}$ a conjunctive query *(CQ).* □

A conjunctive query $\{\bar{x} \mid \varphi\}$ is therefore a notational variant of the formula $\exists \bar{y}. \bigwedge_{\psi \in \varphi} \psi$ in which $\bar{y}$ contains all variables appearing in φ but not in $\bar{x}$. The usual definition of certain answers is given by the following:

Definition 15 (Certain Answer)
Let $\mathcal{K}$ be a partial-$\mathcal{CFDI}_{kc}^{\forall-}$ KB and $Q = \{\bar{x} \mid \varphi\}$ a CQ. A certain answer *to Q over $\mathcal{K}$ is a substitution of constant symbols $\bar{a}$, $[\bar{x} \mapsto \bar{a}]$, such that $\mathcal{K} \models Q[\bar{x} \mapsto \bar{a}]$.* □

As is the case with TBoxes and ABoxes, a CQ can be represented in a *normal form*, a form in which all atoms in the CQ are of the form "$A(x)$" or "$x.f = y$", where A is a primitive concept and f a feature. This can be easily achieved by introducing additional non-answer (existentially quantified) variables and primitive concepts equivalent to the complex ones in the query. For the remainder of the paper, we also assume CQs are always *connected*. (Evaluating disconnected CQs is easily achieved by considering each component separately.)

Example 16 A query asking for all students whose managers are professors is

$$\{(x) \mid \{\texttt{Student}(x), x.\texttt{hasMgrRef} = w, \texttt{Prof}(w)\}\}$$

The second step in query answering, following ABox completion, relies on query reformulation with respect to $\mathcal{T}$. This step is necessary to keep the data complexity of query answering in PTIME: $\mathcal{CFDI}_{kc}^{\forall-}$ can force *exponentially many* anonymous objects *with distinct class membership* to exist due to value restrictions using a construction similar to

1. If $\{A_1(x), \ldots, A_n(x)\} \subseteq \psi$ and $\mathcal{T} \models A_1 \sqcap \ldots \sqcap A_n \sqsubseteq \bot$ then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) - \{\{\bar{y} \mid \psi\}\}$.
2. If $\{x.f = y, x.f = z\} \subseteq \psi$ then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) - \{\{\bar{y} \mid \psi\}\} \cup \{\{\bar{y} \mid \psi\}[z/y]\}$.
3. If $\{x.f = z, y.f = z\} \subseteq \psi$ then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) \cup \{\{\bar{y} \mid \psi\}[x/y]\}$.
4. If $\{A_1(x), \ldots, A_n(x), B(x)\} \subseteq \psi$ and $\mathcal{T} \models A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B$ then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) - \{\{\bar{y} \mid \psi\}\} \cup \{\{\bar{y} \mid \psi - \{B(x)\}\}\}$.
5. If $\{x.f = y, A_1(y), \ldots, A_n(y)\} \subseteq \psi$ and y does not appear elsewhere in ψ nor in $\bar{y}$ then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) \cup \{\{\bar{y} \mid \psi'\}\}$ for all $\psi' = \psi - \{x.f = y, A_1(y), \ldots, A_n(y)\} \cup \{B_{0,i_0}(x), \ldots, B_{n,i_n}(x)\}$ for which $\mathcal{T} \models B_{0,i_0} \sqsubseteq \exists f$ and $\mathcal{T} \models B_{i,i_j} \sqsubseteq \forall f.A_i$ for $i > 0$ and where B_{i,i_j} are all such maximal primitive concepts w.r.t. $\sqsubseteq$.
6. If $\{y.f = x, A_1(y), \ldots, A_n(y)\} \subseteq \psi$ and y does not appear elsewhere in ψ nor in $\bar{y}$ then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) \cup \{\{\bar{y} \mid \psi'\}\}$, for all $\psi' = \psi - \{y.f = x, A_1(y), \ldots, A_n(y)\} \cup \{B_{0,i_0}(x), \ldots, B_{n,i_n}(x)\}$ for which $\mathcal{T} \models B_{0,i_0} \sqsubseteq \exists f^{-1}$ and $\mathcal{T} \models \forall f.B_{i,i_j} \sqsubseteq A_i$ for $i > 0$ and where B_{i,i_j} are all such maximal primitive concepts w.r.t. $\sqsubseteq$.

Figure 4: Query Rewriting Rules for $\{\bar{y} \mid \psi\} \in \mathsf{Fold}_{\mathcal{T}}(Q)$.

(St. Jacques, Toman, and Weddell 2016). While, in principle we could create a *representative* for each such possibility along the lines of the *combined approach* (Lutz, Toman, and Wolter 2009; Kontchakov et al. 2010), the resulting completed ABox would no longer be polynomial in the size of the original knowledge base. To avoid the need for such an "expensive" ABox completion, our approach treats matches to anonymous individuals by *query reformulation*—an idea first suggested in (Calvanese et al. 2007). Note however that our rewriting differs from "perfect rewriting": since we perform ABox completion for all objects in a given ABox (see Fig. 3) we can greatly simplify the query rewriting and make the result smaller since we no longer need to expand the query, e.g., w.r.t. concept hierarchies.

Definition 17 *Let $Q = \{\bar{y} \mid \varphi\}$ be a CQ. We write $\mathsf{Fold}_{\mathcal{T}}(Q)$ to denote the set of CQs (implicitly denoting the union of their results) that is obtained by applying exhaustively the rewrite rules in Fig. 4 to the initial set $\{\{\bar{y} \mid \varphi\}\}$.* □

The key idea underlying this definition is that, to find query answers, it is now sufficient to *match* queries in $\mathsf{Fold}_{\mathcal{T}}(Q)$ explicitly against the (completed) ABox $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$; matches outside the ABox are captured by query reformulation (rules 5 and 6); removed parts of the query (rules 1,2,4) are implied by $\mathcal{T}$; rule 3 makes possible the application of rules 5 and 6.

Lemma 18 *Let Q be a CQ with at least one answer variable. Then $\bar{a}$ is a certain answer to Q over $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if and only if $\bar{a}$ is an answer for some $\{\bar{x} \mid \psi\} \in \mathsf{Fold}_{\mathcal{T}}(Q)$ over $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$.*

Proof (sketch): By observing that the extended ABox $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$ is essentially a part of the minimal model of $\mathcal{K}$ (since $\mathcal{K}$ is Horn) and that every element of $\mathsf{Fold}_{\mathcal{T}}(Q)$ implies Q, it is easy to see that whenever (1-6) are satisfied, there is a match of Q in the minimal model and thus $\bar{a}$ is an answer. Conversely, if a match of Q 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 the answer variables must be bound to ABox individuals) and the remainder of the match must be forest-like. Hence, an element in $\mathsf{Fold}_{\mathcal{T}}(Q)$ matches in the ABox since the remaining conjuncts must be implied by $\mathcal{T}$. □

Example 19 Suppose we have an additional axiom in our TBox which specializes the range of `hasMgrRef` for `StudentWorkers` to be professors: `StudentWorker` $\sqsubseteq \forall$`hasMgrRef.Prof`. To find $\mathsf{Fold}_{\mathcal{T}}(Q)$ for the query in Example 16, we consider the subset $\{x.\texttt{hasMgrRef} = w, \texttt{Prof}(w)\}$ is replaced by $\{\texttt{StudentWorker}(x)\}$ by applying rule 5: using the above axiom and axioms in our original ontology, `StudentWorker` $\sqsubseteq \exists$`hasMgrRef` can be derived from `StudentWorker` $\sqsubseteq$ `Student` (from the definition of `StudentWorker`), and `Student` $\sqsubseteq \forall$`hasMgrRef.Employee` $\sqcap \exists$`hasMgrRef`, which is obtained from the foreign key `hasMgr`, once features become partial, as in Example 11. Then, the query becomes the set $\{\texttt{Student}(x), \texttt{StudentWorker}(x)\}$, which in turn is reduced to $\{\texttt{StudentWorker}(x)\}$, since the ontology has `StudentWorker` $\sqsubseteq$ `Student`.

For CQ without answer variables, we need an additional step that checks whether the query, when it can be *folded* to a concept (called C below) matches in the tree part of every model of $\mathcal{K}$. We use the KB consistency check to determine whether a concept is *forced* in a model of a KB. Thus, given a *partial-$\mathcal{CFDI}_{kc}^{\forall-}$* TBox $\mathcal{T}$ we check for every combination of primitive concepts $A_1, \ldots, A_k$ (that could potentially label an object in an ABox):

- $(\mathcal{T}, \{A_1(a), \ldots, A_k(a)\})$ is consistent, and
- $(\mathcal{T} \cup \{C \sqsubseteq \bot\}, \{A_1(a), \ldots, A_k(a)\})$ is not consistent.

If so we say that $A_1, \ldots, A_k$ force C in $\mathcal{T}$. To account for forcing we extend the query rewriting in Definition 17 as follows:

7. If $\bar{x} = \langle\rangle$ in $Q = \{\bar{x} \mid \varphi\}$ and φ is equivalent to a concept C. Then $\mathsf{Fold}(Q) := \mathsf{Fold}(Q) \cup \{\{\langle\rangle \mid A_1(x) \wedge \ldots \wedge A_k(x)\}$ for every combination of primitive concepts $A_1, \ldots, A_k$ that force C in $\mathcal{T}$.

This construction accounts for matches in the anonymous part of the minimal model of $\mathcal{K}$, and yields the following Lemma:

Lemma 20 *Let Q be a CQ without answer variables and $\mathcal{K}$ a partial-$\mathcal{CFDI}_{kc}^{\forall-}$ KB. Then $\mathcal{K} \models Q$ if and only if* *at least one* $\{\langle\rangle \mid \psi\} \in \mathsf{Fold}_{\mathcal{T}}(Q)$ *evaluates to true over* $\mathsf{Completion}_{\mathcal{T}}(\mathcal{A})$.

Proof (sketch): The first condition is similar to Lemma 18, the second allows for queries that can be folded into a concept to be realized completely outside of the (extended) ABox. Non-emptiness of the models of C indeed corresponds to finding an object that makes the query true in the minimal model. □

Combining the results in Lemmata 18 and 20 yields the needed query reformulation for all CQ (and in turn for all UCQ and other syntactically positive queries).

6 Summary and Future Work

The contributions in this paper are twofold. First, we introduce $\mathcal{CFDI}_{kc}^{\forall -}$, a new dialect of the $\mathcal{CFD}$ family of DLs admitting a limited use of concept conjunction in left-hand sides of TBox subsumption while maintaining parameterized tractability of reasoning. We show how this makes it possible to simulate partial functions, and also how conjunctive queries can be reformulated to enable OBDA over a $\mathcal{CFDI}_{kc}^{\forall -}$ KB. We have also combined our results with referring expressions to provide a richer framework in which to accomplish OBDA over relational data sources, and, in particular, to avoid *object id invention* needed, e.g., to capture PFD-generated equalities in Fig. 3 (these results are beyond the limits of this paper). Hence the techniques presented here are expected to perform as well or better than the experimental results reported in (St. Jacques, Toman, and Weddell 2016). For future work, we plan to explore how referring expressions can be used to account for other equalities outside of an ABox that more powerful DLs might induce, e.g, by limited use of the "same-as" concept constructor (Borgida and Patel-Schneider 1994).

Acknowledgments

We thank the anonymous reviewers for their suggestions, which helped considerably to improve the paper. The research was supported by NSERC Discovery Grants from the government of Canada.

References

Bienvenu, M.; Kikot, S.; Kontchakov, R.; Podolskii, V. V.; Ryzhikov, V.; and Zakharyaschev, M. 2017. The complexity of ontology-based data access with OWL 2 QL and bounded treewidth queries. In *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2017*, 201–216.

Borgida, A., and Patel-Schneider, P. F. 1994. A Semantics and Complete Algorithm for Subsumption in the CLASSIC Description Logic. *J. of AI Research* 1:277–308.

Brachman, R. J. 1977. What's in a concept: structural foundations for semantic networks. *International journal of man-machine studies* 9(2):127–152.

Calì, A.; Gottlob, G.; and Pieris, A. 2012. Towards more expressive ontology languages: The query answering problem. *Artif. Intell.* 193:87–128.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. Autom. Reasoning* 39(3):385–429.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2008. Path-Based Identification Constraints in Description Logics. In *KR*, 231–241.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2013. Data complexity of query answering in description logics. *Artif. Intell.* 195:335–360.

Hull, R., and King, R. 1987. Semantic database modeling: Survey, applications, and research issues. *ACM Comput. Surv.* 19(3):201–260.

International Organization for Standardization. 2016. SQL – Part 2: Foundation (SQL/Foundation). ISO/IEC 9075-2:2016.

Khizder, V. L.; Toman, D.; and Weddell, G. 2000. Reasoning about Duplicate Elimination with Description Logic. In *DOOD*, 1017–1032.

Kontchakov, R.; Lutz, C.; Toman, D.; Wolter, F.; and Zakharyaschev, M. 2010. The combined approach to query answering in DL-Lite. In *Proc. KR*, 247–257.

Lutz, C.; Seylan, I.; Toman, D.; and Wolter, F. 2013. The combined approach to OBDA: Taming role hierarchies using filters. In *ISWC (1)*, 314–330.

Lutz, C.; Toman, D.; and Wolter, F. 2009. Conjunctive query answering in the description logic EL using a relational database system. In *Proc. IJCAI*, 2070–2075.

Mylopoulos, J.; Bernstein, P. A.; and Wong, H. K. T. 1980. A language facility for designing database-intensive applications. *ACM Trans. Database Syst.* 5(2):185–207.

Simancik, F.; Motik, B.; and Horrocks, I. 2014. Consequence-based and fixed-parameter tractable reasoning in description logics. *Artif. Intell.* 209:29–77.

St. Jacques, J.; Toman, D.; and Weddell, G. E. 2016. Object-relational queries over $\mathcal{CFDI}_{nc}$ knowledge bases: OBDA for the SQL-Literate. In *Proc. IJCAI*, 1258–1264.

Toman, D., and Weddell, G. 2005. On Reasoning about Structural Equality in XML: A Description Logic Approach. *Theoretical Computer Science* 336(1):181–203.

Toman, D., and Weddell, G. E. 2014. On adding inverse features to the description logic $\mathcal{CFD}_{nc}^{\forall}$. In *PRICAI 2014*, 587–599.

Zaniolo, C. 1983. The database language GEM. In *ACM SIGMOD Int. Conf. on Management of Data*, 207–218.