On Partial Features in the $\mathcal{DLF}$ Dialects of Description Logic with Inverse Features

David Toman and Grant Weddell

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

Abstract. The $\mathcal{DLF}$ dialects of description logic are fragments of first order logic with underlying signatures based on unary predicate symbols and on unary function symbols interpreted as total functions. In earlier work, we have shown how computational properties of logical consequence for dialects of this family without inverse features are preserved when unary function symbols are interpreted instead as partial functions, and when a "$\exists f$" concept constructor for feature value existence is added that can be used to enforce function totality. In this paper, we resolve a number of open problems mentioned in this earlier work that concerns $\mathcal{DLF}$ dialects with inverse features. Our main result shows how the dialect $\mathcal{CFDI}_{nc}^{\forall-}$ can be extended with a limited form of conjunction on left-hand-sides of inclusion dependencies that enables a straightforward simulation of partial functions together with the $\exists f$ concept constructor.

1 Introduction

The $\mathcal{DLF}$ dialects of *description logics* (DLs) are fragments of first order logic with underlying signatures that replace binary predicate symbols, called *roles*, with unary function symbols, interpreted as total functions, called *features*. In earlier work [7], we have shown how computational properties of logical consequence for dialects of this family are preserved when unary function symbols are now interpreted as partial functions, where equality is based on the so-called strict interpretation of undefined values, and when an "$\exists f$" concept constructor is added for identifying subsets of a domain for which f-values must exist. Note that the latter yields an ability to define cases in which partial functions become total functions, such as to say that *every employee has a salary*, or to define cases in which partial functions are not meaningful, such as to say that *departments do not have a salary*.

In this paper, we resolve a number of open problems mentioned in this earlier work that concern $\mathcal{DLF}$ dialects *with inverse features*, among which is the more recent $\mathcal{CFDI}_{nc}^{\forall-}$ dialect in the $\mathcal{CFD}$ sub-family [6]. Members of this sub-family are distinguished by having PTIME complexity for logical consequence. One of our results shows how $\mathcal{CFDI}_{nc}^{\forall-}$ can be extended with a form of conjunction on left-hand-sides of inclusion dependencies while still retaining PTIME complexity for logical consequence. We show how the added expressiveness enables a straightforward simulation of partial functions and the $\exists f$ concept constructor.

The paper is organized as follows. In the next section, we provide the necessary background and definitions: an overview of the syntax and semantics of the $\mathcal{DLF}$ dialects, and of their general extension to partial functions as we have proposed in [7]. In Section 3, we introduce the dialects $\mathcal{DLFI}$ and $\mathcal{DLFDI}$, and how, by following the same reduction introduced in [7], each can simulate their partial extensions in a straightforward manner. The results mentioned above regarding the dialect $\mathcal{CFDI}_{nc}^{\forall-}$ are then presented in Section 4. Mainly, this entails introducing an extension we call $\mathcal{CFDI}_{kc}^{\forall-}$ that admits a limited use of conjunction on the left-hand-sides of inclusion dependencies. We then show how $\mathcal{CFDI}_{kc}^{\forall-}$ can also be used to simulate its partial extension in a similar fashion to how we did this for $\mathcal{DLFI}$ and $\mathcal{DLFDI}$ in Section 3.

2 Background and Definitions

We begin with a review of the basic definitions for member dialects of the $\mathcal{DLF}$ family in which features replace roles and are interpreted as total functions. Following this, we introduce the necessary modifications that enable features to be interpreted instead as partial functions.

Definition 1 (Feature-based DLs) Let F and PC be sets of feature names and primitive concept names, respectively. A *path expression* is defined by the grammar "$\mathsf{Pf} ::= f.\,\mathsf{Pf} \mid id$" for $f \in \mathsf{F}$.[1] We define derived *concept descriptions* by the grammar on the left-hand-side of Figure 1.

An *inclusion dependency* $\mathcal{C}$ is an expression of the form $C_1 \sqsubseteq C_2$. A *terminology* (TBox) $\mathcal{T}$ consists of a finite set of inclusion dependencies. A *posed question* $\mathcal{Q}$ is a single inclusion dependency.

The *semantics* of expressions is defined with respect to a structure $\mathcal{I} = (\triangle, \cdot^{\mathcal{I}})$, where $\triangle$ is a domain of "objects" and $\cdot^{\mathcal{I}}$ an interpretation function that fixes the interpretations of primitive concepts A to be subsets of $\triangle$ and primitive features f to be total functions $f^{\mathcal{I}} : \triangle \to \triangle$. The interpretation is extended to path expressions, $id^{\mathcal{I}} = \lambda x.x$, $(f.\,\mathsf{Pf})^{\mathcal{I}} = \mathsf{Pf}^{\mathcal{I}} \circ f^{\mathcal{I}}$ and derived concept descriptions C as defined in the center column of Figure 1. An interpretation $\mathcal{I}$ *satisfies an inclusion dependency* $C_1 \sqsubseteq C_2$ if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ and is a *model of* $\mathcal{T}$ ($\mathcal{I} \models \mathcal{T}$) if it satisfies all inclusion dependencies in $\mathcal{T}$. The *logical implication problem* asks if $\mathcal{T} \models \mathcal{Q}$ holds, that is, if $\mathcal{Q}$ is satisfied in all models of $\mathcal{T}$. □

Here, we do *not* consider so-called ABoxes, that is, sets of assertions about membership of individuals in descriptions, nor do we consider the associated problem of knowledge base consistency. Note, however, that such issues can be reduced to logical implication problems involving posed questions that utilize value restrictions and equational *same-as* descriptions [5].

The logical implication problem for TBoxes and posed questions characterized so far, that allow arbitrary concepts in inclusion dependencies, is not decidable for a variety of reasons. For example, see [4] for one case involving arbitrary

[1] We also simplify this notation by allowing a syntactic composition "$\mathsf{Pf}_1\,.\,\mathsf{Pf}_2$" that stands for their concatenation.

Syntax	Semantics: Defn of "$\cdot^{\mathcal{I}}$"	
$C ::= A$	$A^{\mathcal{I}} \subseteq \triangle$	(primitive concept; $A \in \mathsf{PC}$)
$\mid\ C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$	(conjunction)
$\mid\ C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$	(disjunction)
$\mid\ \neg C$	$\triangle \setminus C^{\mathcal{I}}$	(negation)
$\mid\ \forall\, \mathsf{Pf}\,.C$	$\{x : \mathsf{Pf}^{\mathcal{I}}(x) \in C^{\mathcal{I}}\}$	(value restriction)
$\mid\ \exists f^{-1}$	$\{f^{\mathcal{I}}(x) : x \in \triangle\}$	(inverse feature)
$\mid\ C : \mathsf{Pf}_1, ..., \mathsf{Pf}_k \rightarrow \mathsf{Pf}_0$	$\{x : \forall\, y \in C^{\mathcal{I}}.\, \bigwedge_{i=1}^{k} \mathsf{Pf}_i^{\mathcal{I}}(x) = \mathsf{Pf}_i^{\mathcal{I}}(y) \rightarrow \mathsf{Pf}_0^{\mathcal{I}}(x) = \mathsf{Pf}_0^{\mathcal{I}}(y)\}$	(PFD)
$\mid\ \top$	$\triangle$	(top)
$\mid\ \bot$	$\emptyset$	(bottom)
$\mid\ (\mathsf{Pf}_1 = \mathsf{Pf}_2)$	$\{x : \mathsf{Pf}_1^{\mathcal{I}}(x) = \mathsf{Pf}_2^{\mathcal{I}}(x)\}$	(same-as)

Fig. 1. Syntax and Semantics of $\mathcal{DLFD}/\mathcal{CFD}$ Concepts.

PFDs and ABoxes encoded in the above manner. However, restrictions on occurrences of concept constructors has led to a number of decidable fragments that range from light-weight to expressive dialects of feature-based DLs. The restrictions that obtain $\mathcal{DLFI}$, $\mathcal{DLFDI}$, $\mathcal{CFDI}_{nc}^{\forall -}$ and $\mathcal{CFDI}_{kc}^{\forall -}$, the focus of our attention, are given in Section 3 for the first two cases and in Section 4 for the last two cases.

The two definitions that follow introduce the necessary modifications to our characterization of feature-based DLs to accommodate features interpreted as partial functions. To refer to such modifications, we follow a notational convention of qualifying particular dialects with the word "*partial*" whenever we intend such modifications to apply, as in *partial*$-\mathcal{DLFDI}$ for example.

Definition 2 (Partial Features and Existential Restrictions) The syntax of feature-based DLs is extended with an additional concept constructor of the form "$\exists f$", called an *existential restriction*. Semantics is given as follows:

1. Features $f \in F$ are now interpreted as *partial* functions on $\triangle$ (i.e., the result can be *undefined* for some elements of $\triangle$); and
2. The $\exists f$ concept constructor is interpreted as $\{x : \exists y \in \triangle . f^{\mathcal{I}}(x) = y\}$.

In this setting, a path function Pf naturally denotes a partial function resulting from the composition of partial functions. We also adopt the *strict* interpretation of undefined values. This means that equality holds only when both of its arguments are defined and denote the same object, and that set membership ($\in$) requires only defined values to be members of its right hand side argument. □

Observe that features are still *functional*, and that there is therefore no need for a qualified existential restriction of the form "$\exists f.C$", with semantics given as follows:

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

Indeed, such a restriction can be simulated by assuming "$\exists f.C$" to be shorthand for "$(\exists f \sqcap \forall f.C)$". Assuming this, we now write "$(\exists \mathsf{Pf})$" in the following as

shorthand for "$(\exists f_1 \sqcap \forall f_1.(\exists f_2 \sqcap \forall f_2.(\ldots(\exists f_k)\ldots)))$". All that remains for our modifications is to revise the semantics of the PFD constructor to account for the presence of partial features. We adopt the minimum necessary revision needed for recognizing when one violates a *PFD inclusion dependency* of the form

$$\text{“}C_1 \sqsubseteq C_2 : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}_0\text{”}\,.$$

At the least, this should happen when: (1) 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 (2) $\mathsf{Pf}_i^{\mathcal{I}}(e_1) = \mathsf{Pf}_i^{\mathcal{I}}(e_2)$ holds only for $i > 0$. This yields the following modification to the interpretation of PFDs in the presence of partial features that we now adopt:

$$(C : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}_0)^{\mathcal{I}} = \{x : \forall y.y \in C^{\mathcal{I}} \wedge x \in (\exists \mathsf{Pf}_0)^{\mathcal{I}} \wedge y \in (\exists \mathsf{Pf}_0)^{\mathcal{I}} \wedge \bigwedge_{i=1}^{k}(x \in (\exists \mathsf{Pf}_i)^{\mathcal{I}} \wedge y \in (\exists \mathsf{Pf}_i)^{\mathcal{I}} \wedge \mathsf{Pf}_i^{\mathcal{I}}(x) = \mathsf{Pf}_i^{\mathcal{I}}(y)) \rightarrow \mathsf{Pf}_0^{\mathcal{I}}(x) = \mathsf{Pf}_0^{\mathcal{I}}(y)\}.$$

Observe that this definition coincides with the original semantics of the PFD constructor given in Figure 1 when features are interpreted as total functions.

The following illustrates the use of existential restrictions and PFDs in $\mathcal{DLF}$ dialects with partial features. This includes two examples of constraints mentioned informally in our introductory comments together with two further examples showing how PFDs can be used to capture keys and functional dependencies. Note that all can be captured in any of the dialects mentioned below:

1. $EMP \sqsubseteq \exists salary$ (*every employee has a salary*);
2. $DEPT \sqsubseteq \neg\exists salary$ (*departments do not have a salary*);
3. $DEPT \sqsubseteq DEPT : manager \rightarrow id$ (*no two departments have the same manager*); and
4. $EMP \sqsubseteq EMP : paygrade \rightarrow salary$ (*employee pay grades determine salaries*).

3 Expressive Logics with Inverses in the $\mathcal{DLF}$ Family

We now introduce a pair of expressive feature-based DLs and show how logical consequence for their *partial* variants can be simulated in a straightforward fashion. The first is called $\mathcal{DLFI}$ and allows both TBox and posed question dependencies to contain concepts formed from primitive concepts and bottom using the following concept constructors: *conjunction*, *disjunction*, *negation*, *value restriction* and *inverse features*. The second is called $\mathcal{DLFDI}$, and allows, in addition, the PFD concept constructor to appear on the right hand sides of inclusion dependencies. Both logics are a special case of the dialect $\mathcal{DLFAD}$ introduced in [3]. This earlier DL admitted *qualified* inverse features together with restrictions on their use which yielded cases for which the logical implication problem was complete for EXPTIME. One of the restrictions relating to a *coherency* condition on terminologies introduced in [3] is obtained when inverse features are unqualified, as is the case for both $\mathcal{DLFI}$ and $\mathcal{DLFDI}$. Thus, overall restrictions on syntax for these DLs yield expressive Boolean complete dialects with a logical implication problem that is complete for EXPTIME as well. Conversely, extensions, such as allowing PFDs on the left-hand sides of inclusion dependencies, or allowing equational constraints in the posed questions (or equivalently ABoxes), make logical consequence undecidable [4].

We now proceed to demonstrate that partial features can be effectively *simulated* in $\mathcal{DLFI}$ and $\mathcal{DLFDI}$ by introducing an auxiliary primitive concept G that stands for *existing or generated* objects, and by using *value restrictions* to assign membership of objects generated by the $\exists f$ constructor to this concept. All remaining inclusion dependencies are then simply preconditioned by this auxiliary concept.

Formally, let $\mathcal{T}$ be a *partial*$-\mathcal{DLFI}$ TBox in which all inclusion dependencies are of the form $\top \sqsubseteq C$. We define a $\mathcal{DLFI}$ TBox $\mathcal{T}_{\mathcal{DLFI}}$ as

$$\begin{aligned}\mathcal{T}_{\mathcal{DLFI}} = \{&G \sqsubseteq C[\exists f \mapsto \forall f.G, \text{ for all } f \in F] \mid \top \sqsubseteq C \in \mathcal{T}\} \\ &\cup \{\forall f.G \sqsubseteq G \mid f \in F\},\end{aligned}$$

where G is a primitive concept not occurring in $\mathcal{T}$. Note that the substitution $[\exists f \mapsto \forall f.G, \text{ for all } f \in F]$ is applied simultaneously to *all* occurrences of the $\exists f$ constructor in the concept C.

Theorem 3 Let $\mathcal{T}$ be a *partial*$-\mathcal{DLFI}$ TBox in which all inclusion dependencies are of the form $\top \sqsubseteq C$. Then

$$\mathcal{T} \models \top \sqsubseteq C \text{ if and only if } \mathcal{T}_{\mathcal{DLFI}} \models G \sqsubseteq C[\exists f \mapsto \forall f.G, \text{ for all } f \in F],$$

for G a fresh primitive concept.

Proof (sketch): For any $\mathcal{I}$ where $\mathcal{I} \models \mathcal{T}_{\mathcal{DLFI}}$, we can define an interpretation $\mathcal{J} = (G^{\mathcal{I}}, \cdot^{\mathcal{I}}_{|G^{\mathcal{I}}})$. It is easy to verify that $\mathcal{J} \models \mathcal{T}$ and also that $\mathcal{J} \models \top \sqsubseteq C$ since $\mathcal{I} \models G \sqsubseteq C[\exists f \mapsto \forall f.G, \text{ for all } f \in F]$.

For the other direction, we need to extend a model $\mathcal{J}$ of $\mathcal{T}$ to a model $\mathcal{I}$ of $\mathcal{T}_{\mathcal{DLFI}}$ by setting $G^{\mathcal{I}} = \triangle^{\mathcal{J}}$ and by adding *missing* features connecting $\mathcal{I}$ to complete F^* trees with all nodes in $(\neg G)^{\mathcal{I}}$. This way, either $\mathcal{I}$ coincides with $\mathcal{J}$ or satisfies dependencies in $\mathcal{T}_{\mathcal{DLFI}}$ and $G \sqsubseteq C[\exists f \mapsto \forall f.G, \text{ for all } f \in F]$ vacuously. □

To extend this construction to the full *partial*$-\mathcal{DLFDI}$ logic, it is sufficient to *encode* the path function existence preconditions in terms of the auxiliary concept G as follows: if $A \sqsubseteq B : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}_0 \in \mathcal{T}$ then

$$A \sqcap (\prod_{i=0}^{k} \forall\, \mathsf{Pf}_i\, .G) \sqsubseteq B \sqcap (\prod_{i=0}^{k} \forall\, \mathsf{Pf}_i\, .G) : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}_0 \qquad (1)$$

is added to $\mathcal{T}_{\mathcal{DLFDI}}$. Here, we are assuming w.l.o.g. that A and B are primitive concept names ($\mathcal{DLFDI}$ allows one to give such names to complex concepts).

Theorem 4 Let $\mathcal{T}$ be a *partial*$-\mathcal{DLFDI}$ TBox in which all inclusion dependencies are of the form $\top \sqsubseteq C$ or $A \sqsubseteq B : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}_0$. Then

$\mathcal{T} \models \top \sqsubseteq C$ if and only if $\mathcal{T}_{\mathcal{DLFDI}} \models G \sqsubseteq C[\exists f \mapsto \forall f.G, \text{ for all } f \in F]$, and
$\mathcal{T} \models A \sqsubseteq B : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \to \mathsf{Pf}$ if and only if $\mathcal{T}_{\mathcal{DLFDI}} \models (1)$,

for G a fresh primitive concept.

Proof (sketch): The claim follows by observing that (1) captures properly the semantics of PFDs and then by appealing to Theorem 3. □

Corollary 5 Logical implication is EXPTIME-complete for *partial*$-\mathcal{DLFI}$ and for *partial*$-\mathcal{DLFDI}$. □

Similar results can be obtained for other members of the $\mathcal{DLFI}$ family.

4 Tractable Logics with Inverses in the $\mathcal{CFD}$ Sub-Family

We now introduce a new member of the $\mathcal{DLFI}$ sub-facility of DLs called $\mathcal{CFDI}^{\forall-}_{kc}$, and show how the added expressiveness of partial functions and existential restriction can be simulated in $\mathcal{CFDI}^{\forall-}_{kc}$ in the same general way as was done above for the expressive cases. (Recall that members of this sub-family have PTIME complexity for determining logical consequence.) $\mathcal{CFDI}^{\forall-}_{kc}$ is an extension of the $\mathcal{CFD}$ dialect $\mathcal{CFDI}^{\forall-}_{nc}$ in [6], the first member of this sub-family to allow the use of feature inversion. The extension adds to this earlier dialect a capability for a limited use of conjunction in left-hand-sides of inclusion dependencies which, among other things, yields the ability to simulate partial features in *partial*$-\mathcal{CFDI}^{\forall-}_{kc}$.

Our new DL, like all members of the $\mathcal{CFD}$ family, allows the use of an ABox (optionally captured by using *same-as* in left-hand-sides of posed questions) and therefore requires PFDs to adhere to one of the following two forms to avoid both undecidability [4] and indeed intractability [6]:

$$\begin{array}{rll} \text{PFD} ::= & A : \mathsf{Pf}_1, \ldots, \mathsf{Pf} \,.\, \mathsf{Pf}_i, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf} & (\textit{key}) \\ \mid & A : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_i \,.f, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}_i \,.g & (\textit{functional dependency}) \end{array} \quad (2)$$

With this restriction, introduced in [6], posed questions can contain inclusion dependencies formed from concepts in Figure 1, albeit with a few mild restrictions when tractability in the size of the posed question is required. For simplicity, however, we assume that the concepts in the posed question $\mathcal{Q} = E_1 \sqsubseteq E_2$ adhere to the following grammar:

$$E \ ::= \ A \ \mid \ \bot \ \mid \ E \sqcap E \ \mid \ \forall \mathsf{Pf} \,.E \ \mid \ (\mathsf{Pf}_1 = \mathsf{Pf}_2).$$

More complex posed questions, e.g., ones that contain the PFD constructor [5], can be equivalently expressed in the above grammar (perhaps as a sequence of posed questions).

Inclusion dependencies $C \sqsubseteq D$ in a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox are respectively given by the following grammars:

$$\begin{array}{l} C ::= A \ \mid \ \forall f.A \ \mid \ A_1 \sqcap A_2 \\ D ::= A \ \mid \ \bot \ \mid \ \forall f.A \ \mid \ \exists f^{-1} \ \mid \ \text{PFD} \end{array}$$

For technical reasons we assume, w.l.o.g., that either C or D is a primitive concept for the remainder of the paper.

In comparison to $\mathcal{CFDI}^{\forall-}_{nc}$ introduced in [6], this new dialect now allows inclusion dependencies of the form "$(A_1 \sqcap A_2) \sqsubseteq A_3$". In general, unrestricted use of conjunction on the left-hand side of inclusion dependencies leads to EXPTIME-completeness of the associated reasoning problems, and so it is necessary to somehow restrict this use of conjunction. The intuition behind the restriction

we adopt below is to limit the number of *conjuncts* that need to be considered on the left-hand sides of inclusion dependencies. Unfortunately, this property is not a simple syntactic restriction, e.g., on the form of inclusion dependencies in a TBox. Indeed, it is easy to see that allowing inclusion dependencies in $\mathcal{CFDI}^{\forall-}_{kc}$ of the form above is equivalent to allowing arbitrary conjunctions (due to inference). Hence, a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox is instead given as follows:

Definition 6 (Restricted Conjunction) Let $\mathcal{T}$ be a TBox with inclusion dependencies satisfying the above grammar rules, and let $k > 0$. We say that $\mathcal{T}$ 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 $\{A_1, \ldots, A_n\} \cup \{B\}$, 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 $\{A_1, \ldots, A_n\}$. □

The following Lemma shows that verifying whether an TBox is an $\mathcal{CFDI}^{\forall-}_{kc}$ TBox can be verified in PTIME (for a fixed k):

Lemma 7 Let $\mathcal{T}$ be a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox and $k > 0$. Then $\mathcal{T}$ is not a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox if there are primitive concepts $A_1, \ldots, A_{k+1}$ and B such that $\mathcal{T} \models (A_1 \sqcap \ldots \sqcap A_{k+1}) \sqsubseteq B$ for which the condition in Definition 6 fails.

Proof (sketch): Consider an inclusion dependency $(A_1 \sqcap \ldots \sqcap A_n) \sqsubseteq B$ where $n > 2$ is the smallest value such that the dependency is a logical consequence of $\mathcal{T}$ and such that there no $(A_{i_1} \sqcap \cdots \sqcap A_{i_k}) \sqsubseteq B$ that is also a logical consequence of $\mathcal{T}$. Observe that there must be a linear input resolution proof (of the first-order translation of the problem) of $B(x)$ from $\mathcal{T} \cup \{A_1(x), \ldots, A_n(x)\}$ [2]. Since the resolution steps with $A_1(x), \ldots, A_n(x)\}$ can be performed last and they all must be present, there is a clause $\neg A_1(x), \ldots, \neg A_n(x)$ in the proof. However, since all input clauses to the problem are binary or ternary and since the proof starts with $\neg B(x)$, there must be a one-shorter clause preceding this clause in this proof. A contradiction. □

We now consider TBox and concept satisfiability in $\mathcal{CFDI}^{\forall-}_{kc}$, and show an PTIME algorithm by extending the approach for $\mathcal{CFDI}^{\forall-}_{nc}$ in [6].

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:

Definition 8 (TBox Closure) Let $\mathcal{T}$ be a $\mathcal{CFDI}^{\forall-}_{kc}$ TBox in normal form. We define $\mathsf{Clos}(\mathcal{T})$ to be the least set of subsumptions such that:

1. $D \sqsubseteq D$;
2. $D \sqsubseteq E$ and $E \sqsubseteq F$ then $D \sqsubseteq F$;
3. $D \sqsubseteq E$ and $F \subseteq E$, $F \neq \emptyset$ then $D \sqsubseteq F$;
4. $D \sqsubseteq E$ and $D \sqsubseteq F$ then $D \sqsubseteq E + F$ (for all $E + F \subseteq E \cup F$);
5. $A \sqsubseteq B \in \mathcal{T}$ and $A \subseteq D$ then $D \sqsubseteq B$;
6. $\forall f.\bot \sqsubseteq \bot$ and $\bot \sqsubseteq \forall f.\bot$;
7. $D \sqsubseteq E$ then $\forall f.D \sqsubseteq \forall f.E$;

8. $D \sqsubseteq \exists f^{-1}$ and $\forall f.D \sqsubseteq \forall f.E$ then $D \sqsubseteq E$

where (1) D, E, and F are sets of primitive concepts of size at most k, or sets of value restrictions with respect to a particular feature applied to such sets of primitive concepts, where (2) $E+F$ is a subset of size between 1 and k of $E \cup F$, and where (3) A and B are concepts allowed on the left and right-hand sides of subsumptions in $\mathcal{T}$. We also conflate sets of concepts and their conjunctions and value restriction applied to a conjunction with a conjunction of individual value restrictions. □

Note that $\mathsf{Clos}(\mathcal{T})$ is polynomial in $|\mathcal{T}|$ (and exponential in k). It is also easy to verify that each inclusion added to $\mathsf{Clos}(\mathcal{T})$ by the inferences (1-4) in Definition 8 is logically implied by $\mathcal{T}$.

Theorem 9 (Primitive Concept Satisfiability) Let $\mathcal{T}$ be a $\mathcal{CFDI}_{kc}^{\forall-}$ TBox in normal form and A a primitive concept description. Then A is satisfiable with respect to T if and only if $A \sqsubseteq \bot \notin \mathsf{Clos}(\mathcal{T})$.

Proof (sketch): Given $\mathsf{Clos}(\mathcal{T})$, an object o, and a primitive concept A, we define the following family of subsets of PC indexed by paths of features (and their inverses), starting from o, as follows:

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

We say that S_x is *defined* if it conforms to one of the three above cases, and that it is *consistent* if $\bot \notin S_x$.

It is easy to see that

1. If $S_{f(x)}$ is not consistent, then S_x is not consistent.
2. If $S_{f^-(x)}$ is defined and not consistent, then S_x is not consistent.

We build a model of $\mathcal{T}$ in which $o \in A^{\mathcal{I}}$ for some $o \in \triangle$ as follows:

- $\triangle = \{x \mid S_x$ is defined$\}$;
- $f^{\mathcal{I}} = \{(x, f(x)) \mid S_{f(x)}$ is defined$\} \cup \{(f^-(x), x) \mid S_{f^-(x)}$ is defined$\}$; and
- $A^{\mathcal{I}} = \{x \mid S_x$ is defined, $A \in S_x\}$.

Observe all defined sets S_x must be consistent. Otherwise, A ($\in S_0$) must be inconsistent, implying in turn that $A \sqsubseteq \bot \in \mathsf{Clos}(\mathcal{T})$, a contradiction. Hence, $\mathcal{I} = (\triangle, .^{\mathcal{I}})$ is a model of $\mathcal{T}$ (it satisfies all dependencies in $\mathsf{Clos}(\mathcal{T})$) such that $o \in A^{\mathcal{I}}$. □

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

$\mathsf{E}(\mathsf{Pf}_1, \mathsf{Pf}_2) \rightarrow \mathsf{E}(\mathsf{Pf}_2, \mathsf{Pf}_1)$
$\mathsf{E}(\mathsf{Pf}_1, \mathsf{Pf}_2) \wedge \mathsf{E}(\mathsf{Pf}_2, \mathsf{Pf}_3) \rightarrow \mathsf{E}(\mathsf{Pf}_1, \mathsf{Pf}_3)$
$\mathsf{E}(\mathsf{Pf}_1, \mathsf{Pf}_2) \rightarrow \mathsf{E}(\mathsf{Pf}_1 . f, \mathsf{Pf}_2 . f)$, for $\{\mathsf{Pf}_1 . f, \mathsf{Pf}_2 . f\} \subseteq \mathsf{PF}(\mathcal{T}, \mathcal{Q})$
$\mathsf{E}(\mathsf{Pf}_1, \mathsf{Pf}_2) \wedge \mathsf{C}_C(\mathsf{Pf}_1) \rightarrow \mathsf{C}_C(\mathsf{Pf}_2)$

$\mathsf{C}_{C_1 \sqcap C_2}(\mathsf{Pf}) \rightarrow \mathsf{C}_{C_1}(\mathsf{Pf})$ and $\mathsf{C}_{C_1 \sqcap C_2}(\mathsf{Pf}) \rightarrow \mathsf{C}_{C_2}(\mathsf{Pf})$
$\mathsf{C}_{\forall \mathsf{Pf}' . C}(\mathsf{Pf}) \rightarrow \mathsf{C}_C(\mathsf{Pf} . \mathsf{Pf}')$ for $\mathsf{Pf} . \mathsf{Pf}' \in \mathsf{PF}(\mathcal{T}, \mathcal{Q})$
$\mathsf{C}_{(\mathsf{Pf}_1 = \mathsf{Pf}_2)}(\mathsf{Pf}) \rightarrow \mathsf{E}(\mathsf{Pf} . \mathsf{Pf}_1, \mathsf{Pf} . \mathsf{Pf}_2)$
$\mathsf{C}_{C:\mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}_0}(\mathsf{Pf}) \wedge \mathsf{C}_C(\mathsf{Pf}') \wedge (\bigwedge_{0<i\leq k} \mathsf{E}(\mathsf{Pf} . \mathsf{Pf}_i, \mathsf{Pf}' . \mathsf{Pf}_i)) \rightarrow \mathsf{E}(\mathsf{Pf} . \mathsf{Pf}_0, \mathsf{Pf}' . \mathsf{Pf}_0))$

$\mathsf{C}_{A_1}(\mathsf{Pf}) \wedge \ldots \wedge \mathsf{C}_{A_k}(\mathsf{Pf}) \rightarrow \mathsf{C}_D(\mathsf{Pf})$ for all $(A_1 \sqcap \ldots \sqcap A_k \sqsubseteq D) \in \mathcal{T}$
$\mathsf{C}_A(\mathsf{Pf} . f) \rightarrow \mathsf{C}_D(\mathsf{Pf})$ for all $(\forall f.A \sqsubseteq D) \in \mathcal{T}$

Fig. 2. Expansion Rules.

The above theorem can be used to check satisfiability of complex (non-PFD) concepts; e.g., satisfiability of $\forall \mathsf{Pf} . B$ w.r.t. $\mathcal{T}$ can be tested by checking satisfiability of a new primitive concept A w.r.t. (the normalized version of) $\mathcal{T} \cup \{A \sqsubseteq \forall \mathsf{Pf} . B\}$.

The theorem also provides a technique for checking satisfiability of finite conjunctions of primitive concepts with respect to $\mathcal{T}$:

Corollary 10 Let $\mathcal{T}$ be a $\mathcal{CFDI}_{kc}^{\forall -}$ TBox in normal form and $A_1, \ldots, A_k$ primitive concepts. Then $A_1 \sqcap \ldots \sqcap A_k$ is satisfiable with respect to $\mathcal{T}$ if and only if A is satisfiable with respect to $\mathcal{T} \cup \{A \sqsubseteq A_1, \ldots, A \sqsubseteq A_k\}$, for A a fresh primitive concept (note that $\mathcal{T} \cup \{A \sqsubseteq A_1, \ldots, A \sqsubseteq A_k\}$ is a $\mathcal{CFDI}_{kc}^{\forall -}$ TBox whenever $\mathcal{T}$ is). □

Logical Implication and Knowledge Base Consistency

Allowing *inverse features* affects how PFDs interact with a posed question. In particular, PFDs in which all path functions have a common prefix may apply to (pairs of) anonymous individuals mandated by the existence of anonymous inverse features. In general, to enforce PFDs with respect to a posed question *while avoiding any need to explicitly create anonymous predecessor objects*, we add additional logically implied PFDs to a given TBox as follows:

Definition 11 (PFD Enrichment for Inverses) Let $\mathcal{T}$ be a $\mathcal{CFDI}_{kc}^{\forall -}$ TBox in normal form such that $A \sqsubseteq B : f.\mathsf{Pf}_1, \ldots, f.\mathsf{Pf}_k \rightarrow f.\mathsf{Pf} \in \mathcal{T}$ ($A \sqsubseteq B : f.\mathsf{Pf}_1, \ldots, f.\mathsf{Pf}_k \rightarrow id \in \mathcal{T}$) where $\mathsf{Pf}_i \neq id$ for all $1 \leq i \leq k$. Then we require that $A \sqsubseteq \forall f.A'$, $B \sqsubseteq \forall f.B'$, and $A' \sqsubseteq B' : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow \mathsf{Pf}$ ($A' \sqsubseteq B' : \mathsf{Pf}_1, \ldots, \mathsf{Pf}_k \rightarrow id$), where A' and B' are fresh primitive concepts, are also in $\mathcal{T}$. (For further details on this, see [1].) □

With these restrictions we can now show that the logical implication problem for $\mathcal{CFDI}_{kc}^{\forall -}$ is in PTIME. Our proof is based on encoding a given problem as a collection of Horn clauses. The reduction introduces terms that correspond to path expressions, and relies on the fact that the number of required terms is polynomial in the size of the problem itself.

Definition 12 (Expansion Rules) Let $\mathcal{T}$ and $\mathcal{Q}$ be a *partial*$-\mathcal{CFD}$ terminology and a posed question, respectively. We write $\mathsf{CON}(\mathcal{T},\mathcal{Q})$ to denote the set of all subconcepts appearing in $\mathcal{T}$ and $\mathcal{Q}$, define $\mathsf{PF}(\mathcal{T},\mathcal{Q})$ to be the set

$$\{\mathsf{Pf}\,.\,\mathsf{Pf}' \mid \mathsf{Pf} \textit{ is a prefix of a path expression in } \mathcal{Q} \textit{ and } \mathsf{Pf}' \textit{ is a feature occurring in } \mathcal{T} \textit{ or } id\},$$

write C_C to denote unary predicates for $C \in \mathsf{CON}(\mathcal{T},\mathcal{Q})$, and introduce a unary predicate D and a binary predicate E, with all predicates ranging over the universe $\mathsf{PF}(\mathcal{T},\mathcal{Q})$. The *expansion rules* for a given terminology $\mathcal{T}$, denoted $\mathsf{R}(\mathcal{T})$, are defined in Figure 2. To deal with the possibility of inconsistency, we add the following rule that can be applied *after the rules in Figure 2 are are exhaustively applied.*[2]

> If $\mathsf{C}_{A_1}(\mathsf{Pf}),\ldots,\mathsf{C}_{A_k}(\mathsf{Pf}) \in \mathsf{R}(\mathcal{T})$ for some $\mathsf{Pf} \in \mathsf{PF}(\mathcal{T},\mathcal{Q})$ and $A_1 \sqcap \ldots \sqcap A_k$ is not consistent in $\mathcal{T}$ then add $\mathsf{C}_\bot(\mathsf{Pf})$ to $\mathsf{R}(\mathcal{T})$.

A *goal* for each concept E is a set of ground assertions defined as follows:

$$\mathsf{G}_E = \begin{cases} \{\mathsf{C}_A(id)\} & \text{for } E = A; \\ \{\mathsf{C}_\bot(id)\} & \text{for } E = \bot; \\ \{\mathsf{E}(\mathsf{Pf}_1,\mathsf{Pf}_2)\} & \text{for } E = (\mathsf{Pf}_1 = \mathsf{Pf}_2); \\ \mathsf{G}_{E_1} \cup \mathsf{G}_{E_2} & \text{for } E = E_1 \sqcap E_2; \text{ and} \\ \{\mathsf{C}_C(\mathsf{Pf}'\,.\,\mathsf{Pf}) \mid \mathsf{C}_C(\mathsf{Pf}) \in \mathsf{G}_{E'}\} & \\ \quad \cup \{\mathsf{E}(\mathsf{Pf}'\,.\,\mathsf{Pf}_1, \mathsf{Pf}'\,.\,\mathsf{Pf}_2) \mid \mathsf{E}(\mathsf{Pf}_1,\mathsf{Pf}_2) \in \mathsf{G}_{E'}\} & \text{for } E = \forall\,\mathsf{Pf}'\,.E'. \end{cases}$$

Given two concept descriptions E_1 and E_2, we say that

$$\mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\} \models \mathsf{G}_{E_2}$$

if $\mathsf{G}_{E_2} \subseteq M$ for every minimal ground model M of $\mathsf{R}(\mathcal{T})$ over $\mathsf{PF}(\mathcal{T},\mathcal{Q})$ that contains $\mathsf{C}_{E_1}(id)$ and $\mathsf{D}(id)$. □

Intuitively, $\mathsf{PF}(\mathcal{T},\mathcal{Q})$ represents a finite graph of objects, predicates $\mathsf{E}(\mathsf{Pf}_1,\mathsf{Pf}_2)$ express equality of the objects at the end of paths Pf_1 and Pf_2, and predicates $\mathsf{C}_{C'}(\mathsf{Pf})$ express that the object at the end of path Pf is in the interpretation of concept C'. Note that the expansion rules do not need to take inverses into account due to Definition 11.

Logical implication for $\mathcal{CFDI}_{kc}^{\forall-}$ TBoxes $\mathcal{T}$ and posed questions $\mathcal{Q}$ can now be solved as follows:

Theorem 13 Let $\mathcal{T}$ be a terminology and $\mathcal{Q}$ a posed question of the form $E_1 \sqsubseteq E_2$ in $\mathcal{CFDI}_{kc}^{\forall-}$. Then

$$\mathcal{T} \models \mathcal{Q} \text{ iff } \mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\} \models \mathsf{G}_{E_2} \text{ or } \mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\} \models \mathsf{G}_{\forall\,\mathsf{Pf}\,.\,\bot}(id) \text{ for some } \mathsf{Pf} \in \mathsf{PF}(\mathcal{T},\mathcal{Q}).$$

Proof (sketch): If $\mathsf{C}_\bot(\mathsf{Pf})$ for $\mathsf{Pf} \in \mathsf{PF}(\mathcal{T},\mathcal{Q})$ appear in M, where M is the least model of $\mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\}$ $\mathsf{R}(\mathcal{T})$, then the concept E_1 is unsatisfiable w.r.t. $\mathcal{T}$

[2] Observe that this is where we rely on the our previous PTIME result concerning the satisfiability of finite conjunctions of primitive concepts.

since only implied facts appear in M, and therefore the subsumption holds for any E_1 and $\mathcal{T}$.

Otherwise, if $\mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\} \not\models \mathsf{G}_{E_1}$, then there must be a model M of $\mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\}$ such that $G \notin M$ for some $G \in \mathsf{G}_{E_1}$. We construct an interpretation $\mathcal{I}_M$ such that $\mathcal{I}_M \models \mathcal{T}$ but $\mathcal{I}_M \not\models \mathcal{Q}$. The interpretation $\mathcal{I}_M$ contains an object o for each equivalence class defined on the set $\mathsf{PF}(\mathcal{T}, \mathcal{Q})$ by the interpretation of E. The class membership of these objects is determined by the membership of the corresponding path in the interpretations of the C_C predicates in M. Note that, due to the syntactic restriction imposed on PFDs, this is sufficient to satisfy all PFDs in $\mathcal{T}$ since any precondition or a non-trivial consequence of a PFD can only manifest on some path belonging to $\mathsf{PF}(\mathcal{T}, \mathcal{Q})$ and beginning at the distinguished object o. To complete the construction of $\mathcal{I}_M$, we simply attach a unique complete tree F^* labeled as in Theorem 9 to each leaf node (i.e., a node that is missing successors). Nodes of these complete trees belong to all primitive descriptions in $\mathcal{I}_M$ and thus satisfy $\mathcal{T}$. Conversely, assume $\mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\} \models \mathsf{G}_{E_1}$ but that $\mathcal{T} \not\models \mathcal{Q}$. Then there must be an interpretation $\mathcal{I}$ and an object $o \in \triangle$ such that $\mathcal{I} \models \mathcal{T}$ and $o \in E_1^{\mathcal{I}} - E_2^{\mathcal{I}}$. Thus, there is a model $M_{\mathcal{I}}$ of $\mathsf{R}(\mathcal{T})$ such that $\mathsf{C}_{E_1}(id) \in M_{\mathcal{I}}$. In this model, the element $id \in \mathsf{PF}(\mathcal{T}, \mathcal{Q})$ serves as the counterpart of the object o and the interpretations of the predicates C_C and E is *extracted* from $\mathcal{I}$ by navigating all (pairs of) path functions in $\mathsf{PF}(\mathcal{T}, \mathcal{Q})$. However, since $o \notin E_2^{\mathcal{I}}$, it must be the case that $M_{\mathcal{I}}$ is a strict subset of the least model of $\mathsf{R}(\mathcal{T}) \cup \{\mathsf{C}_{E_1}(id)\}$; a contradiction. □

Corollary 14 Let $\mathcal{T}$ be a terminology and $\mathcal{Q}$ a posed question in $\mathcal{CFDI}_{kc}^{\forall-}$. Then the implication problem $\mathcal{T} \models \mathcal{Q}$ is complete for PTIME. □

Knowledge base consistency can now be reduced to logical implication as shown in [5].

Logical Consequence in *partial*$-\mathcal{CFDI}_{kc}^{\forall-}$

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.

Definition 15 Let $\mathcal{T}$ be a *partial*$-\mathcal{CFDI}_{kc}^{\forall-}$ TBox. We define a $\mathcal{CFDI}_{(k+1)c}^{\forall-}$ TBox $\mathcal{T}_{\mathcal{CFDI}_{(k+1)c}^{\forall-}}$ associated with $\mathcal{T}$ by initializing with the inclusion dependency $\forall f.G \sqsubseteq G$ and then mapping inclusion dependencies in $\mathcal{T}$ to $\mathcal{T}_{\mathcal{CFDI}_{(k+1)c}^{\forall-}}$ according to the following:

$$\begin{array}{rcll} A_1 \sqsubseteq B & \mapsto & (A_1 \sqcap G) \sqsubseteq B & \\ (A_1 \sqcap A_2) \sqsubseteq B & \mapsto & (A_1 \sqcap A_2) \sqsubseteq A_3, (A_3 \sqcap G) \sqsubseteq B & \text{(where } A_3 \text{ is fresh)} \\ \forall f.A_1 \sqsubseteq B & \mapsto & \forall f.A_1 \sqsubseteq A_2, (A_2 \sqcap G) \sqsubseteq B & \text{(where } A_2 \text{ is fresh)} \\ A_1 \sqsubseteq \exists f & \mapsto & A_1 \sqsubseteq \forall f.G & \end{array}$$

□

Indeed, it is easy to verify that $\mathcal{T}_{\mathcal{CFDI}_{(k+1)c}^{\forall-}}$ is a $\mathcal{CFDI}_{(k+1)c}^{\forall-}$ TBox and that:

Theorem 16 Let $\mathcal{T}$ be a *partial*$-\mathcal{CFDI}_{kc}^{\forall-}$ TBox and $\mathcal{Q}$ a posed question. Then

$$\mathcal{T} \models \mathcal{Q} \text{ if and only if } \mathcal{T}_{\mathcal{CFDI}_{(k+1)c}^{\forall-}} \models \mathcal{Q}[\exists f \mapsto \forall f.G, \text{ for all } f \in F].$$ □

References

1. Jason St. Jacques, David Toman, and Grant E. Weddell. Object-relational queries over $\mathcal{CFDI}_{nc}$ knowledge bases: OBDA for the SQL-Literate. In *Proc. International Joint Conference on Artificial Intelligence, IJCAI*, pages 1258–1264, 2016.
2. John W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, 1987.
3. David Toman and Grant E. Weddell. On the interaction between inverse features and path-functional dependencies in description logics. In *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 603–608, 2005.
4. David Toman and Grant E. Weddell. On keys and functional dependencies as first-class citizens in description logics. *J. Aut. Reasoning*, 40(2-3):117–132, 2008.
5. 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.
6. David Toman and Grant E. Weddell. On adding inverse features to the description logic $\mathcal{CFD}^{\forall}_{nc}$. In *PRICAI 2014: Trends in Artificial Intelligence - 13th Pacific Rim International Conference on Artificial Intelligence, Gold Coast, QLD, Australia, December 1-5, 2014.*, pages 587–599, 2014.
7. David Toman and Grant E. Weddell. On partial features in the *DLF* family of description logics. In Richard Booth and Min-Ling Zhang, editors, *PRICAI 2016: Trends in Artificial Intelligence - 14th Pacific Rim International Conference on Artificial Intelligence, Phuket, Thailand, August 22-26, 2016, Proceedings*, volume 9810 of *Lecture Notes in Computer Science*, pages 529–542. Springer, 2016.