Explaining User Errors in Description Logic Knowledge Base Completion

Barış Sertkaya*

TU Dresden, Germany
sertkaya@tcs.inf.tu-dresden.de

Abstract. In our previous work we have developed a method for completing a Description Logic knowledge base w.r.t. a fixed interpretation by asking questions to a domain expert. Our experiments showed that during this process the domain expert sometimes gives wrong answers to the questions, which cause the resultant knowledge base to have unwanted consequences. In the present work we consider the problem of explaining the reasons of such unwanted consequences in knowledge base completion. We show that in this setting the problem of deciding the existence of an explanation within a specified cardinality bound is NP-complete, and the problem of counting explanations that are minimal w.r.t. set inclusion is #P-complete. We also provide an algorithm that computes one minimal explanation by performing at most polynomially many subsumption tests.

1 Introduction

Description Logics (DLs) [BCM+03] are a successful family of logic-based knowledge representation formalisms that are used to represent the conceptual knowledge of an application domain in a structured and formally well-understood way. They 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 [HPSvH03].

As a consequence of this standardization, several ontology editors support OWL [HTR06,OVSM04,KPS+06], 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. 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 our previous work [BGSS07], we have considered a different quality dimension: *completeness*. We have developed a method that, given a DL knowledge base (KB) describing an application domain,

* Supported by German Research Foundation (DFG, BA 1122/12-1)

supports the knowledge engineer in checking whether the KB contains all the relevant information about the domain, namely: are all the relevant

- subclass/superclass relationships that hold in the domain captured by the KB?
- individuals existing in the domain represented in the KB?

Clearly, such questions cannot be answered by an automated tool alone. In order to check whether a given relationship between classes—which does not already follow from the KB—holds in the domain, one needs to ask a domain expert, and the same is true for questions regarding the existence of individuals not represented in the KB. The method developed in the aforementioned work supports the knowledge engineer in checking whether the KB captures all relevant information about the application domain, and extending it appropriately if this is not the case. The method achieves this by asking the knowledge engineer questions of the form "*is it true that instances of the classes* $C_1, \ldots, C_n$ *are also instances of the classes* $D_1, \ldots, D_m$*?*". The knowledge engineer is expected to either confirm it, in which case a new axiom of the application domain has been discovered and it is added to the KB, or to reject it, in which case she is asked to provide a counterexample. The method is based on *attribute exploration* [Gan84], which is a novel knowledge acquisition algorithm developed in Formal Concept Analysis (FCA) [GW99]. The use of attribute exploration ensures that, on the one hand, during KB completion the interaction with the expert is kept to a minimum, and on the other hand, the resultant KB is complete in a certain well-defined sense.

Our experiments with a prototype implementation of the KB completion method showed that during completion the knowledge engineer sometimes introduces errors to the KB by confirming questions that actually are not true in the application domain. As a result, the completed KB has unwanted consequences. In the present work we investigate the problem of finding *explanations* of such unwanted consequences, i.e., subsets of the axioms added to the KB during completion, from which these unwanted consequences follow. While looking for explanations, we do not consider the whole completed KB, but only a subset of it containing the axioms added during the completion. In [BPS07], Baader et. al. have investigated axiom pinpointing in the DL $\mathcal{EL}^+$ in a similar setting where the explanations are searched only within a subset of the KB. They have shown that even for the propositional Horn fragment, in this setting the problem of deciding the existence of an explanation within a specified cardinality bound is NP-complete, and there can be exponentially many explanations that are minimal w.r.t. set inclusion. In our setting, the axioms added to the KB during completion are propositional Horn as well. However, the difference is that they are not arbitrary propositional Horn axioms. They are of a restricted syntactical form such that the resulting set of axioms form a canonical base called the Duquenne-Guigues Base [GD86]. We show here that the intractability result in [BPS07] still holds under this restriction, and the problem of counting minimal explanations is #P-complete. Moreover, despite these negative results we

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

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

provide an algorithm that computes one minimal explanation by performing at most polynomially many subsumption tests.

2 Description Logics

In DLs, one formalizes the relevant notions of an application domain by *concept descriptions.* A concept description is an expression built from *atomic concepts* which are unary predicates, and *atomic roles*, which are binary predicates, by using the *concept constructors* provided by the particular DL language in use. The set of atomic concepts is usually represented with N_C, and the set of atomic roles is usually represented with N_R. In the present paper, we do not fix a specific set of constructors since our results apply to arbitrary DLs as long as they allow for the constructors conjunction and negation (see the upper part of Table 1).

Typically, a *DL knowledge base* consists of a *terminological box* (*TBox*) which defines the terminology of an application domain, and an *assertional box* (*ABox*) which contains facts about a specific world. In its simplest form, a TBox is a set of *concept definitions* of the form $A \equiv C$ that assigns the concept name A to the concept description C. The concept names occurring on the left-hand side of a concept definition are called *defined concepts*, and the others are called *primitive concepts.* We call a finite set of *general concept inclusion (GCI)* axioms a *general TBox.* A GCI is an expression of the form $C \sqsubseteq D$, where C and D are two possibly complex concept descriptions. It states a subconcept/superconcept relationship between the two concept descriptions. An ABox is a set of *concept assertions* and *role assertions* (see the lower part of Table 1). A concept assertion $C(a)$ means that the individual a is an instance of the concept C, and a role assertion $r(a,b)$ means that the individuals a and b are related via the r relation.

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 concept definitions and 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 KB $(\mathcal{T}, \mathcal{A})$. If there is such a model we say that the KB is *consistent*.

Given a KB $(\mathcal{T}, \mathcal{A})$, concept descriptions C, D, and an individual name a, the traditional 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 $(\mathcal{T}, \mathcal{A})$. Given a TBox $\mathcal{T}$ and a GCI $C \sqsubseteq_{\mathcal{T}} D$, we call a $\mathcal{T}' \subseteq \mathcal{T}$ an *explanation* of $C \sqsubseteq D$ if $C \sqsubseteq_{\mathcal{T}'} D$. For most DLs, subsumption and instance problems are decidable, and there exist highly optimized DL reasoners such as FaCT++ [TH06], RacerPro [HM01], and and Pellet [SP04] that can solve these problems for very expressive DLs on large practical KBs.

3 DL Knowledge Base Completion

Intuitively, a KB is supposed to describe an intended model. For a fixed set M of "interesting" concepts, we say that a KB is *complete* if it contains all the relevant knowledge about subconcept/superconcept relationships that hold between these concepts in the intended interpretation. To be more precise, if a subsumption relationship 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. More formally, let us say that the element $d \in \Delta^{\mathcal{I}}$ of an interpretation $\mathcal{I}$ *satisfies* the subsumption relation $C \sqsubseteq D$ if $d \notin C^{\mathcal{I}}$ or $d \in D^{\mathcal{I}}$, and that $\mathcal{I}$ *satisfies* this relation if every element of $\Delta^{\mathcal{I}}$ satisfies it. In addition, let us call the individual name a a *counterexample* in $(\mathcal{T}, \mathcal{A})$ to the subsumption relation $C \sqsubseteq D$ if $\mathcal{T}, \mathcal{A} \models C(a)$ and $\mathcal{T}, \mathcal{A} \models \neg D(a)$, and say that $\mathcal{A}$ *refutes* $C \sqsubseteq D$ if $\mathcal{A}$ contains a counterexample to this subsumption relation. Based on these, completeness of a DL KB is defined as follows:

Definition 1. *Let $(\mathcal{T}, \mathcal{A})$ be a consistent DL KB, 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 statements are equivalent for all subsets L, R of M, where $\sqcap L$ stands for $\bigsqcap_{C \in L} C$:*

1. *$\sqcap L \sqsubseteq \sqcap R$ is satisfied by $\mathcal{I}$;*
2. *$\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$ holds;*
3. *$(\mathcal{T}, \mathcal{A})$ does not contain a counterexample to $\sqcap L \sqsubseteq \sqcap R$.*

In [BGSS07], we have developed a KB completion method that is based on Formal Concept Analysis (FCA) [GW99]. FCA is a field of mathematics based on the lattice-theoretic formalization of the notions of a concept and a conceptual hierarchy. Our method uses the well known knowledge acquisition algorithm of FCA, namely *attribute exploration* [Gan84]. Given a KB $(\mathcal{T}, \mathcal{A})$ and a set of concept descriptions M, at each iteration our method produces a subsumption statement $\sqcap L \sqsubseteq \sqcap R$ (where $L, R \subseteq M$) that is not refuted by $\mathcal{A}$. It first asks the DL reasoner whether this subsumption relation already follows from $\mathcal{T}$. If not then this knowledge is missing in the KB and a domain expert is asked whether this subsumption relation holds in the application domain. The question asked to the expert is of the form: "*in your application domain, is it true that instances of* $\sqcap L$ *are also instances of* $\sqcap R$*?*". We assume that the domain expert has enough information about the application domain to be able to answer such questions. If she answers "*yes*" then a new axiom of the application domain, i.e., an axiom that does not yet follow from the knowledge base, has been discovered and a new GCI $\sqcap L \sqsubseteq \sqcap R$ is added to $\mathcal{T}$. If she answers "*no*" then she is asked to extend $\mathcal{A}$ (either by adding a new individual, or by modifying an existing individual) such that $\mathcal{A}$ contains a counterexample to $\sqcap L \sqsubseteq \sqcap R$. The iteration continues until all such questions are answered. Once all such questions are answered, the resulting KB will be complete in the sense that is introduced in Definition 1 (for details of the completion algorithm see [BGSS07]).

One important point here is that the KB completion algorithm does not naively enumerate all possible subsumption relations $\sqcap L \sqsubseteq \sqcap R$ that are not refuted by $\mathcal{A}$. This would mean too many unnecessary questions to the expert. The algorithm produces the questions in a certain lexicographic order such that the interaction with the expert is kept to a minimum. More precisely, the algorithm asks the expert the minimum number of questions that have a "*yes*" answer, i.e., questions that result in a new GCI in the TBox. In FCA terminology, such a set of axioms is called a *Duquenne-Guigues Base* [GD86]. It is well known that among all sets of axioms that have exactly the same consequences, the Duquenne-Guigues Base contains the smallest number of axioms. That is, no set of axioms with smaller cardinality can have the same set of consequences as the Duquenne-Guigues Base. The axioms of a Duquenne-Guigues Base, thus the questions that had a "*yes*" answer during a completion process, have the following property which is going to be used in Section 4 (for more information on the Duquenne-Guigues Base and its properties see [GW99]):

Lemma 1. *Let $\mathcal{T}$ be a set of GCIs that is a Duquenne-Guigues Base on the finite set of concepts M. Then every GCI $\sqcap L \sqsubseteq \sqcap R$ in $\mathcal{T}$ where $L, R \subseteq M$ satisfies the following:*

1. *L is closed w.r.t. $\mathcal{T}' := \mathcal{T} \setminus \{\sqcap L \sqsubseteq \sqcap R\}$, which means that for every $\sqcap L' \sqsubseteq \sqcap R' \in \mathcal{T}'$, $L' \subseteq L$ implies $R' \subseteq L$.*
2. *$L \cup R$ is closed w.r.t. $\mathcal{T} \setminus \{\sqcap L \sqsubseteq \sqcap R\}$.*

Based on the algorithm presented in [BGSS07] we have implemented a first experimental version of the method as an extension called InstExp[1] to the Swoop ontology editor [KPS+06]. Our first experiments with InstExp showed that during completion, unsurprisingly the expert sometimes makes errors when answering the questions. In the simplest case, the error makes the KB inconsistent, which can easily be detected by DL reasoning and the expert can be notified about it. However, in this case an explanation for the reason of inconsistency is often needed to understand and fix the error. The situation gets more

[1] available under `http://lat.inf.tu-dresden.de/~sertkaya/InstExp`

complicated if the error does not immediately lead to inconsistency but the expert realizes in the later steps, or only after completion that she has accepted a wrong GCI in one of the previous steps. In this case, the completed KB will have unwanted consequences. In the next section we are going to investigate axiom pinpointing in the KB completion setting. We are going to look for methods for explaining user errors introduced to the TBox during KB completion.

4 Axiom pinpointing in KB Completion

In [BPS07] Baader et. al. have considered axiom pinpointing in the DL $\mathcal{EL}^+$ in a setting where the TBox consists of two kinds of axioms, namely the trusted ones whose correctness is no longer doubted, and the refutable ones whose correctness is not yet for sure. Trusted axioms form the so-called *static* part of the TBox, and the others form the *refutable* part. The static part of the TBox is assumed to be always present, and axioms explaining a certain consequence are searched only in the refutable part of the TBox. In our KB completion scenario we have a similar setting. We assume that the axioms in the initial TBox, which we have at the beginning of completion, are trusted i.e., they have no unwanted consequences. However, as mentioned before, during completion sometimes by mistake the domain expert confirms questions that actually are not true in the application domain, which introduce errors to the TBox. In this case the completed KB will have unwanted consequences. Therefore we consider the GCIs added to the TBox during completion as refutable ones, and for finding explanations we do not consider the whole TBox but only a subset of it that contains the GCIs that have been added by the domain expert. Namely, the GCIs that have been either confirmed by the DL reasoner or by the domain expert during completion.

One important point here that differs from [BPS07] is that, the GCIs that have been confirmed (either by the DL reasoner, or by the domain expert) during completion form a Duquenne-Guigues Base. Thus the GCIs in this set have the restricted syntactical form satisfying the property in Lemma 1. This is the main distinguishing feature of our work. In [BPS07] Baader et. al. have considered the complexity of axiom pinpointing in the DL $\mathcal{EL}^+$ for the setting described above. They have shown that even for the propositional Horn fragment, the problem of deciding the existence of an explanation within a specified cardinality bound is NP-complete, and there can be exponentially many explanations that are minimal w.r.t. set inclusion. Here we show that in our KB completion scenario, despite the restricted form of the GCIs in the TBox, the above problem remains intractable. First we give an example showing that a GCI can have exponentially many explanations that are minimal w.r.t. set inclusion.

Example 1. Consider the TBox

$$\mathcal{T} := \{X \sqcap B_{i-1} \sqsubseteq P_i \sqcap Q_i, Y \sqcap P_i \sqsubseteq B_i, Y \sqcap Q_i \sqsubseteq B_i \mid 1 \leq i \leq n\}.$$

It is not difficult to see that none of the left-hand sides is contained in another left-hand side or in the union of left- and right-hand sides of another axiom, i.e., it obeys the property mentioned in Lemma 1. Moreover its size is linear in n, and it has 2^n minimal subsets that explain the axiom $B_0 \sqcap X \sqcap Y \sqsubseteq B_n$ since for each i, $1 \leq i \leq n$, B_i can be generated by the axiom $Y \sqcap P_i \sqsubseteq B_i$ or by $Y \sqcap Q_i \sqsubseteq B_i$.

Now we show that the problem of checking the existence of an explanation within a specified cardinality bound still remains NP-complete despite the restricted form of the GCIs in the TBox. In the following, for a set of concept names L, $\sqcap L$ denotes the conjunction $\bigsqcap_{C \in L} C$.

Problem: MINIMUM CARDINALITY EXPLANATION
Input: A set $\mathcal{T}$ of GCIs satisfying the properties in Lemma 1, sets L and R of concept names occurring in $\mathcal{T}$ such that $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$, a natural number n.
Question: Is there an explanation of $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$ in $\mathcal{T}$ with cardinality less than or equal to n, i.e., is there a set of GCIs $\mathcal{T}' \subseteq \mathcal{T}$ such that $\sqcap L \sqsubseteq_{\mathcal{T}'} \sqcap R$ and $|\mathcal{T}'| \leq n$?

Theorem 1. MINIMUM CARDINALITY EXPLANATION *is* NP*-complete.*

Proof. The problem is in NP. We can nondeterministically guess a subset $\mathcal{T}'$ of $\mathcal{T}$ with cardinality n, and in polynomial time check whether $\sqcap L \sqsubseteq_{\mathcal{T}'} \sqcap R$. This test can indeed be done in polynomial time by using the linear time result for propositional Horn clauses in [DG84] since our GCIs, whose left- and right-hand sides both consist of only conjunctions of concept names, can be written as Horn clauses.

In order to show NP-hardness, we are going to give a reduction from the NP-complete problem VERTEX COVER [GJ90]. Recall that a vertex cover of the graph $\mathcal{G} = (V, E)$ is a set $W \subseteq V$ such that for every edge $\{u, v\} \in E$, $u \in W$ holds, or $v \in W$ holds. The problem VERTEX COVER is defined as follows:

Problem: VERTEX COVER
Input: Graph $\mathcal{G} = (V, E)$, a natural number n.
Question: Is there a vertex cover of $\mathcal{G}$ of size less than or equal to n?

Consider an instance of the VERTEX COVER problem given by $\mathcal{G} = (V, E)$, where $V = \{v_1, \ldots, v_l\}$, $E = \{e_1, \ldots, e_k\}$, and edge $e_i = \{v_{i1}, v_{i2}\}$. For every vertex $v \in V$ we introduce a concept name X_v, for every edge $e_i, 1 \leq i \leq k$, we introduce a concept name Q_i, and finally two more additional concept names A and B. Using these concept names we construct the following set of GCIs:

$$\mathcal{T} := \{\mathsf{X}_v \sqsubseteq \bigsqcap_{\{i \;|v \,\in\, e_i\}} \mathsf{Q}_i \mid v \in V\} \;\cup\; \{\mathsf{A} \sqcap \bigsqcap_{1 \leq i \leq k} \mathsf{Q}_i \sqsubseteq B\}.$$

Note that none of the GCIs in $\mathcal{T}$ contains the left-hand side of another GCI in its left-hand side or in the union of its left- and right-hand sides. That is, $\mathcal{T}$ satisfies the property mentioned in Lemma 1. In addition to $\mathcal{T}$, we construct the following GCI that follows from $\mathcal{T}$:

$$\psi: \quad \mathsf{A} \sqcap \bigsqcap_{v \in V} \mathsf{X}_v \sqsubseteq B.$$

Obviously, this construction can be done in polynomial time. Assume $W \subseteq V$ is a vertex cover of $\mathcal{G}$. Then the following subset of $\mathcal{T}$ constructed by using W is an explanation of ψ:

$$\mathcal{T}' := \{\mathsf{X}_w \sqsubseteq \bigsqcap_{\{i \ | w \ \in \ e_i\}} \mathsf{Q}_i \mid w \in W\} \ \cup \ \{\mathsf{A} \sqcap \bigsqcap_{1 \leq i \leq k} \mathsf{Q}_i \sqsubseteq B\}.$$

It is not difficult to see that $\bigsqcap_{w \in W} \mathsf{X}_w \sqsubseteq_{\mathcal{T}'} \bigsqcap_{w \in W} \bigsqcap_{\{i \ | w \ \in \ e_i\}} \mathsf{Q}_i$. Since W is a vertex cover, it contains at least one vertex from every edge e_i, $1 \leq i \leq k$. Thus, $\bigsqcap_{w \in W} \bigsqcap_{\{i \ | w \ \in \ e_i\}} \mathsf{Q}_i \equiv \bigsqcap_{1 \leq i \leq k} \mathsf{Q}_i$, which implies $\bigsqcap_{w \in W} \mathsf{X}_w \sqsubseteq_{\mathcal{T}'} \bigsqcap_{1 \leq i \leq k} \mathsf{Q}_i$, which in turn implies that $\mathsf{A} \sqcap \bigsqcap_{v \in V} \mathsf{X}_v \sqsubseteq_{\mathcal{T}'} \mathsf{A} \sqcap \bigsqcap_{1 \leq i \leq k} \mathsf{Q}_i \sqsubseteq_{\mathcal{T}'} \mathsf{B}$. Thus, we have shown that $\mathcal{T}'$ is an explanation of ψ. Note that if the size of W is n, then $\mathcal{T}'$ contains exactly $n + 1$ axioms. Thus if $\mathcal{G}$ has a vertex cover of size less than or equal to n, then ψ has an explanation in $\mathcal{T}$ of size less than or equal to $n + 1$. The other direction of the claim is shown easily in the similar way. □

In applications where one is interested in all explanations that are minimal w.r.t. set inclusion, it might be useful to know in advance how many of them exist. Next we consider this counting problem. It turns out that it is hard for the counting complexity class #P [Val79a], i.e., it is intractable.

Problem: #MINIMAL EXPLANATION
Input: A set $\mathcal{T}$ of GCIs satisfying the properties in Lemma 1, and sets L and R of concept names occurring in $\mathcal{T}$ such that $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$.
Output: Number of all minimal explanations of $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$ in $\mathcal{T}$, i.e.,
$|\{\mathcal{T}' \subseteq \mathcal{T} \mid \sqcap L \sqsubseteq_{\mathcal{T}'} \sqcap R \text{ and } \forall \mathcal{T}'' \subsetneq \mathcal{T}'. \sqcap L \not\sqsubseteq_{\mathcal{T}''} \sqcap R\}|$.

Theorem 2. #MINIMAL EXPLANATION *is* #P*-complete.*

Proof. The problem is in #P. Given a set of GCIs $\mathcal{T}$ that has the property in Lemma 1, another GCI $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$ and a set $\mathcal{T}' \subseteq \mathcal{T}$, we can in polynomial time verify whether $\sqcap L \sqsubseteq_{\mathcal{T}'} \sqcap R$ using the method mentioned in the proof of Theorem 1.

In order to show #P-hardness, we are going to give a parsimonious reduction from the #P-complete problem #MINIMAL VERTEX COVER, which is the problem of counting the minimal vertex covers of a given graph. It has been shown to be #P-complete in [Val79b]. In our reduction we are going to use the same construction used in the proof of Theorem 1, i.e., from a given graph $\mathcal{G}$ we construct the same set of GCIs $\mathcal{T}$, and the same GCI ψ as in Theorem 1. What we need to show here is that this construction establishes a bijection between minimal vertex covers of $\mathcal{G}$ and minimal explanations of ψ in $\mathcal{T}$.

First we show that it is injective: assume $W \subseteq V$ is a *minimal* vertex cover of $\mathcal{G}$, then the following set of GCIs is a *minimal* explanation of ψ in $\mathcal{T}$:

$$\mathcal{T}' := \{\mathsf{X}_w \sqsubseteq \bigsqcap_{\{i \ | w \ \in \ e_i\}} \mathsf{Q}_i \mid w \in W\} \ \cup \ \{\mathsf{A} \sqcap \bigsqcap_{1 \leq i \leq k} \mathsf{Q}_i \sqsubseteq B\}.$$

In the proof of Theorem 1 we have already shown that $\mathcal{T}'$ is an explanation. Here we need to show that it is minimal as well. If W is minimal, then removal of any

Algorithm 1 Computing one minimal explanation

Input: The set of axioms $\mathcal{T}$ obtained from completion, and sets of concept names L and R s.t. $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$.
$\mathcal{T}' := \mathcal{T}$
for all $t \in \mathcal{T}'$ **do**
  if $\sqcap L \sqsubseteq_{\mathcal{T}' \setminus \{t\}} \sqcap R$ **then** {if $\mathcal{T}' \setminus \{t\}$ is still an explanation}
    $\mathcal{T}' := \mathcal{T}' \setminus \{t\}$
  end if
end for
return $\mathcal{T}'$

vertex w from W will result in a $Y \subsetneq W$ such that $v_{i1} \notin Y$ and $v_{i2} \notin Y$ for some edge e_i. This implies that removal of the corresponding GCI $\mathsf{X}_w \sqsubseteq \bigsqcap_{\{i \mid w \in e_i\}} \mathsf{Q}_i$ from $\mathcal{T}'$ will result in a $\mathcal{T}''$ such that Q_i does not appear on the right-hand side of any of the GCIs, which means that $\mathcal{T}''$ cannot explain ψ, i.e., $\mathcal{T}'$ is minimal.

Now we show that it is surjective: assume $\mathcal{T}'$ is a minimal explanation. Then every Q_i, $1 \leq i \leq k$, occurs at least once on the right-hand side of some GCI of the form $\mathsf{X}_w \sqsubseteq \bigsqcap_{\{i \mid w \in e_i\}} \mathsf{Q}_i$ where $w \in W$, because otherwise $\mathcal{T}'$ cannot explain ψ. Moreover, removal of any GCI of this form from $\mathcal{T}'$ results in a set of GCIs that is not an explanation. That is, removal of any w from W results in a $Y \subsetneq W$ such that $v_{i1} \notin Y$ and $v_{i2} \notin Y$ for some i, i.e., W is minimal. □

Despite these negative results, it is not difficult to find one minimal explanation with at most polynomially many subsumption tests. We can just start with the whole set of axioms obtained from the completion process, iterate over these axioms and eliminate an axiom if the remaining ones still have the consequence in question. It is formally described in Algorithm 1. Obviously, the algorithm terminates since there are only finitely many GCIs in $\mathcal{T}'$, and it is correct since the resulting $\mathcal{T}'$ still explains $\sqcap L \sqsubseteq \sqcap R$, but none of the axioms in $\mathcal{T}'$ can be removed without destroying this property.

5 Concluding remarks

In [BPS07] it was shown that given a set of minimal explanations, the problem of checking whether there exists a minimal explanation that is not contained in the given set is NP-complete. This means that, unless P = NP the set of all minimal explanations cannot be computed in output polynomial time [JPY88], i.e., polynomial in the size of the input *and the output*. We do not know whether this is also the case in our setting for GCIs with restricted form. As future work, on the theoretical side we are going to consider this problem of computing all minimal explanations in the knowledge base completion setting. On the practical side, we are going to implement Algorithm 1 into our KB completion tool InstExp.

In relational databases [Mai83], the notion of Duquenne-Guigues Base occurs as the minimum cover of a given set of functional dependencies, i.e., the minimum (w.r.t. cardinality) set of functional dependencies from which the given set of functional dependencies follow. It is well known that obtaining a minimum cover from a given set of functional dependencies F can be done in time polynomial in the size of F [Mai80]. A corresponding algorithm in the FCA setting has been given in [Rud07]. At this point one might think that our results here can be obtained simply by using the polynomial time algorithm in [Rud07] and the NP-hardness result in [BPS07]. However, this is not the case. If the original set of GCIs $\mathcal{T}$ contains an explanation for a certain consequence, the minimum cardinality set of GCIs obtained from $\mathcal{T}$ by using the algorithm in [Rud07] also contains an explanation, but it is not possible to know the cardinality of this explanation.

References

[BCM+03] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, eds. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BGSS07] F. Baader, B. Ganter, B. Sertkaya, and U. Sattler. Completing Description Logic Knowledge Bases using Formal Concept Analysis. In M. M. Veloso, ed., *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, pp. 230–235. AAAI Press, 2007.

[BPS07] F. Baader, R. Peñaloza, and B. Suntisrivaraporn. Pinpointing in the Description Logic $\mathcal{EL}^+$. In J. Hertzberg, M. Beetz, and R. Englert, eds., *Proceedings of the 30th German Conference on Artificial Intelligence (KI2007)*, volume 4667 of *Lecture Notes in Artificial Intelligence*, pp. 52–67. Springer-Verlag, 2007.

[DG84] W. F. Dowling and J. H. Gallier. Linear-Time Algorithms for Testing the Satisfiability of Propositional Horn Formulae. *Journal of Logic Programming*, 3:267–284, 1984.

[Gan84] B. Ganter. Two Basic Algorithms in Concept Analysis. Technical Report Preprint-Nr. 831, Technische Hochschule Darmstadt, Darmstadt, Germany, 1984.

[GD86] J.-L. Guigues and V. Duquenne. Familles minimales d'implications informatives resultant d'un tableau de données binaries. *Mathématiques, Informatique et Sciences Humaines*, 95:5–18, 1986.

[GJ90] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Company, New York, NY, USA, 1990. ISBN 0716710455.

[GW99] B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, Berlin, Germany, 1999.

[HM01] V. Haarslev and R. Möller. RACER system description. In *Proceedings International Joint Conference on Automated Reasoning (IJCAR 2001)*. 2001.

[HPSvH03] 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.

[HTR06] M. Horridge, D. Tsarkov, and T. Redmond. Supporting Early Adoption of OWL 1.1 with Protege-OWL and FaCT++. In *Proceedings of the Sec-*

ond International Workshop OWL: Experiences and Directions (OWLED 2006). CEUR-WS, 2006.

[JPY88] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. On Generating All Maximal Independent Sets. *Information Processing Letters*, 27(3):119–123, 1988.

[KPS+06] 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.

[Mai80] D. Maier. Minimum Covers in Relational Database Model. *Journal of the ACM*, 27(4):664–674, 1980.

[Mai83] D. Maier. *The Theory of Relational Databases*. Computer Science Press, Maryland, 1983.

[OVSM04] D. Oberle, R. Volz, S. Staab, and B. Motik. An Extensible Ontology Software Environment. In S. Staab and R. Studer, eds., *Handbook on Ontologies*, International Handbooks on Information Systems, pp. 299–320. Springer-Verlag, 2004.

[Rud07] S. Rudolph. Some Notes on Pseudo-closed Sets. In S. O. Kuznetsov and S. Schmidt, eds., *Proceedings of the 5th International Conference on Formal Concept Analysis, (ICFCA 2007)*, volume 4390 of *Lecture Notes in Computer Science*, pp. 151–165. Springer-Verlag, 2007.

[SP04] E. Sirin and B. Parsia. Pellet: An OWL DL Reasoner. In *Proceedings of the 2004 International Workshop on Description Logics (DL2004)*, volume 104 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.

[TH06] D. Tsarkov and I. Horrocks. FaCT++ Description Logic Reasoner: System Description. In U. Furbach and N. Shankar, eds., *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pp. 292–297. Springer-Verlag, 2006.

[Val79a] L. G. Valiant. The Complexity of Computing the Permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.

[Val79b] L. G. Valiant. The Complexity of Enumeration and Reliability Problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.