Completing Description Logic Knowledge Bases using Formal Concept Analysis*

Franz Baader[1], Bernhard Ganter[1], Ulrike Sattler[2] and Barış Sertkaya[1]

[1] TU Dresden, Germany
[2] The University of Manchester, UK

Abstract. We propose an approach for extending both the terminological and the assertional part of a Description Logic knowledge base by using information provided by the knowledge base and by a domain expert. The use of techniques from Formal Concept Analysis ensures that, on the one hand, the interaction with the expert is kept to a minimum, and, on the other hand, we can show that the extended knowledge base is complete in a certain, well-defined sense.

1 Introduction

Description Logics (DLs) [1] are employed in various application domains, such as natural language processing, configuration, databases, and bio-medical ontologies, but their most notable success so far is due to the fact that DLs provide the logical underpinning of OWL, the standard ontology language for the semantic web [6]. As a consequence of this standardization, several ontology editors support OWL [9, 10, 8], and ontologies written in OWL are employed in more and more applications. As the size of these ontologies grows, tools that support improving their quality become more important. The tools available until now use DL reasoning to detect inconsistencies and to infer consequences, i.e., implicit knowledge that can be deduced from the explicitly represented knowledge. There are also promising approaches that allow to pinpoint the reasons for inconsistencies and for certain consequences, and that help the ontology engineer to resolve inconsistencies and to remove unwanted consequences [13, 7]. These approaches address the quality dimension of *soundness* of an ontology, both within itself (consistency) and w.r.t. the intended application domain (no unwanted consequences). In the present paper, we are concerned with a different quality dimension: *completeness*. We provide a basis for formally well-founded techniques and tools that support the ontology engineer in checking whether an ontology contains all the relevant information about the application domain, and to extend the ontology appropriately if this is not the case.

A DL knowledge base (nowadays often called ontology) usually consists of two parts, the terminological part (TBox), which defines concepts and also states

* Supported by DFG (GRK 334/3) and the EU (IST-2005-7603 FET project TONES and NoE 507505 Semantic Mining).

additional constraints (so-called general concept inclusions, GCIs) on the interpretation of these concepts, and the assertional part (ABox), which describes individuals and their relationship to each other and to concepts. Given an application domain and a DL knowledge base (KB) describing it, we can ask whether the KB contains all the relevant information about the domain: Are all the relevant constraints that hold between concepts in the domain captured by the TBox? Are all the relevant individuals existing in the domain represented in the ABox?

As an example, consider the OWL ontology for human protein phosphatases that has been described and used in [16]. This ontology was developed based on information from peer-reviewed publications. The human protein phosphatase family has been well characterised experimentally, and detailed knowledge about different classes of such proteins is available. This knowledge is represented in the terminological part of the ontology. Moreover, a large set of human phosphatases has been identified and documented by expert biologists. These are described as individuals in the assertional part of the ontology. One can now ask whether the information about protein phosphatases contained in this ontology is complete. Are all the relationships that hold among the introduced classes of phosphatases captured by the constraints in the TBox, or are there relationships that hold in the domain, but do not follow from the TBox? Are all possible kinds of human protein phosphatases represented by individuals in the ABox, or are there phosphatases that have not yet been included in the ontology or even not yet been identified?

Such questions cannot be answered by an automated tool alone. Clearly, to check whether a given relationship between concepts—which does not follow from the TBox—holds in the domain, one needs to ask a domain expert, and the same is true for questions regarding the existence of individuals not described in the ABox. The rôle of the automated tool is to ensure that the expert is asked as few questions as possible; in particular, she should not be asked trivial questions, i.e., questions that could actually be answered based on the represented knowledge. In the above example, answering a non-trivial question regarding human protein phosphatases may require the biologist to study the relevant literature, query existing protein databases, or even carry out new experiments. Thus, the expert may be prompted to acquire new biological knowledge.

Attribute exploration is an approach developed in Formal Concept Analysis (FCA) [5] that is used to acquire knowledge about an application domain by querying an expert. One of the earliest applications of this approach is described in [15], where the domain is lattice theory, and the goal of the exploration process is to find, on the one hand, all valid relationships between properties of lattices (like being distributive), and, on the other hand, to find counterexamples to all the relationships that do not hold. To answer a query whether a certain relationship holds, the lattice theory expert must either confirm the relationship (by using results from the literature or carrying out a new proof for this fact), or give a counterexample (again, by either finding one in the literature or constructing a new one).

Although this sounds very similar to what is needed in our context, we cannot directly use this approach. The main reason is the open-world semantics of description logic knowledge bases. In DLs, if we cannot deduce from the knowledge base that an individual i is an instance of the concept C, we do not assume that i belongs to $\neg C$. In contrast classical FCA assumes complete knowledge in the sense that, whenever it is not explicitly mentioned that an object o has a property p, we assume that o does not have p. There has been some work on how to extend FCA and attribute exploration from complete knowledge to the case of partial knowledge [11, 4]. However, this work is based on assumptions that are different from ours. In particular, it assumes that the expert cannot answer all queries and, as a consequence, the knowledge obtained after the exploration process may still be incomplete. In contrast, our intention is to complete the KB, i.e., in the end we want to have complete knowledge about these relationships. What may be incomplete is the description of individuals used during the exploration process.

Due to space limitation, we introduce our variant of FCA that can deal with the open-world semantics of DL knowledge bases only briefly. For a more comprehensive description, we refer the reader to the long version of this paper [3] or to the technical report [2], which also contains full proofs of our results.

2 Exploring Partial Contexts

In this section, we extend the classical approach to Formal Concept Analysis (FCA), as described in detail in [5], to the case of individuals (called objects in FCA) that have only a partial description in the sense that, for some properties (called attributes in FCA), it is not known whether they are satisfied by the individual or not. The connection between this approach and the classical approach is explained in [2]. In the following, we assume that we have a finite set M of attributes and a (possibly infinite) set of objects.

Definition 1. *A* partial object description (pod) *is a tuple (A, S) where $A, S \subseteq M$ are such that $A \cap S = \varnothing$. We call such a pod a* full object description (fod) *if $A \cup S = M$. A set of pods is called a* partial context *and a set of fods a* full context*.*

Intuitively, the pod (A, S) says that the object it describes satisfies all attributes from A and does not satisfy any attribute from S. For the attributes not contained in $A \cup S$, nothing is known w.r.t. this object.

Definition 2. *We say that the pod (A', S')* extends *the pod (A, S), and write this as $(A, S) \leq (A', S')$, if $A \subseteq A'$ and $S \subseteq S'$. Similarly, we say that the partial context $\mathcal{K}'$* extends *the partial context $\mathcal{K}$, and write this as $\mathcal{K} \leq \mathcal{K}'$, if every pod in $\mathcal{K}$ is extended by some pod in $\mathcal{K}'$. If $\overline{\mathcal{K}}$ is a full context and $\mathcal{K} \leq \overline{\mathcal{K}}$, then $\overline{\mathcal{K}}$ is called a* realizer *of $\mathcal{K}$. If $(\overline{A}, \overline{S})$ is a fod and $(A, S) \leq (\overline{A}, \overline{S})$, then we also say that $(\overline{A}, \overline{S})$* realizes *$(A, S)$.*

The following notion of implication formalizes the informal notion "relationship between properties" used in the introduction.

Definition 3. *An* implication *is of the form $L \to R$ where $L, R \subseteq M$. This implication is* refuted *by the pod (A, S) if $L \subseteq A$ and $R \cap S \neq \varnothing$. It is* refuted *by the partial context $\mathcal{K}$ if it is refuted by at least one element of $\mathcal{K}$. The set of implications that are not refuted by a given partial context $\mathcal{K}$ is denoted by $Imp(\mathcal{K})$. The set of all fods that do not refute a given set of implications $\mathcal{L}$ is denoted by $Mod(\mathcal{L})$.*

For a set of implications $\mathcal{L}$ and a set $P \subseteq M$, the *implicational closure* of P with respect to $\mathcal{L}$, denoted by $\mathcal{L}(P)$, is the smallest subset Q of M such that

- $P \subseteq Q$, and
- $L \to R \in \mathcal{L}$ and $L \subseteq Q$ imply $R \subseteq Q$.

A set $P \subseteq M$ is called $\mathcal{L}$*-closed* if $\mathcal{L}(P) = P$.

Definition 4. *The implication $L \to R$ is said to* follow *from a set $\mathcal{J}$ of implications if $R \subseteq \mathcal{J}(L)$. The set of implications $\mathcal{J}$ is called* complete *for a set of implications $\mathcal{L}$ if every implication in $\mathcal{L}$ follows from $\mathcal{J}$. It is called* sound *for $\mathcal{L}$ if every implication that follows from $\mathcal{J}$ is contained in $\mathcal{L}$. A set of implications $\mathcal{J}$ is called a* base *for a set of implications $\mathcal{L}$ if it is both sound and complete for $\mathcal{L}$, and no strict subset of $\mathcal{J}$ satisfies this property.*

The following is a trivial fact regarding the connection between partial contexts and the implications they do not refute.

Proposition 1. *For a given set $P \subseteq M$ and a partial context $\mathcal{K}$, $\mathcal{K}(P) := M \setminus \bigcup\{S \mid (A, S) \in \mathcal{K}, P \subseteq A\}$ is the largest subset of M such that $P \to \mathcal{K}(P)$ is not refuted by $\mathcal{K}$.*

Attribute exploration with partial contexts The classical attribute exploration algorithm of FCA assumes that there is a domain expert that can answer questions regarding the validity of implications in the application domain. Accordingly, our approach requires an expert that can decide whether an implication is refuted in the application domain or not. In contrast to existing work on extending FCA to the case of partial knowledge [11, 4], we do *not* assume that the expert has only partial knowledge and thus cannot answer all implication questions.

To be more precise, we consider the following setting. We are given an initial (possibly empty) partial context $\mathcal{K}$, an initially empty set of implications $\mathcal{L}$, and a full context $\overline{\mathcal{K}}$ that is a realizer of $\mathcal{K}$. The expert answers implication questions "$L \to R$?" w.r.t. the full context $\overline{\mathcal{K}}$. More precisely, if the answer is "yes," then $\overline{\mathcal{K}}$ does not refute $L \to R$. The implication $L \to R$ is then added to $\mathcal{L}$. Otherwise, the expert extends the current context $\mathcal{K}$ such that the extended context refutes $L \to R$ and still has $\overline{\mathcal{K}}$ as a realizer. Consequently, the following invariant will be satisfied by $\mathcal{K}, \overline{\mathcal{K}}, \mathcal{L}$: $\mathcal{K} \leq \overline{\mathcal{K}} \subseteq Mod(\mathcal{L})$. Our aim is to enrich $\mathcal{K}$ and $\mathcal{L}$ such that eventually $\mathcal{L}$ is not only sound, but also complete for $Imp(\overline{\mathcal{K}})$, and $\mathcal{K}$ refutes all other implications (i.e., all the implications refuted by $\overline{\mathcal{K}}$). As in the classical case, we want to do this by asking as few as possible questions to the expert.

Definition 5. *Let $\mathcal{L}$ be a set of implications and $\mathcal{K}$ a partial context. An implication is called* undecided *w.r.t. $\mathcal{K}$ and $\mathcal{L}$ if it neither follows from $\mathcal{L}$ nor is refuted by $\mathcal{K}$. It is* decided *w.r.t. $\mathcal{K}$ and $\mathcal{L}$ if it is not undecided w.r.t. $\mathcal{K}$ and $\mathcal{L}$.*

In principle, our attribute exploration algorithm tries to decide each undecided implications by either adding it to $\mathcal{L}$ or extending $\mathcal{K}$ such that it refutes the implication. If all implications are decided, then our goal is achieved.

Proposition 2. *Assume that $\mathcal{K} \leq \overline{\mathcal{K}} \subseteq Mod(\mathcal{L})$ and that all implications are decided w.r.t. $\mathcal{K}$ and $\mathcal{L}$. Then $\mathcal{L}$ is complete for $Imp(\overline{\mathcal{K}})$ and $\mathcal{K}$ refutes all implications not belonging to $Imp(\overline{\mathcal{K}})$.*

How can we find—and let the expert decide—all undecided implications without considering all implications? The following proposition motivates why it is sufficient to consider implications whose left-hand sides are $\mathcal{L}$-closed.

Proposition 3. *Let $\mathcal{L}$ be a set of implications and $L \to R$ an implication. Then, $L \to R$ follows from $\mathcal{L}$ iff $\mathcal{L}(L) \to R$ follows from $\mathcal{L}$.*

Concerning right-hand sides, Proposition 1 says that the largest right-hand side R such that $L \to R$ is not refuted by $\mathcal{K}$ is $R = \mathcal{K}(L)$. Putting these two observations together, we only need to consider implications of the form $L \to \mathcal{K}(L)$ where L is $\mathcal{L}$-closed. In order to enumerate all relevant left-hand sides, we can thus use the well-known approach from FCA for enumerating closed sets in the lectic order [5].

Definition 6. *Assume that $M = \{m_1, \ldots, m_n\}$ and fix some linear order $m_1 < m_2 < \cdots m_n$ on M. The* lectic order $<$ *is defined as follows: for $m_i \in M$ and $A, B \subseteq M$ we define $A <_i B$* iff *$m_i \in B \setminus A$* and *$A \cap \{m_1, \ldots, m_{i-1}\} = B \cap \{m_1, \ldots, m_{i-1}\}$. The order $<$ is the union of the orders $<_i$.*

Obviously, $<$ extends the strict subset order, and thus $\varnothing$ is the smallest and M the largest set w.r.t. $<$.

Proposition 4. *Given a set of implications $\mathcal{L}$ and an $\mathcal{L}$-closed set $A \subsetneq M$, the next $\mathcal{L}$-closed set following A in the lectic order is $\mathcal{L}((A \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$ where j is maximal such that $A <_j \mathcal{L}((A \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$.*

In [3], an extension of the usual attribute exploration algorithm that is based on this idea is described. Here, we only give in detail the instantiation used to complete DL KBs (see Algorithm 1 in the next section).

3 Completing DL Knowledge Bases

In order to represent knowledge about an application domain using Description Logics (DLs) (see [1] for more details and references), one usually first defines the relevant concepts of this domain, and then describes relationships between concepts and between individuals and concepts in the knowledge base. To construct concepts, one starts with a set N_C of *concept names* (unary predicates) and a set

Name of constructor	Syntax	Semantics
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
general concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$

Table 1. Conjunction, negation, GCIs, and ABox assertions.

N_R of *role names* (binary predicates), and builds complex *concept descriptions* out of them by using the *concept constructors* provided by the particular *description language* being used. In addition, a set N_I of *individual names* is used to refer to domain elements. In this paper, we do not fix a specific set of constructors, we only assume that the constructors conjunction and negation (see the upper part of Table 1) are available. A *TBox* is a finite set of general concept inclusions (GCIs), and an *ABox* is a finite set of concept and role assertions (see the lower part of Table 1). A *knowledge base (KB)* consists of a TBox together with an ABox. The semantics of concept descriptions, TBoxes, and ABoxes is given in terms of an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ (the *domain*) is a non-empty set, and $\cdot^{\mathcal{I}}$ (the *interpretation function*) maps each concept name $A \in N_C$ to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, each role name $r \in N_R$ to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and each individual name $a \in N_I$ to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Concept descriptions C are also interpreted as sets $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, which are defined inductively, as seen in the semantics column of Table 1 for the constructors conjunction and negation. An interpretation $\mathcal{I}$ is a *model* of the TBox $\mathcal{T}$ (the ABox $\mathcal{A}$) if it satisfies all its GCIs (assertions) in the sense shown in the semantics column of the table. In case $\mathcal{I}$ is a model of both $\mathcal{T}$ and $\mathcal{A}$, it is also called a model of the knowledge base $(\mathcal{T}, \mathcal{A})$. If there is such a model, we call the KB *consistent*.

Given a KB $(\mathcal{T}, \mathcal{A})$, concept descriptions C, D, and an individual name a, the inference problems *subsumption* and *instance* are defined as follows: C is *subsumed* by D w.r.t. $\mathcal{T}$ ($C \sqsubseteq_{\mathcal{T}} D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$; and a is an *instance* of C w.r.t. $\mathcal{T}$ and $\mathcal{A}$ ($\mathcal{T}, \mathcal{A} \models C(a)$) if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ holds for all models of $(\mathcal{T}, \mathcal{A})$. For most DLs, these problems are decidable, and there exist highly optimized DL reasoners such as FaCT, RACER, and Pellet that can solve these problems for very expressive DLs on large practical KBs.

Our approach for completing DL knowledge bases applies to arbitrary DLs, provided that the description language allows at least for conjunction and negation, the TBox formalism for GCIs, the ABox formalism for concept assertions, and the subsumption and the instance problem are decidable.

DLs and partial contexts Let $(\mathcal{T}, \mathcal{A})$ be a consistent DL knowledge base, and M be a finite set of concept descriptions. An individual name a occurring in $\mathcal{A}$ gives rise to the *partial object description* $pod_{\mathcal{T},\mathcal{A}}(a, M) := (A, S)$ where $A := \{C \in M \mid \mathcal{T}, \mathcal{A} \models C(a)\}$ and $S := \{C \in M \mid \mathcal{T}, \mathcal{A} \models \neg C(a)\}$, and the whole ABox induces the partial context $\mathcal{K}_{\mathcal{T},\mathcal{A}}(M) := \{pod_{\mathcal{T},\mathcal{A}}(a, M) \mid a \text{ an individual name in } \mathcal{A}\}$. Note that $pod_{\mathcal{T},\mathcal{A}}(a, M)$ is indeed a pod since $(\mathcal{T}, \mathcal{A})$ was assumed to be consistent. Similarly, any element $d \in \Delta^{\mathcal{I}}$ of an interpre-

tation $\mathcal{I}$ gives rise to the *full object description* $fod_{\mathcal{I}}(d, M) := (\overline{A}, \overline{S})$ where $\overline{A} := \{C \in M \mid d \in C^{\mathcal{I}}\}$ and $\overline{S} := \{C \in M \mid d \in (\neg C)^{\mathcal{I}}\}$, and the whole interpretation induces the full context $\mathcal{K}_{\mathcal{I}}(M) := \{fod_{\mathcal{I}}(d, M) \mid d \in \Delta^{\mathcal{I}}\}$.

Proposition 5. *Let $(\mathcal{T}, \mathcal{A}), (\mathcal{T}', \mathcal{A}')$ be DL KBs such that $\mathcal{T} \subseteq \mathcal{T}'$ and $\mathcal{A} \subseteq \mathcal{A}'$, M a set of concept descriptions, and $\mathcal{I}$ a model of $(\mathcal{T}', \mathcal{A}')$. Then $\mathcal{K}_{\mathcal{T},\mathcal{A}}(M) \leq \mathcal{K}_{\mathcal{T}',\mathcal{A}'}(M) \leq \mathcal{K}_{\mathcal{I}}(M)$.*

Next, we straightforwardly transfer the notion of refutation of an implication from partial (full) contexts to knowledge bases (interpretations).

Definition 7. *The implication $L \to R$ over the attributes M is* refuted *by the knowledge base $(\mathcal{T}, \mathcal{A})$ if it is refuted by $\mathcal{K}_{\mathcal{T},\mathcal{A}}(M)$, and it is* refuted *by the interpretation $\mathcal{I}$ if it is refuted by $\mathcal{K}_{\mathcal{I}}(M)$. If an implication is not refuted by $\mathcal{I}$, then we say that it* holds in $\mathcal{I}$. *In addition, we say that $L \to R$* follows from $\mathcal{T}$ *if $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$, where $\sqcap L$ and $\sqcap R$ respectively stand for the conjunctions $\bigsqcap_{C \in L} C$ and $\bigsqcap_{D \in R} D$.*

Obviously, the implication $L \to R$ holds in $\mathcal{I}$ iff $(\sqcap L)^{\mathcal{I}} \subseteq (\sqcap R)^{\mathcal{I}}$. As an immediate consequence of this fact, we obtain:

Proposition 6. *Let $\mathcal{T}$ be a TBox and $\mathcal{I}$ be a model of $\mathcal{T}$. If $L \to R$ follows from $\mathcal{T}$, then it holds in $\mathcal{I}$.*

Completion of DL KBs: formal definition and algorithm We are now ready to define what we mean by a completion of a DL knowledge base. Intuitively, the knowledge base is supposed to describe an intended model. For a fixed set M of "interesting" concepts, the knowledge base is complete if it contains all the relevant knowledge about implications between these concepts. To be more precise, if an implication holds in the intended interpretation, then it should follow from the TBox, and if it does not hold in the intended interpretation, then the ABox should contain a counterexample.

Definition 8. *Let $(\mathcal{T}, \mathcal{A})$ be a consistent DL knowledge base, M a finite set of concept descriptions, and $\mathcal{I}$ a model of $(\mathcal{T}, \mathcal{A})$. Then $(\mathcal{T}, \mathcal{A})$ is* M-complete *(or* complete *if M is clear from the context)* w.r.t. *$\mathcal{I}$ if the following three statements are equivalent for all implications $L \to R$ over M:*

1. *$L \to R$ holds in $\mathcal{I}$;*
2. *$L \to R$ follows from $\mathcal{T}$;*
3. *$L \to R$ is not refuted by $(\mathcal{T}, \mathcal{A})$.*

Let $(\mathcal{T}_0, \mathcal{A}_0)$ be a DL knowledge base and $\mathcal{I}$ a model of $(\mathcal{T}_0, \mathcal{A}_0)$. Then $(\mathcal{T}, \mathcal{A})$ is a completion *of $(\mathcal{T}_0, \mathcal{A}_0)$ if it is complete and extends $(\mathcal{T}_0, \mathcal{A}_0)$, i.e., $\mathcal{T}_0 \subseteq \mathcal{T}$ and $\mathcal{A}_0 \subseteq \mathcal{A}$.*

We can now describe an instance of attribute exploration for partial contexts that computes a completion of a given knowledge base $(\mathcal{T}_0, \mathcal{A}_0)$ w.r.t. a fixed model $\mathcal{I}$. We assume that the *expert* has or can obtain enough information

Algorithm 1 Completion of DL knowledge bases

Input: $M = \{m_1, \ldots, m_n\}$, $(\mathcal{T}_0, \mathcal{A}_0)$ {attribute set; KB with model $\mathcal{I}$}
$\mathcal{T} := \mathcal{T}_0$, $\mathcal{A} := \mathcal{A}_0$
$\mathcal{L} := \varnothing$ {initial empty set of implications}
$P := \varnothing$ {lectically smallest $\mathcal{L}$-closed subset of M}
while $P \neq M$ **do**
  Compute $\mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$
  if $P \neq \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ **then** {check whether the implication follows from $\mathcal{T}$}
    if $\sqcap P \sqsubseteq_{\mathcal{T}} \sqcap \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ **then**
      $\mathcal{L} := \mathcal{L} \cup \{P \to \mathcal{K}_{\mathcal{T},\mathcal{A}}(P) \setminus P\}$
      $P_{\text{new}} := \mathcal{L}((P \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$ for the max. j that satisfies $P <_j \mathcal{L}((P \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$
    else
      Ask expert if $P \to \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is refuted by $\mathcal{I}$.
      if no **then** $\{\sqcap P \sqsubseteq \sqcap \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is satisfied in $\mathcal{I}\}$
        $\mathcal{L} := \mathcal{L} \cup \{P \to \mathcal{K}_{\mathcal{T},\mathcal{A}}(P) \setminus P\}$
        $P_{\text{new}} := \mathcal{L}((P \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$ for the max. j that satisfies $P <_j \mathcal{L}((P \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$
        $\mathcal{T} := \mathcal{T} \cup \{\sqcap P \sqsubseteq \sqcap(\mathcal{K}_{\mathcal{T},\mathcal{A}}(P) \setminus P)\}$
      else
        Get an ABox $\mathcal{A}'$ from the expert such that $\mathcal{A} \subseteq \mathcal{A}'$, $\mathcal{I}$ is a model of $\mathcal{A}'$, and $P \to \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is refuted by $\mathcal{A}'$
        $\mathcal{A} := \mathcal{A}'$ {extend the ABox}
        $P_{\text{new}} := P$ {P not changed}
      end if
    end if
  else
    $P_{\text{new}} := \mathcal{L}((P \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$ for the max. j that satisfies $P <_j \mathcal{L}((P \cap \{m_1, \ldots, m_{j-1}\}) \cup \{m_j\})$
  end if
  $P := P_{\text{new}}$
end while

about this model to be able to answer questions of the form "Is $L \to R$ refuted by $\mathcal{I}$?". If the answer is "no," then $L \to R$ holds according to the expert's opinion, and is thus added to the implication base computed by the algorithm. In addition, the GCI $\sqcap L \sqsubseteq \sqcap R$ is added to the TBox. Since $L \to R$ is not refuted by $\mathcal{I}$, the interpretation $\mathcal{I}$ is still a model of the new TBox obtained this way. If the answer is "yes," then the expert is asked to extend the current ABox (by adding appropriate assertions on either old or new individual names) such that the extended ABox refutes $L \to R$ and $\mathcal{I}$ is still a model of this ABox. Because of Proposition 6, before actually asking the expert whether the implication $L \to R$ is refuted by $\mathcal{I}$, we can first check whether $\sqcap L \sqsubseteq \sqcap R$ already follows from the current TBox. If this is the case, then we know that $L \to R$ cannot be refuted by $\mathcal{I}$. This completion algorithm for DL knowledge bases is described in more detail in Algorithm 1.

Theorem 1. *Let $(\mathcal{T}_0, \mathcal{A}_0)$ be a consistent knowledge base, M a finite set of concept descriptions, and $\mathcal{I}$ a model of $(\mathcal{T}_0, \mathcal{A}_0)$, and let $(\mathcal{T}, \mathcal{A})$ be the knowledge base computed by Algorithm 1. Then $(\mathcal{T}, \mathcal{A})$ is a completion of $(\mathcal{T}_0, \mathcal{A}_0)$.*

4 Conclusion

We have extended the attribute exploration algorithm from FCA to partial contexts, and have shown how the extended algorithm can be used to complete DL knowledge bases, using both DL reasoning and answers given by a domain expert. This algorithm inherits its complexity from "classical" attribute exploration [5]: in the worst case, which occurs if there are few or many relationships between attributes, it is exponential in the number of attributes. Regarding the number of questions asked to the expert, it easily follows from results shown in [2] that our method asks the minimum number of questions with positive answers. For the questions with negative answers, the behaviour depends on the answers given by the expert: FCA-theory implies that there always exist counterexamples that, if taken in each step, ensure a minimal number of questions with negative answers. In general, however, one cannot assume that the expert provides these "best" counterexamples.

Based on the results presented in the previous two sections, we have implemented a first experimental version of a tool for completing DL knowledge bases as an extension of the ontology editor Swoop [8], which uses the system Pellet as underlying reasoner [14]. We have just started to evaluate our tool on the OWL ontology for human protein phosphatases mentioned in the introduction, with biologists as experts, and hope to get first significant results on its usefulness and performance in the near future. Unsurprisingly, we have observed that the experts sometimes make errors when answering implication questions. Hence we will extend the completion tool such that it supports detecting such errors and also allows to correct errors without having to restart the exploration from scratch.

From a theoretical point of view, we will also look at extensions of our definition of a complete KB. As a formalization of what "all relationships between interesting concepts" really means, we have used subsumption relationships between conjunctions of elements of the set of interesting concepts M. One could also consider more complex relationships by fixing a specific DL $\mathcal{D}$, and then taking, as attributes, all $\mathcal{D}$-concept descriptions over the concept "names" from M. The immediate disadvantage of this extension is that, in general, the set of attributes becomes infinite, and thus termination of the exploration process is no longer guaranteed. An extension of classical attribute exploration (i.e., for full contexts) in this direction is described in [12]. The main idea to deal with the problem of an infinite attribute set used there is to restrict the attention to concept descriptions with a bounded role depth. But even though this makes the attribute set finite, its size is usually too large for practical purposes.

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
2. F. Baader, B. Ganter, U. Sattler, and B. Sertkaya. Completing description logic knowledge bases using formal concept analysis. LTCS-Report LTCS-06-02, Chair for Automata Theory, Inst. for Theoretical Computer Science, TU Dresden, Germany, 2006. See http://lat.inf.tu-dresden.de/research/reports.html.
3. F. Baader, B. Ganter, U. Sattler, and B. Sertkaya. Completing description logic knowledge bases using formal concept analysis. In *Proc. of the Twentieth Int. Joint Conf. on Artificial Intelligence (IJCAI'07)*. AAAI Press, 2007.
4. P. Burmeister and R. Holzer. Treating incomplete knowledge in formal concept analysis. In *Formal Concept Analysis*, vol. 3626 of *LNCS*, pages 114–126. Springer, 2005.
5. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin, Germany, 1999.
6. I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: the making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.
7. A. Kalyanpur, B. Parsia, E. Sirin, and B. C. Grau. Repairing unsatisfiable concepts in OWL ontologies. *The Semantic Web: Research and Applications. Proc. of the 3rd European Semantic Web Conf. (ESWC 2006)*, vol. 4011 of *LNCS*, pages 170–184. Springer, 2006.
8. A. Kalyanpur, B. Parsia, E. Sirin, B. C. Grau, and J. A. Hendler. Swoop: A web ontology editing browser. *Journal of Web Semantics*, 4(2):144–153, 2006.
9. H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen. The protégé OWL plugin: An open development environment for semantic web applications. *Int. Semantic Web Conf.*, vol. 3298 of *LNCS*, pages 229–243. Springer, 2004.
10. D. Oberle, R. Volz, S. Staab, and B. Motik. An extensible ontology software environment. *Handbook on Ontologies*, Int. Handbooks on Information Systems, pages 299–320. Springer, 2004.
11. S. A. Obiedkov. Modal logic for evaluating formulas in incomplete contexts. In *Proc. of the 10th Int. Conf. on Conceptual Structures, (ICCS 2002)*, vol. 2393 of *LNCS*, pages 314–325. Springer, 2002.
12. S. Rudolph. Exploring relational structures via $\mathcal{FLE}$. *12th Int. Conf. on Conceptual Structures (ICCS 2004)*, vol. 3127 of *LNCS*, pages 196–212, 2004.
13. S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. *Proc. of the Eighteenth Int. Joint Conf. on Artificial Intelligence (IJCAI'03)*, pages 355–362. Morgan Kaufmann, 2003.
14. E. Sirin and B. Parsia. Pellet: An OWL DL reasoner. In *Proc. of the 2004 Int. Workshop on Description Logics (DL2004)*, vol. 104 of *CEUR Workshop Proc.*. CEUR-WS.org, 2004.
15. R. Wille. Restructuring lattice theory: An approach based on hierarchies of concepts. *Ordered Sets*, pages 445–470. Reidel, Dordrecht-Boston, 1982.
16. K. Wolstencroft, A. Brass, I. Horrocks, P. W. Lord, U. Sattler, D. Turi, and R. Stevens. A little semantic web goes a long way in biology. In *Proc. of the 4th Int. Semantic Web Conf. (ISWC 2005)*, vol. 3729 of *LNCS*, pages 786–800. Springer, 2005.