TECHNISCHE
UNIVERSITÄT
DRESDEN

Dresden University of Technology
Institute for Theoretical Computer Science
Chair for Automata Theory

LTCS–Report

Exploring finite models in the Description Logic $\mathcal{EL}_{\mathrm{gfp}}$

Franz Baader, Felix Distel

LTCS-Report 08-05

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

Exploring finite models in the Description Logic $\mathcal{EL}_{\mathrm{gfp}}$

Franz Baader, Felix Distel

Inst. für Theoretische Informatik
TU Dresden
Germany
{baader,felix}@tcs.inf.tu-dresden.de

Abstract

In a previous ICFCA paper we have shown that, in the Description Logics $\mathcal{EL}$ and $\mathcal{EL}_{\mathrm{gfp}}$, the set of general concept inclusions holding in a finite model always has a finite basis. In this paper, we address the problem of how to compute this basis efficiently, by adapting methods from formal concept analysis.

1 Introduction

Description Logics (DLs) [4] are a well-investigated family of logic-based knowledge representation formalisms, which are employed in various application domains, such as natural language processing, configuration, databases, and biomedical ontologies, but their most notable success so far is the adoption of the DL-based language OWL [13] as standard ontology language for the semantic web. From the Description Logic point of view, an ontology is a finite set of general concept inclusion axioms (GCIs) of the form $C \sqsubseteq D$, where C, D are concepts defined using an appropriate concept description language. Such a concept description language allows one to construct complex concepts out of concept names (unary predicates, interpreted as sets) and roles (binary predicates, interpreted as binary relations) using certain concept constructors. Complex concepts are again interpreted as sets. To be more precise, given an interpretation of the concept and role names, the semantics of the concept constructors determines, for every complex concept, a unique set as the extension of this concept. The GCI $C \sqsubseteq D$ states that, in a model of the ontology, the extension of the concept C must be a subset of the extension of the concept D.

When defining a DL-based ontology, one must first decide on which vocabulary (i.e., concept and role names) to use, and then define appropriate constraints on the interpretation of this vocabulary using GCIs. The work described in this paper is motivated by the fact that coming up with the right GCIs by hand is usually not an easy task. Instead, we propose an approach where the knowledge engineer is required to provide us with a finite model, which should be seen as an abstraction or approximation of the application domain to be modeled. We then automatically generate a finite basis of the GCIs holding in the model, i.e., a finite set of GCIs that hold in this model and from which all GCIs holding in the model and expressible in the employed concept description language follow. The knowledge engineer can use the computed basis as a starting point for the definition of the ontology. She may want to weaken or even remove some of the GCIs if the chosen model was too restricted, and thus satisfies GCIs that actually do not hold in all intended models. As an example, assume that we want to define a family ontology, using the concept names $\mathsf{Male}, \mathsf{Father}, \mathsf{Female}, \mathsf{Mother}$, and the role name child. Consider a finite model with two families. The first family consists of John, Michelle, and Mackenzie, where John is male and a father (i.e., John belongs to the interpretation of the concept names Male and Father), Michelle is female and a mother, and Mackenzie is female and a child of both John and Michelle. The second family consists of Paul, Linda, and James, where Paul is male and a father, Linda is female and a mother, and James is male and a child of both Paul and Linda. In this model, the GCIs

$$\mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \exists \mathsf{child}.\top \quad \text{and} \quad \mathsf{Mother} \sqsubseteq \mathsf{Female} \sqcap \exists \mathsf{child}.\top$$

hold. The first one says that every father is male and has a child, and the second one says that every mother is female and has a child. If we had used a model consisting of only the first family, then we would have obtained the too specific GCIs $\mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \exists \mathsf{child}.\mathsf{Female}$ and $\mathsf{Mother} \sqsubseteq \mathsf{Female} \sqcap \exists \mathsf{child}.\mathsf{Female}$, where mothers and fathers always have female children.

For the approach sketched above to work, the set of GCIs holding in a finite model and expressible in the employed concept description language must have a *finite* basis. Using methods from formal concept analysis (FCA), we have shown in [7] that this is the case for the language $\mathcal{EL}$, which allows for the concept constructors $\top$ (top concept), $C \sqcap D$ (conjunction), and $\exists r.C$ (existential restriction). Though being quite inexpressive, $\mathcal{EL}$ has turned out to be very useful for representing biomedical ontologies such as SNOMED [17] and the Gene Ontology [20]. A major advantage of using an inexpressive DL like $\mathcal{EL}$ is that it allows for efficient reasoning procedures [3, 9]. Because of the nice algorithmic properties of $\mathcal{EL}$, the new OWL standard will contain a profile, called OWL 2 EL, that is based on $\mathcal{EL}$.

In [7], the existence of a finite basis is actually first shown for $\mathcal{EL}_{\mathrm{gfp}}$, which extends $\mathcal{EL}$ with cyclic concept definitions interpreted with greatest fixpoint semantics. The advantage of using $\mathcal{EL}_{\mathrm{gfp}}$ rather than $\mathcal{EL}$ is that, in $\mathcal{EL}_{\mathrm{gfp}}$, every set of objects (i.e., elements of the domain of a given finite model) always has a most specific concept describing these objects. Going from a set of objects to its most specific concept corresponds to the $\cdot'$ operator in FCA, which goes from a set of objects in a formal context to the set of all attributes that these objects have in common. The existence of most specific concepts in $\mathcal{EL}_{\text{gfp}}$ thus allowed us to employ methods from FCA. In a second step, we have shown in [7] that the $\mathcal{EL}_{\text{gfp}}$-basis can be turned into an $\mathcal{EL}$-basis by unraveling cyclic concept definitions up to a level determined by the cardinality of the given finite model.

In [7], we concentrated on showing the *existence* of a finite basis for $\mathcal{EL}_{\text{gfp}}$ and $\mathcal{EL}$. Of course, if the approach for automatically generating GCIs sketched above is to be used in practice, we also need to find efficient algorithms for computing such bases. This is the topic of the present paper. First, we show that the algorithm for computing an implication basis of a given formal context known from classical FCA can be adapted to our purposes. In contrast to the classical case, we cannot assume that all attributes of the context are known from the beginning. Instead, the set of attribute can be extended during the runtime of the algorithm. This is vital for obtaining an efficient algorithm. In a second step, we then extend this algorithm to an exploration algorithm. The advantage of this second algorithm is that it no longer requires the finite model to be completely represented in the computer from the beginning. As in the case of classical attribute exploration [11], the model is assumed to be "known" by an expert, who during the exploration process extends the represented part of the model in order to provide counterexamples to implication questions.

We concentrate on computing a finite $\mathcal{EL}_{\text{gfp}}$-basis since this basis can be turned into an $\mathcal{EL}$-basis as described in [7]. Due to the space limitation, we cannot give complete proofs of our results. They can be found in [6]. We also assume that the reader is familiar with the basic notion and results of formal concept analysis (FCA).

2 The logics $\mathcal{EL}$ and $\mathcal{EL}_{\text{gfp}}$

On the syntactic side Description Logic (DL) languages typically consist of a set of concept names $\mathcal{N}_C$ a set of role names $\mathcal{N}_r$ and certain constructors that can be used to create concept descriptions. Description logics use a model theoretic semantics. An interpretation $i = (\Delta_i, \cdot^i)$ consists of a set Δ_i, the domain of the interpretation, and a function $\cdot^i$ mapping concept names to subsets of Δ_i and role names to binary relations on Δ_i. Each constructor has its own well defined semantics. Using these semantics $\cdot^i$ can be extended to the set of all concept descriptions.

In this work we focus on the logics $\mathcal{EL}$ and $\mathcal{EL}_{\text{gfp}}$. $\mathcal{EL}$ only allows for conjunction and existential restrictions in order to construct new concepts. This means that

- All concept names $A \in \mathcal{N}_C$ are concept descriptions.
- If C and D are concept descriptions then $C \sqcap D$ is also a concept description.
- If C is a concept description and $r \in \mathcal{N}_r$ is a role name then $\exists r.C$ is also a concept description.

The semantics of conjunction and existential restrictions are defined according to the following rules

$$(C \sqcap D)^i = C^i \sqcap D^i \tag{1}$$
$$(\exists r.C)^i = \{x \in \Delta_i \mid \exists y \in C^i : (x, y) \in r^i\} \tag{2}$$

DL knowledge bases can contain statements about concepts, i. e. terminological knowledge, as well as statements about individuals, i. e. assertional knowledge. The part of the knowledge base that contains the terminological knowledge is called the TBox, whereas the part that contains the assertional knowledge is called the ABox.

In this work we do not make use of ABoxes and thus only introduce TBoxes. A TBox in our setting is a finite collection of statements of the form $A_k \equiv D_k$ where the A_k are concept names and the D_k are concept descriptions. No A_k may occur more than once on the left hand side of one of these statements. This means we do not allow for general TBoxes, which may contain statements other than equivalence statements. We do, however, allow cyclic TBoxes, i. e. a concept name can be used explicitly or implicitly in its own definition. We use *Greatest fixpoint semantics*, which are a common type of semantics for cyclic TBoxes in DL. A *primitive interpretation* i is a mapping that assigns a binary relation over Δ_i to every role name and a subset of Δ_i to every primitive concept name. An interpretation j is *based on* the primitive interpretation i if it coincides with i on all role names and primitive concept names. Note that j is uniquely defined by the set of interpretations of the defined concept names $(A_1^j, \ldots, A_n^j)$. We denote the set of all interpretations that are based on i by $\mathrm{Int}(i)$. The interpretations from $\mathrm{Int}(i)$ can be compared by the following ordering

$$j_1 \preccurlyeq_i j_2 \text{ iff } A_k^{j_1} \subseteq A_k^{j_2} \text{ for all } k, 1 \leq k \leq n.$$

For all every subset of $\mathrm{Int}\, i$ both least upper bounds and greatest lower bounds exist and coincide with pointwise union and pointwise intersection, respectively. Hence $\preccurlyeq_i$ is a complete lattice on $\mathrm{Int}\, i$. One can define a function f mapping every interpretation $j \in \mathrm{Int}\, i$ to the interpretation $f(j)$ defined by the tuple $(A_1^{f(j)}, \ldots, A_n^{f(j)}) := (D_1^j, \ldots, D_n^j)$. By the Knaster-Tarski-Fixpoint Theorem f must have a greatest fixpoint. We call this fixpoint the *gfp-model* of $\mathcal{T}$ corresponding to i. For matters of simplicity we will denote both the primitive interpretation and the corresponding gfp-model by the letter i (or $i_{\mathcal{T}}$ if it is not clear which TBox we are referring to).

2.1 The logic $\mathcal{EL}_{\mathrm{gfp}}$

To compute the extension of a defined concept $\mathcal{EL}$ with terminological cycles and gfp-semantics one has to look at the whole TBox – not just a single concept definition within the TBox. We therefore always need to state which TBox we are referring to. Also, in $\mathcal{EL}$ with terminological cycles and gfp-semantics there is no practical way to compare two concepts that have been defined in two different TBoxes. In such a situation one would always have to find a third TBox, which extends the first two TBoxes. This may become very tedious and can make notation complicated. So for purely pratical reasons we define an $\mathcal{EL}_{\mathrm{gfp}}$-concept description to be a pair consisting of TBox and a defined concept name occurring in this TBox.

Definition 1 ($\mathcal{EL}_{\mathrm{gfp}}$-concept description) *An* $\mathcal{EL}_{\mathrm{gfp}}$-concept description *is a tuple $(\mathcal{T}, A)$ where $\mathcal{T}$ is a TBox and A is a defined concept occurring on the left-hand side of a definition in $\mathcal{T}$.*

Let $i = (\Delta_i, \cdot^i)$ be a primitive interpretation. For every $\mathcal{EL}_{\mathrm{gfp}}$-concept description $C = (\mathcal{T}, A)$ we define the extension C^i to be the set $A^{i_\mathcal{T}}$. Here $i_\mathcal{T}$ is the gfp-model of $\mathcal{T}$ that is based on i. For $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions subsumption is defined as follows.

Definition 2 *Let $C = (\mathcal{T}_C, A_C)$, $D = (\mathcal{T}_D, A_D)$ be two $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions. We say that C is subsumed by D ($C \sqsubseteq D$) iff $C^{i_{\mathcal{T}_C}} \subseteq D^{i_{\mathcal{T}_D}}$ for all primitive interpretations i.*

Standard reasoning problems for $\mathcal{EL}$ with terminological cycles have been addressed by Baader [2]. Baader shows how instance and subsumption relations in $\mathcal{EL}_{\mathrm{gfp}}$ can be characterised using so called $\mathcal{EL}$-description graphs and simulations of such graphs.

Definition 3 ($\mathcal{EL}$-description graphs) *An $\mathcal{EL}$-description graph is a graph $\mathcal{G} = (V, E, L)$ where*

- *V is a set of nodes*
- *$E \subseteq V \times \mathcal{N}_r \times V$ is a set of directed edges labeled by role names*
- *$L : V \to \mathfrak{P}(\mathcal{N}_p)$ is a labeling function*

Let $C = (\mathcal{T}, A)$ be an $\mathcal{EL}_{\mathrm{gfp}}$-description. The corresponding $\mathcal{EL}$-description graph $\mathcal{G}_C$ is the graph $\mathcal{G} = (V_C, E_C, L_C)$ where

- *the vertices of $\mathcal{G}_C$ are the defined concepts of $\mathcal{T}$*
- *if B is a defined concept and*

$$B \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \exists r_1.B_1 \sqcap \exists r_l.B_l$$

its definition in $\mathcal{T}$, then

  - $L_C(B) = \{P_1, \ldots, P_m\}$, *and*
  - B *is the source of the edges* (B, r_1, B_1), $\ldots$, $(B, r_2, B_l) \in E_C$.

Conversely, every $\mathcal{EL}$-description graph can be transformed into an $\mathcal{EL}_{\mathrm{gfp}}$-TBox. An model i can also be transformed into an $\mathcal{EL}$-description graph $\mathcal{G}_i = (V_i, E_i, L_i)$.

- *The vertices of $\mathcal{G}_i$ are the elements of Δ_i.*
- $E_i = \{(x, r, y) \mid (x, y) \in r^i\}$
- $L_i(x) = \{P \in \mathcal{N}_{\mathrm{prim}} \mid x \in P^i\}$ *for all* $x \in \Delta_i$.

Baaders characterisations of instance and subsumption make use of simulations between $\mathcal{EL}$-description graphs. These are defined as follows.

Definition 4 (Simulation) *Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two $\mathcal{EL}$-description graphs. The binary relation $Z \subseteq V_1 \times V_2$ is a* simulation *from $\mathcal{G}_1$ to $\mathcal{G}_2$ iff*

1. $(v_1, v_2) \in Z$ *implies* $L_1(v_1) \subseteq L_2(v_2)$, *and*
2. *if* $(v_1, v_2) \in Z$ *and* $(v_1, r, v_1') \in E_1$, *then there exists a node* $v_2' \in V_2$ *such that* $(v_1', v_2') \in Z$ *and* $(v_2, r, v_2') \in E_2$.

We write $Z : \mathcal{G}_1 \rightleftarrows \mathcal{G}_2$ to express that Z is a simulation from $\mathcal{G}_1$ to $\mathcal{G}_2$.

Then instance relations in a given model can be characterised as follows.

Proposition 1 *Let i be a model. Then the following statements are equivalent for any $\mathcal{EL}_{\mathrm{gfp}}$-description $C = (\mathcal{T}, A)$ and every $x \in \Delta_i$.*

- $x \in \mathcal{T}^i$
- *There is a simulation $Z : \mathcal{G}_C \rightleftarrows \mathcal{G}_i$ such that $(A, x) \in Z$.*

This result eventually leads to the following theorem which characterises subsumption.

Theorem 1 *Let $C = (\mathcal{T}_C, A_C)$ and $D = (\mathcal{T}_D, A_D)$ be two $\mathcal{EL}_{\mathrm{gfp}}$-descriptions. Then the following two statements are equivalent.*

- $C \sqsubseteq D$
- *There is a simulation $Z : \mathcal{G}_D \overrightarrow{\sim} \mathcal{G}_C$ such that $(A_D, A_C) \in Z$.*

It is easy to see that acyclic $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions (i.e., ones where the TBox component is acyclic) correspond exactly to $\mathcal{EL}$-concept descriptions. This shows that $\mathcal{EL}$ can indeed be seen as a sublanguage of $\mathcal{EL}_{\mathrm{gfp}}$. In the following, we will not distinguish an acyclic $\mathcal{EL}_{\mathrm{gfp}}$-concept description from its equivalent $\mathcal{EL}$-concept description.

2.2 GCIs and Model-based Most Specific Concepts

We formally define what we mean when we speak of a basis for the GCIs of a model i. This means, we first need to define what we mean by a GCI.

Definition 5 (GCIs) *A* GCI *is a pair (C, D) of concept descriptions C and D. If (C, D) is a GCI we also write $C \to D$. We say that a GCI $C \to D$* holds in the model *$i = (\Delta_i, i)$ iff $C^i \subseteq D^i$ holds.*

Let $\mathcal{B}$ be a set of $\mathcal{L}$-GCIs and $C \to D$ a GCI. If $C \to D$ holds in all models $i = (\Delta_i, i)$, in which all GCIs from $\mathcal{B}$ hold, then we say that $C \to D$ follows from *$\mathcal{B}$* (in *$(\mathcal{L}, \mathcal{I})$*).

In DL GCIs are commonly written using $\sqsubseteq$ instead of $\to$. We prefer the $\to$-notation whenever we are dealing with GCIs that have not yet been confirmed by an expert or that are not part of the TBox of some knowledge base. This is on the one hand to emphasize the connection with the implications from classical FCA and on the other hand to avoid confusion with subsumption which is also denoted by $\sqsubseteq$.

Definition 6 (Basis) *For a given model i we say that a set of GCIs $\mathcal{B}$ is a* basis *for the GCIs holding in i if $\mathcal{B}$ is*

- sound *for i, i.e., it contains only GCIs holding in i, and*
- complete *for i, i.e., any GCI that holds in i follows from $\mathcal{B}$.*

In classical FCA there exists the so-called Duquenne-Guigues basis. The Duquenne-Guigues basis has yet another desirable property beyond being sound and complete: it has minimal cardinality. Certain ideas that are used in the construction of the Duquenne-Guigues basis are useful in our setting as well. The major ideas are:

1. In classical FCA it suffices to consider implications of the form $A \to A''$ (Given an implication $A \to B$ that holds in a Formal Context $\mathbb{K}$. Then $A \to B$ follows from $A \to A''$.)

2. It suffices to consider implications whose left-hand-sides are pseudo-closed.

Neither the notion of the $\cdot'$-operators nor the notion of pseudo-closedness exist in Description logics. Therefore we need to come up with something that exhibits similar properties in a DL setting. For the $\cdot'$ operator these are model-based most specific concept that we first introduced in [7]. In classical FCA A' is the set of attributes common to the objects in A. This is equivalent to defining $A' = B_{\max}$, where $B_{\max}$ is the greatest subset of M such that $A \subseteq B'_{\max}$. This motivates the following definition.

Definition 7 (Model-based most specific concepts) *Let $i \in \mathcal{I}$ be an interpretation and X a set $X \subseteq \Delta_i$. Let $C \in \mathcal{L}$ be the least concept description such that*

$$X \subseteq C^i. \tag{3}$$

By least concept description *we mean that every other concept description $\bar{C}$ which satisfies (3) also satisfies $C \sqsubseteq \bar{C}$. Then C is called a model-based most specific concept of X in i. Observe that model-based most specific concept are unique, up to equivalence. Therefore it makes sense to denote the model-based most specific concept by X^i (existence provided).*

The notation X^i may seem confusing at first glance, because it can be confused with the extension of a concept description C^i. Again, this is a reference to classical FCA, where there are two different operators, both denoted by $\cdot'$. In reference to classical FCA we call a concept description C *an intent* if is equivalent to a most specific concept $X^i \equiv C$.

Unfortunately model-based most specific concept need not exist for most description logics, in particular where there are cycles in the model. However, in the case of $\mathcal{EL}_{\text{gfp}}$, $\mathcal{FL}_{\text{gfp}}$ and $\mathcal{FLE}_{\text{gfp}}$ they do exist. This has been shown in [7]. We will therefore concentrate mainly on these logics. The following Lemma presents some simple rules for model-based most specific concepts, none of which are difficult to prove. Proves can be found in [5].

Lemma 1 *Let $\mathcal{L}$ be a language for which X^i exists for every $X \subseteq \Delta_i$ and every $i \in \mathcal{I}$. Let $i \in \mathcal{I}$ be an interpretation, $X, Y \in \Delta_i$ sets of objects and C, D be concept descriptions. Then the following statements hold*

1. $X \subseteq Y \Rightarrow X^i \sqsubseteq Y^i$
2. $C \sqsubseteq D \Rightarrow C^i \subseteq D^i$
3. $X \subseteq X^{ii}$
4. $C^{ii} \sqsubseteq C$
5. $X^i \equiv X^{iii}$
6. $C^i = C^{iii}$
7. $X \subseteq C^i \Leftrightarrow X^i \sqsubseteq C$.

Our main objective when defining model-based most specific concepts was to find some operator that exhibits a property similar to 1. More precisely, given a concept description C we are looking for a concept description D such that all GCIs of the form $C \to E$ that hold in i follow from $C \to D$. It turns out that $D = C^{ii}$ does the job.

Lemma 2 *Let $\mathcal{L}$ be a language and $i \in \mathcal{I}$ be an interpretation such that X^i exists for every $X \subseteq \Delta_i$. Let C and D be two concept descriptions. Then*

- *$C \to C^{ii}$ holds in i, and*
- *if $C \to D$ holds in i, then $C \to D$ follows from $\{C \to C^{ii}\}$.*

2.3 Two finite GCI bases for $\mathcal{EL}_{\mathrm{gfp}}$

Recall that in our framework, a basis is defined to be a set of GCIs that is sound and complete for a model. We do not make any requirements concerning irredundancy of the set of GCIs. But still it is desirable to find bases that are irredundant, or at least finite (A finite basis will always contain an irredundant subset). Just like the Duquenne-Guigues basis is a compact and elegant way to sum up the implicational knowledge for a given Formal Context, such an irredundant basis is a smart way to sum up the implicational knowledge for a given model.

The problem of proving the existence of finite bases for arbitrary models in $\mathcal{EL}_{\mathrm{gfp}}$ and $\mathcal{EL}$ has been treated in [7]. An important step in the proof for $\mathcal{EL}_{\mathrm{gfp}}$ has been to show that the set

$$\mathcal{B}_{\mathrm{acyclic}} = \{A \to A^{ii} \mid A \text{ acyclic } \mathcal{EL}_{\mathrm{gfp}}\text{-concept description}\}$$

is sound and complete. An acyclic $\mathcal{EL}_{\mathrm{gfp}}$-concept description is an $\mathcal{EL}_{\mathrm{gfp}}$-concept description whose description graph is a tree. In other words an $\mathcal{EL}_{\mathrm{gfp}}$-concept description is acyclic iff it uses only the expressivity of standard $\mathcal{EL}$. $\mathcal{B}_{\mathrm{acyclic}}$ itself is not finite, because there are infinitely many acyclic concept descriptions. But it can nevertheless be used to construct finite bases. The idea is, that one can do structural induction over the left hand sides in $\mathcal{B}_{\mathrm{acyclic}}$ – something that cannot be done over cyclic structures. The finite basis presented in [7] looks like this.

$$\begin{aligned}\mathcal{B}_{[7]} := &\{P \to P^{ii} \mid P \in \mathcal{N}_p \cup \{\top\}\} \\ &\cup \{\exists r.C \to (\exists r.C)^{ii} \mid r \in \mathcal{N}_r, C \in \mathcal{C}\} \\ &\cup \{C_1 \sqcap C_2 \to (C_1 \sqcap C_2)^{ii} \mid C_1, C_2 \in \mathcal{C}\},\end{aligned}$$

where $\mathcal{C}$ is a set of $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions such that for every $\mathcal{EL}_{\mathrm{gfp}}$-concept description D there is exactly one $C \in \mathcal{C}$ such that $C^i = D^i$. In $\mathcal{B}_{[7]}$ primitive concept names, conjunction and existential restrictions are dealt with independently (We have three classes of premises: Those of the form P, $P \in \mathcal{N}_p$, those of the form $\exists r.C$, $r \in \mathcal{N}_r$, $C \in \mathcal{C}$, and those of the form $C_1 \sqcap C_2$, $C_1, C_2 \in \mathcal{C}$). A similar basis $\mathcal{B}_i$ is obtained by merging these three classes of concept descriptions into one.

Definition 8 *Let i be a finite $\mathcal{EL}_{\mathrm{gfp}}$-model. The sets M_i, Λ_i are defined as*

$$M_i := N_c \cup \{\exists r.X^i \mid r \in N_r \text{ and } X \subseteq \Delta_i\} \quad \text{and} \quad \Lambda_i := \{\textstyle\bigsqcap U \mid U \subseteq M_i\}.$$

Furthermore define $\mathcal{B}_i := \{C \to C^{ii} \mid C \in \Lambda_i\}$.

Since N_c, N_r, and Δ_i are finite, M_i and Λ_i are finite as well. Thus, the basis $\mathcal{B}_i$ is finite as well.

Concept Descriptions from Λ_i are Concept Descriptions where every subdescription that is "behind" an existential quantification is an intent. They can be written as the conjunction of primitive concept names and descriptions of the form $\exists r.X^i$ where r is a role name and X^i, $X \subseteq \Delta_i$, is a concept intent for i. Like in the proof for $\mathcal{B}_{[7]}$ completeness of $\mathcal{B}_i$ can be shown via structural induction. Given an acyclic concept description A the description graph for A is a tree. The leaves of this tree correspond to a conjunction of primitive concept names, i. e. to some concept description $C \in \Lambda_i$. Thus $C \to C^{ii} \in \mathcal{B}_i$. Thus the subdescriptions of A that correspond to leaves in the description graph can be replaced by their concept intents. In a next step the same thing can be done to all vertices that have only leaves as successors, and so on ...

Lemma 3 *$\mathcal{B}_i$ is a finite basis for i.*

Proof: $\mathcal{B}_i$ is finite because in any finite model i there can be only finitely many (up to equivalence) descriptions X^i where $X \subseteq \Delta_i$. It is sound, because for every model i all GCIs of the form $C \to C^{ii}$ hold in i.

Completeness: It suffices to prove that every GCI $A \to A^{ii}$ where A is acyclic follows from $\mathcal{B}_i$. This can be done via induction over the structure of A. Since the proof is very straightforward we only present the inductive case where $A \equiv \exists r.B$ where $r \in \mathcal{N}_r$ and where we already know that $B \rightarrow B^{ii}$ follows from $\mathcal{B}_i$. This implies that $A \rightarrow \exists r.B^{ii}$ follows from $\mathcal{B}_i$ since obviously $\exists r.B \rightarrow \exists r.B^{ii}$ follows from $B \rightarrow B^{ii}$. But $\exists r.B^{ii} \in \Lambda_i$ and thus $\exists r.B^{ii} \rightarrow (\exists r.B^{ii})^{ii} \in B_i$. One can prove that $A^{ii} \equiv (\exists r.B^{ii})^{ii}$ and therefore $A \rightarrow A^{ii}$ follows from $\mathcal{B}_i$. The case where $A \equiv B_1 \sqcap B_2$, where $B_1 \rightarrow B_1^{ii}$ and $B_2 \rightarrow B_2^{ii}$ follow from $\mathcal{B}_i$ can be done accordingly. This proves completeness of $\mathcal{B}_i$. Since $\mathcal{B}_i$ is sound and complete it is a basis for i. $\square$

It is not hard to see that the set Λ_i of premises is closed with respect to conjunction $\sqcap$. Furthermore, it is possible to prove that all concept intents X^i are elements of Λ_i.

Lemma 4 *All concept intents X^i, where $X \subseteq \Delta_i$, are contained in Λ_i.*

Proof: Every $\mathcal{EL}_{\text{gfp}}$-concept description, and thus also X^i can be unraveled as follows:

$$X^i \equiv \prod_{P\in\mathcal{P}} P \sqcap \prod_{1\leq k\leq n} \exists r_k.D_k,$$

for some set $\mathcal{P} \subseteq \mathcal{N}_p$, some natural number n, some role names $r_k \in \mathcal{N}_r$ and $\mathcal{EL}_{\text{gfp}}$-concept descriptions D_k, $k \in \{1,\ldots,n\}$. For every $k \in \{1,\ldots,n\}$, we know that $D_k^{ii} \sqsubseteq D_k$ and thus also $\exists r_k.D_k^{ii} \sqsubseteq \exists r_k.D_k$. This yields

$$X^i \equiv \prod_{P\in\mathcal{P}} P \sqcap \prod_{1\leq k\leq n} \exists r_k.D_k \sqsupseteq \prod_{P\in\mathcal{P}} P \sqcap \prod_{1\leq k\leq n} \exists r_k.D_k^{ii}.$$

Let x be an element of X^{ii}. Then by the definition of the semantics of $\sqcap$ we get

$$\begin{aligned} x \in X^{ii} &\Leftrightarrow \forall P \in \mathcal{P} : x \in P^i \text{ and } \forall k \in \{1,\ldots,n\} : x \in (\exists r_k.D_k)^i \\ &\Leftrightarrow \forall P \in \mathcal{P} : x \in P^i \text{ and } \forall k \in \{1,\ldots,n\} : \exists y \in D_k^i : (x,y) \in r^j \\ &\Leftrightarrow \forall P \in \mathcal{P} : x \in P^i \text{ and } \forall k \in \{1,\ldots,n\} : \exists y \in D_k^{iii} : (x,y) \in r^j \\ &\Leftrightarrow \forall P \in \mathcal{P} : x \in P^i \text{ and } \forall k \in \{1,\ldots,n\} : x \in (\exists r_k.D_k^{ii})^i \\ &\Leftrightarrow x \in \left(\prod_{P\in\mathcal{P}} P \sqcap \prod_{1\leq r\leq n} \exists r_k.D_k^{ii} \right)^i \end{aligned}$$

So we have shown that

$$X \subseteq X^{ii} = \left(\prod_{P\in\mathcal{P}} P \sqcap \prod_{1\leq r\leq n} \exists r_k.D_k^{ii} \right)^i.$$

Since X^i is the most specific concept for X it follows that

$$X^i \sqsubseteq \left(\prod_{P\in\mathcal{P}} P \sqcap \prod_{1\leq r\leq n} \exists r_k.D_k^{ii} \right).$$

Together with the above this yields

$$X^i \equiv \left(\prod_{P \in \mathcal{P}} P \sqcap \prod_{1 \leq r \leq n} \exists r_k.D_k^{ii} \right).$$

and thus $X^i \in \Lambda_i$. $\square$

Both bases $\mathcal{B}_{[7]}$ and $\mathcal{B}_i$ have the disadvantage that they are not easy to compute. In order to compute $\mathcal{B}_{[7]}$ one needs to know all concept extents D^i of the underlying model i. On the other hand it is necessary to know all concept intents X^i in order to compute $\mathcal{B}_i$. Both the set of intents and extents may be exponentially large – in the worst case one would have to compute the closures or model-mscs for all subsets of Δ_i. This is particularly undesirable if we want to extend the algorithms to a knowledge exploration process where the underlying model may change in every step.

3 Formal Concept Analysis

In this section we briefly recall the most important definitions from Formal Concept Analysis.

Definition 9 (Formal Context) *A* formal context $\mathbb{K} = (G, M, I)$ *is a data structure consisting of a set of objects G, a set of attributes M and a binary relation $I \subseteq G \times M$. If $(g, m) \in I$ then we say that the object g has the attribute m.*

Definition 10 (The $\cdot'$ operators) *Let $A \subseteq G$ be a set of objects. Then define*

$$A' = \{m \in M \mid \forall g \in A : (g, m) \in I\}.$$

We call A' the intent of A. *Let $B \subseteq M$ be a set of attributes. Define*

$$B' = \{g \in G \mid \forall m \in B : (g, m) \in I\}.$$

B' is called the extent of B.

A set of attributes A for which $A = A''$ is called a *(concept) intent*. Likewise a set of objects B for which $B = B''$ is called a *(concept) extent*.

Definition 11 (implications) *An implication is a pair (A, B) of sets of attributes $A, B \subseteq M$. For better readability we write $A \to B$.*

We say that an implication $A \to B$ holds in a context $\mathbb{K}$ iff $A' \subseteq B'$.

We say that a set of attributes $A \subseteq M$ respects an implication $B \to C$ iff $B \nsubseteq A$ or $C \subseteq A$. If $\mathcal{L}$ is a set of implications and $A \subseteq M$ respects all implications from $\mathcal{L}$ then we say the A is *closed with respect to* $\mathcal{L}$.

Definition 12 *An implication $A \to B$* follows *from a set of implications $\mathcal{B}$ iff $A \to B$ holds in every context $\mathbb{K}$ in which $\mathcal{B}$ holds.*

This definition is equivalent to the following: An implication $A \to B$ follows from a set of implications $\mathcal{B}$ iff every set of attributes $C \subseteq M$ which is closed with respect to $\mathcal{B}$ also respects $A \to B$.

We say that a set of implications $\mathcal{B}$ is an *implication basis for the context* $\mathbb{K}$ if $\mathcal{B}$ is a sound and complete set of implications for $\mathbb{K}$.

Definition 13 (pseudo-intent) *Let $\mathbb{K} = (G, M, I)$ be a formal context. A set of attributes $P \subseteq M$ is called a* pseudo-intent *iff P is not an intent and for every pseudo-intent $Q \subsetneq P$ it holds that $Q'' \subseteq P$.*

This recursive definition may appear a bit awkward at first glance. But it is not hard to see that pseudo-intents are well-defined. Obviously for the empty set it can be decided whether it is a pseudo-intent or not, since it does not have any strict subsets. If it can be decided for all subsets of a set P whether they are pseudo-intents or not, then it can also be decided for P. So well-definedness of pseudo-intents follows by induction.

Definition 14 (Duquenne-Guigues basis) *The set of implications*

$$\mathcal{B} = \{P \to P'' \mid P \text{ pseudo-intent of } \mathbb{K}\}$$

is called the Duquenne-Guigues basis of $\mathbb{K}$*. It is*

- *sound for $\mathbb{K}$, i. e. all implications from $\mathcal{B}$ hold in $\mathbb{K}$, and*
- *complete, i. e. all implications that hold in $\mathbb{K}$ follow from $\mathcal{B}$, and*
- *it has minimal cardinality among all sets of implications that are complete for $\mathbb{K}$.*

The recursive definition of pseudo-intents may be more common, but there is an equivalent, alternative definition that does not make use of recursion. It is based on the notion of quasi-closedness. Let $\mathbb{K} = (G, M, I)$ be a formal context. A set of attributes $Q \subseteq M$ is called a *quasi-closed* iff for subset $R \subseteq Q$ either $R'' \subseteq Q$ or $R'' = Q''$. The following alternative definition of pseudo-intents has been introduced by Bernhard Ganter in [10].

Theorem 2 (alternative definition of pseudo-intents) *Let $\mathbb{K} = (G, M, I)$ be a formal context. A set of attributes $P \subseteq M$ is a pseudo-intent iff*

- *P is not an intent, and*
- *P is quasi-closed, and*
- *P is minimal among all quasi-closed sets Q with $Q'' = P''$, i. e. there is no quasi-closed set Q such that $Q \subsetneq P$ and $Q'' = P''$.*

In [12] Ganter et al. present an algorithm for computing all intents and pseudo-intents of a given formal context. The advantage of this algorithm is that it works with polynomial delay. This means that after finding some intent or pseudo-intent the time that it takes to compute the next intent or pseudo-intent is bounded polynomially in the size of the input. The Next-Closure Algorithm uses a so-called lectic order over the power set of M. Provided an order $\leq$ over M itself it is defined as follows

$$A < B :\Leftrightarrow \exists m \in B \setminus A : \forall n < m : (n \in A \Leftrightarrow n \in B),$$

i. e. the least element in which A and B differ is contained in B. The Next-Closure Algorithm will first come up with the lectically smallest pseudo-intent and then produce all intents and pseudo-intents in the lectic order. At the core of the Next-Closure Algorithm is a method that given a set of implications $\mathcal{L}$ and a set of attributes A computes the lectically next set B that is closed with respect to $\mathcal{L}$. This means that B has the following properties.

- If $C \to D \in \mathcal{L}$ then $C \subseteq B$ implies $D \subseteq B$.
- B is lectically greater than A
- Among all sets with these properties B is the lectically smallest.

The attribute exploration technique of FCA is based upon the Next-Closure Algorithm. The exploration starts with an incomplete context and computes the lectically first pseudo-intent P_0. Then it asks the expert whether $P_0 \to P_0''$ holds in the "real world". If the expert accepts the implication then it is added to the set of implications $\mathcal{B}$. If she rejects it then she is asked to provide a counter-example which is then added to the context. And here a nice property of the Duquenne-Guigues basis comes into play: If $P_0, \ldots, P_n$ are the lectically first n intents and pseudo-intents in $\mathbb{K}$ then $P_0, \ldots, P_n$ are also the lectically first n intents and pseudo-intents in $\bar{\mathbb{K}}$, where $\bar{\mathbb{K}}$ is obtained from $\mathbb{K}$ by adding a counter-example. Therefore, one does not need to start from scratch but can keep the previously obtained implications. When the expert interaction is finished the Next-Closure Algorithm is used to compute the lectically next pseudo-intent P. Then the implication $P \to P''$ is presented to the expert, and so on .... The algorithm terminates when all implications either follow from $\mathcal{B}$ or are refuted by $\mathbb{K}$.

3.1 Dealing with growing sets of attributes

In classical attribute exploration, the exploration algorithm generates implications that are presented to an expert (usually human). If the expert refutes the implication then she is asked to provide a counter-example. This counter-example is then added to the set of objects. Thus classical attribute exploration is a setting, in which the set of *objects* can grow, while the set of attributes remains unchanged.

In FCA there is a dual method for attribute exploration, namely *object exploration*, i. e. the set of attributes extended, while the set of objects remains unchanged. Instead of creating implications between sets of attributes as questions to the expert, implications between sets of objects are computed.

An exploration algorithm where not only the set of objects but also the set of attributes are allowed to grow is *Concept Exploration* [19]. Here the questions that are being asked are not of the form "Does attribute set A imply attribute set B?" but of the form "Is s a subconcept of t?". Concept Exploration can be used to complete both the set of objects and the set of attributes in the sense that the final context contains all relevant concepts. Concept Exploration has several known problems, for example it need not terminate.

Both object exploration and concept exploration are not what we want. We are interested in implications between attribute sets, which neither object exploration nor concept exploration can create. Also, we are not aiming at some sort of completeness with respect to the attributes (At a later stage we shall be aiming at completeness with respect to the objects, though). What we want is an extension to the NextClosure algorithm for computing intents and pseudointents which allows a little more flexibility with respect to adding attributes manually.

Ideally, during an exploration process or during the computation of the Duquenne-Guigues-Basis the expert obtains new knowledge about the field. This new knowledge might point her towards new attributes that she wishes to include. In a classical setting she would then have to start the exploration process from scratch. So as a first step we restrict ourselves to a setting where the *set of objects* is fixed while the set of attributes can grow. We start with a context $\mathbb{K}_0 = (G, M_0, I_0)$. In each step, new attributes are added and a new context $\mathbb{K}_k = (G, M_k, I_k)$ is obtained. We require that $M_{k-1} \subseteq M_k$ for all $k \geq 1$ and that I_k agrees with I_{k-1} on the old attribute set, i. e. for all $g \in G$ and for all $m \in M_{k-1}$ we have $(g, m) \in I_k$ iff $(g, m) \in I_{k-1}$. We will sometimes use the $\cdot'$-operators. To make clear which context we are referring to we place a index after the operators, i. e. A''^{k} for A'' computed in the context $\mathbb{K}_k$. We furthermore need an order on the set of attributes. At one point we will compute the next closure for a given set of attributes. To determine what the *next* closure is, we need an order on the set of attributes. The only restriction we make here is that attributes that are obtained in a later step have higher order. For example if $a \in M_k \setminus M_{k-1}$ and

$b \in M_{k-1}$ then we require that $b \leq a$. This order on the set of attributes gives rise to a lectic order on the power set of the attribute set, which is defined as follows. Given two sets $A \subseteq M_k$ and $B \subseteq M_k$ we say that $A \leq B$ iff the largest attribute which distinguishes A and B belongs to B. Given a set of attributes A and a set of implications $\mathcal{C}$ the next closure can be computed just like in the classical Next-Closure Algorithm. An outline of the algorithm is presented as Algorithm 1.

Algorithm 1 Algorithm for computing an implication basis that allows adding attributes

Input: $\mathbb{K}_0 = (G, M_0, I_0)$
$\Pi_0 = \emptyset$, $P_0 = \emptyset$, $k = 0$
while $P_k \neq \mathsf{null}$ **do**
  $\Pi_{k+1} = \Pi_k \cup \{P_k\}$
  $k = k + 1$
  Input: $\mathbb{K}_k = (G, M_k, I_k)$
  if $M_k = M_{k-1} = P_k$ **then**
    $P_k = \mathsf{null}$
  else
    $P_k =$ lectically smallest subset of M_k that is

- closed with respect to $\{P_j \to P_j''^{k} \mid P_j \in \Pi_k\}$, and
- lectically larger than P_{k-1}.

  end if
end while

Whether the algorithm terminates depends on whether there is some $n \in \mathbb{N}$ such that for all steps $k > n$ no new attributes are added and whether only finitely many attributes are added in each step. Then there is a final set of attributes M_n which is finite. Termination results from the fact that M_n has only a finite number of subsets. Since every $P_k \subseteq M_n$ is lectically greater that P_{k-1} the full set $P = M_n$ must be reached at some point.

Note that the algorithm does not compute implications in the first place. What it does is compute the premises for the implication. The final set of implications will be

$$\mathcal{B}_n = \{P_k \to P_k''^{n} \mid 0 \leq k \leq n\},$$

where n is such that the algorithm terminates for $k = n$. Note that the sets P_k need not be pseudo-intents of the final context $\mathbb{K}_n$. The notation P stands for *premise*, not for pseudo-intent. But $\mathcal{B}$ is still sound since all implications of the form $C \to C''$ hold in $\mathbb{K}_n$. Completeness of $\mathcal{B}_n$ is shown in the next lemma.

Lemma 5 *Assume that Algorithm 1 terminates after the n-th step. Let Q be a set of attributes that is quasi-closed in $\mathbb{K}_n$. Then there is some $P_k \in \Pi_n$ such that $P_k \subseteq Q$, $P_k^{\prime\prime_n} = Q^{\prime\prime_n}$.*

We know that $\{Q \to Q^{\prime\prime_n} \mid Q \text{ is a quasi-closed in } \mathbb{K}_n\}$ *is complete (every pseudo-intent is quasi-closed). Thus $\mathcal{B}_n = \{P_k \to P_k^{\prime\prime_n} \mid P_k \in \Pi_n\}$ must also be complete for $\mathbb{K}_n$.*

Proof: Let m be the natural number for which P_m is the lectically largest set from Π_n that is lectically smaller than Q. We know that P_{m+1} is a subset of M_{m+1}. P_{m+1} is lectically larger or equal to Q. Thus in particular the largest element of P_{m+1} is greater or equal to the largest element of Q (from the definition of the lectic order). We have defined the order on the attributes in such a way that M_{m+1} must contain all attributes of M_n that are smaller than the largest element of P_{m+1} and therefore $Q \subseteq M_{m+1}$.

Assume that Q is closed with respect to $\{P_k \to P_k^{\prime\prime_m} \mid k \leq m\}$. Then in the m-th iteration of the while-loop of Algorithm 1 we obtain $P_{m+1} = Q$ and thus $Q \in \Pi_n$ and $Q \to Q^{\prime\prime_n} \in \mathcal{B}_n$.

If Q is not closed with respect to $\{P_k \to P_k^{\prime\prime_m} \mid k \leq m\}$, then there must be some natural number $j \leq m$ such that Q does not respect $P_j \to P_j^{\prime\prime_m}$, i.e. $P_j \subseteq Q$ but $P_j^{\prime\prime_m} \not\subseteq Q$. It holds that

$$P_j^{\prime_m} = \{o \in G \mid \forall a \in P_j : (o,a) \in I_m\} = \{o \in G \mid \forall a \in P_j : (o,a) \in I_n\} = P_j^{\prime_n},$$

because the requirements that we have imposed on I_m. Then

$$\begin{aligned} P_j^{\prime\prime_m} = \{a \in M_m \mid \forall o \in P_j^{\prime_m} : (o,a) \in I_m\} &= \{a \in M_m \mid \forall o \in P_j^{\prime_n} : (o,a) \in I_n\} \\ &\subseteq \{a \in M_n \mid \forall o \in P_j^{n} : (o,a) \in I_n\} = P_j^{\prime\prime_n}. \end{aligned}$$

Since $P_j^{\prime\prime_m} \subseteq P_j^{\prime\prime_n}$ it follows that Q does not respect $P_j \to P_j^{\prime\prime_n}$ either. Quasi-closedness of Q implies that $P_j^{\prime\prime_n} = Q^{\prime\prime_n}$. □

Note that in every step k we must keep P_k in the set of premises, even if P_k is an intent in $\mathbb{K}_k$. This is because it might happen that $P_k = P_k^{\prime\prime_k}$ but $P_k \neq P_k^{\prime\prime_n}$ because the attributes in $P_k^{\prime\prime_n} \setminus P_k^{\prime\prime_k}$ have only been added at a later point. Thus P_k need not be an intent in $\mathbb{K}_n$ anymore.

So now we have an algorithm that computes a sound and complete set of implications for a formal context, where the full set of attributes is not known from the beginning. More attributes can be added, after a premise has been computed. No requirements have been made as to the nature of the attributes. They can be whatever the expert deems interesting. Alternatively, they can be computed automatically in some way, as will be the case in our method.

Our method will be working with contexts, where the attributes are concept description. There may be dependencies between concept descriptions that hold in every model i. For example the concept description $\exists r.P$ is subsumed by the descriptions $\exists r.\top$. Thus the GCI $\exists r.P \to \exists r.\top$ holds in every model and does not provide any information about the specific model i. Thus one would not want to include it in a potential basis. For that reason one might want to include such GCIs as background knowledge.

The NextClosure Algorithm in its standard form cannot handle background knowledge in the form of implications. Fortunately, there is Gerd Stumme's approach for handling background knowledge [18]. It turns out that background knowledge can be included in a straightforward manner. Stumme defines $\mathcal{S}$-pseudo-intents as follows.

Definition 15 *Let (G, M, I) be a context and $\mathcal{S}$ a set of implications holding in (G, M, I). A set $P \subseteq M$ is called $\mathcal{S}$-pseudo-intent, if P respects all implications from $\mathcal{S}$ and if for every $\mathcal{S}$-pseudo-intent Q with $Q \subsetneq P$ it holds that $Q'' \subseteq P$.*

In this definition and the rest of the work we require that all implications from $\mathcal{S}$ hold in the underlying context (G, M, I).

In Stumme's work it is shown that then the set $\mathcal{B}_{\mathcal{S}} = \{P \to P'' \mid P \text{ is } \mathcal{S}\text{-pseudo-intent in } \mathbb{K}\}$ satisfies the following properties.

- $\mathcal{B}_{\mathcal{S}} \cup \mathcal{S}$ is a sound and complete set of implications for $\mathbb{K}$, and
- $\mathcal{B}_{\mathcal{S}}$ has minimal cardinality among all sets $\mathcal{B}$ for which $\mathcal{B} \cup \mathcal{S}$ is a sound and complete set of implications for $\mathbb{K}$.

Furthermore Stumme presents a slightly modified version of the Next-Closure Algorithm which can be used to find all $\mathcal{S}$-pseudo-intents of a context.

As with standard pseudo-intents there is an alternative, non-recursive characterization for $\mathcal{S}$-pseudo-intents that makes use of the notion of quasi-closedness. This can be shown along the lines of the proof for standard pseudo-intents. Since it has not been included in [18] we present it here.

Lemma 6 *Let (G, M, I) be a finite context and $\mathcal{S}$ a set of implications holding in (G, M, I). A set $P \subseteq M$ is an $\mathcal{S}$-pseudo-intent iff*

- *P respects all implications from $\mathcal{S}$, and*
- *P is quasi-closed, and*
- *P is minimal among all quasi-closed sets with the same intent that respect $\mathcal{S}$.*

Proof: We first prove the *only if* direction We show that every $\mathcal{S}$-pseudo-intent P satisfies all three properties.

P respects all implications from $\mathcal{S}$: This is trivial since $\mathcal{S}$-pseudo-intents by definition have this property.

Quasi-Closedness: Assume that P is not quasi-closed. Then there exists some set $Q \subsetneq P$, $Q'' \nsubseteq P$ and $Q'' \subsetneq P''$. Without loss of generality assume that Q is maximal among all sets with these properties. Since P is an $\mathcal{S}$-pseudo-intent, Q cannot be a pseudo-intent. Hence there must be some $\mathcal{S}$-pseudo-intent $R \subsetneq Q$, $R'' \nsubseteq Q$. $R \subsetneq Q \subsetneq P$ and the fact that P is an $\mathcal{S}$-pseudo-intent imply $R'' \subseteq P$.

Consider $T = R'' \cup Q$. We prove that T satisfies $T \subseteq P$, $T'' \nsubseteq P$ and $T'' \subsetneq P''$, as well as $Q \subsetneq T$. This creates a contradiction to maximality of Q. *(1)* $Q \subsetneq R'' \cup Q = T$ follows trivially from $R'' \nsubseteq Q$. *(2)* Since $Q \subsetneq T$ we obtain $Q'' \subseteq T''$. Thus $T'' \nsubseteq P$ follows from $Q'' \nsubseteq P$. *(3)* $R \subsetneq Q$ implies $R'' \subseteq Q''$ and thus $T = R'' \cup Q \subseteq Q''$. Hence $T'' \subseteq (Q'')'' = Q'' \subsetneq P''$. *(4)* $P \subseteq T$ would imply $P'' \subseteq T''$, a contradiction to (3). Thus $P \nsubseteq T$. From $R'' \subseteq P$ and $Q \subsetneq P$ we get $T = R'' \cup Q \subseteq P$. Therefore $T \subsetneq P$. The existence of T contradicts maximality of Q. Hence, such a set Q cannot exist and therefore P must be quasi-closed.

Minimality: Assume there is another quasi-closed set $Q \subsetneq P$, $Q'' = P''$, such that Q respects all implications from $\mathcal{S}$. Then Q cannot be an $\mathcal{S}$-pseudo-intent. Thus there must be some $\mathcal{S}$-pseudo-intent $R \subsetneq Q$ such that $R'' \nsubseteq Q$. Quasi-closedness of Q implies $R'' = Q'' = P''$ a contradiction. This proves that P is minimal with the desired properties.

We continue by proving the *if*-direction, i. e. show that every quasi-closed set P that respects all implications from $\mathcal{S}$ and is minimal among all sets with these properties must be an $\mathcal{S}$-pseudo-intent. Since we already know that P respects all implications from $\mathcal{S}$ we only need to show that for every $\mathcal{S}$-pseudo-intent $Q \subsetneq P$ it holds that $Q'' \subseteq P$. If $Q'' \neq P''$ then $Q'' \subseteq P$ since P is quasi-closed. However, if $Q'' = P''$ then minimality of P implies $P = Q$. Therefore P is an $\mathcal{S}$-pseudo-intent. □

Stumme's work provides the necessary tools to extend Algorithm 1 to handle background knowledge presented in the form of implications. Recall that Algorithm 1 works in a setting where the set of objects G is fixed, but the set of attributes is allowed to grow: $M_0 \subseteq M_1 \subseteq \ldots$. Furthermore for every natural number k the relation I_k is required to agree with I_{k+1} on all pairs from $G \times M_k$. We introduce background knowledge to this setting. We can allow the background knowledge to change during the course of the computation. We define $\mathcal{S}_0 \subseteq \mathcal{S}_1 \subseteq \ldots$ to be a sequence of implication sets. This means that whenever a new premise P_k is computed, one can add both new attributes and new background implications to obtain a new attribute set M_{k+1} and a new background implication set $\mathcal{S}_{k+1}$. Like in Algorithm 1 we write A''^{k} for A'' computed in $\mathbb{K}_k$. We also keep the requirement that attributes that have been added at a later stage have higher order than their "older" counterparts. An outline of the algorithm is presented as Algorithm 2.

Algorithm 2 Algorithm for "on the fly"-construction an implication basis with background knowledge

```
Input: 𝕂_0 = (G, M_0, I_0), 𝒮_0
Π_0 = ∅, P_0 = ∅, k = 0
while P_k ≠ null do
   Π_{k+1} = Π_k ∪ {P_k}
   k = k + 1
   Input: 𝕂_k = (G, M_k, I_k), 𝒮_k
   if M_k = M_{k−1} = P_k then
      P_k = null
   else
      P_k = lectically smallest set of attributes that is
         • closed with respect to {P_j → P_j''^k | P_j ∈ Π_k} and 𝒮_n, and
         • lectically larger than P_{k−1}.
   end if
end while
```

Again termination is guaranteed if and only if there is some $n \in \mathbb{N}$ such that $M_n = M_k$ holds for all $k \geq n$. The set of implications $\mathcal{B}_{\mathcal{S},n} = \{P_j \to P_j^{\prime\prime n} \mid P_j \in \Pi_n\}$. In perfect analogy to the proof of Lemma 5 one can prove the following lemma.

Lemma 7 *Assume that Algorithm 1 terminates after the n-th step. Let Q be a set of attributes that is quasi-closed in $\mathbb{K}_n$ and closed with respect to $\mathcal{S}$. Then there is some $P_k \in \Pi_n$ such that $P_k \subseteq Q$, $P_k^{\prime\prime n} = Q^{\prime\prime n}$.*

Because $\mathcal{S} \cup \{Q \to Q^{\prime\prime n} \mid Q$ is a quasi-closed in $\mathbb{K}_n$ and $\mathcal{S}$-closed$\}$ *is complete,* $\{P \to P^{\prime\prime n} \mid P \in \Pi_n\} \cup \mathcal{S}$ *must thus also be complete for* $\mathbb{K}_n$.

4 Computing an $\mathcal{EL}_{\mathrm{gfp}}$-GCI basis using FCA

Even though the general ideas for the framework in Section 2.2 have been taken from Formal Concept Analysis, there remain some differences between the two areas. For example in FCA is working with Formal Contexts, which do not allow for relational dependencies among objects, while we use DL-style models as the underlying data structure. When we try to translate a model into a formal context it is not immediately clear how this should be done. Simply taking all possible concept descriptions as attributes would result in an infinite context. In such a context it is not even clear whether the Duquenne-Guigues basis exists.

We will therefore have to restrict ourselves to a finite set of concept descriptions M as the set of attributes of a context. The set of objects would be the domain Δ_i of an underlying model i. We also need to define relations I_k. With the objects being elements of Δ_i and the attributes being concept descriptions there is a natural way to define I_k: For an attribute $C \in M_k$ and an object $x \in \Delta_i$ define $(x, C) \in I_k$ iff $x \in C^i$. Contexts with DL-attributes have been defined in a similar way in previous work [15, 16]. In this work we call such a context where the attribute set is a set of concept descriptions and the instance relation is obtained from a model i a *context induced by M and i*. In such a context, a subset of the attribute set can be turned into an $\mathcal{EL}_{\mathrm{gfp}}$-concept description by forming the conjunction over all elements of this set (denoted by $\bigsqcap U$). Conversely, an $\mathcal{EL}_{\mathrm{gfp}}$-concept-description C corresponds to the set of all attributes that subsume C (denoted by $\mathrm{pr}_{\mathbb{K}_n}(C)$, the projection of C to $\mathbb{K}_n$). Appendix A presents some technical results about the connection between concept descriptions and attribute sets in an induced context. It also looks at the connection between the $\cdot^i$ operators and the $\cdot'$ operators for an induced context.

It is not clear, which concept descriptions should be elements of M. But one hint how to do it comes from the structure of Λ_i, the set of premises of the GCIs in $\mathcal{B}_i$. Every concept description $C \in \Lambda_i$ can be written as the conjunction of primitive concept names and concept descriptions of the form $\exists r.X^i$, $r \in \mathcal{N}_r$, $X \in \Delta_i$. Thus every concept description $C \in \Lambda_i$ corresponds to a subset of M_i. Hence it might be worthwhile to consider a context whose set of attributes is M_i.

Now, if we look at $\mathcal{B}_i$ again, it becomes clear, that there is still some room for improvement. The left-hand sides of $\mathcal{B}_i$ simply corresponds to all subsets of M_i. We would produce less redundancy if we take only those premises that are pseudointents in the induced context.

There is, however, still the problem that, before we can build this context, we need to know all i-intents. What's worse is that most of the i-intents would be obtained twice: The first time when computing M_i and the second time as right-hand sides for the GCIs. So it would be nice to have a method where the set of attributes M is computed on the fly. As new GCIs are found, their right hand sides would be added to M.

During the rest of this section, we present an method that combines these two ideas: Using FCA methods to reduce the size of the GCI basis and computing i-intents on the fly.

For a given model i we have presented the sound and complete set of GCIs $\mathcal{B}_i$ in Section 2.3. There is, however, one thing that is problematic about this basis.

Before starting to compute GCIs, one has to find the concept intents X^i where $X \subseteq \Delta_i$. There are two reasons why this is bad. The first reason is that there may be exponentially many such intents [14]. So in the worst case, one would have to wait for an unreasonably long time before the actual computation even starts. The second reason is that computing the constituents (and thus the intents of the model i) requires full access to the model i. In the setting that we consider in this section and the previous ones full access to the model is granted. Later we would like to generalize our approach to an exploration algorithm. In such an algorithm one has only partial knowledge about the model and more information is gathered during the course of the exploration. Since the set of intents in the partial model need not be the same as the set of intents in the full model it is not possible to compute the intents beforehand.

4.1 Not using background knowledge

Let us look at the basis $\mathcal{B}_i$ in a little more detail. $\mathcal{B}_i$ is the set of all GCIs of the form $C \to C^{ii}$ where the left-hand sides C of these GCIs are contained in the set Λ_i. Λ_i is the set of all conjunctions over the elements of

$$M_i = \mathcal{N}_p \cup \{\exists r.X^i \mid r \in N_r, X \subseteq \Delta_i\}.$$

Therefore it is natural to look at contexts, that are induced by some subset of M_i.

The number of i-intents X^i can be exponential in the size of the model. Hence also M_i can become exponentially large. Therefore we want to avoid having to compute the whole set before starting the process. Instead more attributes should be computed on the fly. An idea is to instantiate Algorithm 1 in the following way. We are given a finite model i. We initiate Algorithm 1 with the context $\mathbb{K}_0 = (\Delta_i, \mathcal{N}_p, I_0)$. The context $\mathbb{K}_0$ is obtained as the induced context by i and the set of primitive concept names $\mathcal{N}_p$. In every step k of the algorithm the following things happen.

- A new premise P_k is found.
- P_k is a subset of M_k.
- Algorithm 1 asks for a new context with an extended attribute set M_{k+1}. M_{k+1} is obtained by adding to M_k all attributes of the form $\exists r.(\bigsqcap P_k)^{ii}$ where r is some role name. (This is done only, if no equivalent concept description is already present in the attribute set.)
- $\mathbb{K}_{k+1}$ is computed as the induced context by the new attribute set M_{k+1} and i.

Algorithm 3 Exploration for computing a basis for a given model i

Input: model $i = (\Delta_i, \cdot^i)$
// Initialisation
$M_0 = \mathcal{N}_p$, $\mathbb{K}_0 =$ the context induced by M_0 and i
$\Pi_0 = \emptyset$, $P_0 = \emptyset$, $k = 0$
while $P_k \neq \mathsf{null}$ **do**
  $\Pi_{k+1} = \Pi_k \cup \{P_k\}$
  $M_{k+1} = M_k \cup \{\exists r.(\bigsqcap P_k)^{ii} \mid r \in \mathcal{N}_r\}$
  $k = k + 1$
  if $M_k = M_{k-1} = P_k$ **then**
    $P_k = \mathsf{null}$
  else
    $P_k =$ lectically next set of attributes that is closed with respect to $\{P_j \to P_j^{\prime\prime_k} \mid 1 \leq j < k\}$
  end if
end while

For Algorithm 1 we know that it terminates if there is some $n \in \mathbb{N}$ such that starting from the n-th step of the algorithm no more new attributes are being added. This is guaranteed by the fact that there are only finitely many i-intents. Therefore only finitely many attributes of the form $\exists r.(\bigsqcap P_k)^{ii}$ can be added.

Assume that Algorithm 3 terminates after the n-th iteration of the while-loop. Lemma 5 proves that then the set $\{P_k \to P_k^{\prime\prime_n} \mid P_k \in \Pi_n\}$ is sound and complete for $\mathbb{K}_n$. We are, however, not interested in a basis for this context, but in a basis for the underlying model i. The rest of this section will be dealing with the proof that the corresponding set of $\mathcal{EL}_{\mathrm{gfp}}$-GCIs

$$\{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$$

is a basis for i. As a subset of $\mathcal{B}_i$ it is obviously finite and sound. The harder part shall be proving completeness.

The first thing to prove is that our algorithm is capable of finding all elements of

$$M = \mathcal{N}_p \cup \{\exists r.X^i \mid r \in N_r, X \subseteq \Delta_i\}.$$

We show that for every role name $r \in \mathcal{N}_r$ the set M_n is closed with respect to the following mapping: $U \subseteq M_n$ is mapped to the concept description $\exists r.(\bigsqcap U)^{ii}$. In a later step we prove that every concept description from M_i can be obtained from $\mathcal{N}_p$ by repeatedly applying one of these mappings. This means, that once the algorithm has terminated, every concept description from Λ_i can be expressed in terms of M_n.

Lemma 8 *Assume that Algorithm 3 terminates after the n-th iteration. Let M_n be the final set of attributes. Then for every role name $r \in \mathcal{N}_r$ and every subset $U \subseteq M_n$ there is a concept description $C \in M_n$ such that $C \equiv \exists r.(\bigsqcap U)^{ii}$.*

Proof: $V = \mathrm{pr}_{\mathbb{K}_n}((\bigsqcap U)^{ii})$ is an intent in $\mathbb{K}_n$ by Lemma 20. As an intent it is also quasi-closed. Therefore Lemma 5 shows that there is some $P_k \in \Pi_n$ such that $P_k \subseteq V$ and $P_k''^{n} = V''^{n} = V$. In the k-th iteration of the algorithm the attributes $\exists r.(\bigsqcap P_k)^{ii}$ are added to the set of attributes. $\exists r.(\bigsqcap P_k)^{ii} \equiv \exists r.(\bigsqcap U)^{ii}$ is a consequence of Lemma 21. We have thus proved that for every set $U \subseteq M_n$ and for every $r \in \mathcal{N}_r$ there is a description $C \in M_n$ such that $C \equiv \exists r.(\bigsqcap U)^{ii}$. $\square$

Lemma 9 *Let O be a set of concept descriptions with the following properties*

- *For every role name $r \in \mathcal{N}_r$ and every subset $U \subseteq O$ there is some $C \in O$ such that $C \equiv \exists r.U^{ii}$.*
- *O contains $\mathcal{N}_p$.*

Then for every role name $r \in \mathcal{N}_r$ and every set $X \subseteq \Delta_i$ there is some concept description $C \in O$ such that $C \equiv \exists r.X^i$.

Proof: Let $X \subseteq \Delta_i$ be some set of objects. We prove that there is some concept description $C \in M_n$ such that $C \equiv \exists r.X^i$ by induction over the depth of X^i. We say that an intent X^i has depth d if there is an acyclic concept description D of role depth d such that $X^i \equiv D^{ii}$. By a result from [7] such an acyclic concept description D exists for every concept description C, i. e. every i-intent has finite depth.

Base case: Let X^i be an i-intent of depth 0. This means that X^i can be expressed as

$$X^i \equiv \bigsqcap_{P \in \mathcal{P}} P$$

for some set of primitive concept names $\mathcal{P} \subseteq \mathcal{N}_p$. Since $\mathcal{N}_p \subseteq O$ it follows from the hypothesis that for every $r \in \mathcal{N}_r$ there is some concept description $C \in O$ such that

$$C \equiv \exists r.(\bigsqcap \mathcal{P})^{ii} \equiv \exists r.X^i.$$

Step case: Now assume that for every intent X^i of depth less than k and for every $r \in \mathcal{N}_r$ there is some $C_X \in O$ with $C_X \equiv \exists r.X^i$. Let Y^i, $Y \subseteq \Delta_i$ be some intent of role depth k. Then there must be some acyclic concept description D of role depth k such that $Y^i \equiv D^{ii}$. D can be written as

$$D \equiv \bigsqcap_{P \in \mathcal{P}_D} P \sqcap \bigsqcap_{1 \leq l \leq s} \exists r_l.E_l.$$

where $\mathcal{P}_D \subseteq \mathcal{N}_p$ is a set of primitive concept names, $r_l \in \mathcal{N}_r$ is a role name and E_l is an acyclic concept description of role depth less than k for all $l \in \{1, \ldots, s\}$.

Then

$$Y^i \equiv D^{ii} \equiv \left(\bigsqcap_{P \in \mathcal{P}_D} P \sqcap \bigsqcap_{l \in L} \exists r_l.E_l \right)^{ii} \equiv \left(\bigcap_{P \in \mathcal{P}_D} P^i \cap \bigcap_{l \in L} (\exists r_l.E_l)^i \right)^i$$
$$\equiv \left(\bigcap_{P \in \mathcal{P}_D} P^i \cap \bigcap_{l \in L} \{x \in \Delta_i \mid \exists y \in E_l^i : (x,y) \in r_l^i\} \right)^i$$
$$\equiv \left(\bigcap_{P \in \mathcal{P}_D} P^i \cap \bigcap_{l \in L} \{x \in \Delta_i \mid \exists y \in E_l^{iii} : (x,y) \in r_l^i\} \right)^i$$
$$\equiv \left(\bigcap_{P \in \mathcal{P}_D} P^i \cap \bigcap_{l \in L} (\exists r_l.E_l^{ii})^i \right)^i \equiv \left(\bigsqcap_{P \in \mathcal{P}_D} P \sqcap \bigsqcap_{l \in L} \exists r_l.E_l^{ii} \right)^{ii} \quad (4)$$

From the induction hypothesis we know that there are concept descriptions $C_l \in O$, for all $l \in \{1, \ldots, s\}$ such that $C_l \equiv \exists r.E_l^{ii}$ for all $l \in \{1, \ldots, s\}$. Define $U = \mathcal{P}_D \cup \{C_l \mid 1 \leq l \leq s\}$. Then $U \subseteq O$. We obtain

$$Y^i \equiv \left(\bigsqcap_{P \in \mathcal{P}_D} P \sqcap \bigsqcap_{l \in L} C_l \right)^{ii} \equiv (\bigsqcap U)^{ii}.$$

From the hypothesis we obtain that there is some $C_Y \in O$ such that $C_Y \equiv \exists r.Y^i$. $\square$

Lemma 9 and Lemma 8 prove that once Algorithm 3 terminates the final attribute set M_n contains all attributes of the form $\exists r.X^i$ for $r \in \mathcal{N}_r$, $X \subseteq \Delta_i$. This means that all concept descriptions from Λ_i can be expressed in terms of M_n (Recall that Λ_i is the set of premises of the sound an complete set of GCIs $\mathcal{B}_i$ presented in Section 2.3).

We are still trying to prove that

$$\{\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$$

with the sets P_k obtained in Algorithm 3 is complete for i. Since we have already proved completeness of $\mathcal{B}_i$ we try to deduce completeness of $\{\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$ from completeness of $\mathcal{B}_i$. The following result helps us to do this.

Lemma 10 *Let n be the number of the iteration for which Algorithm 3 terminates. Let $L \in \Lambda_i$ be a premise of the GCI set $\mathcal{B}_i$. Then either L is an i-intent or there is some $P_k \in \Pi_n$ such that $L \sqsubseteq \bigsqcap P_k$, $L \not\sqsupseteq (\bigsqcap P_k)^{ii}$.*

Proof: Assume that L is not an i-intent, i. e. $L \neq L^{ii}$. Then $\mathrm{pr}_{\mathbb{K}_n}(L) \neq \mathrm{pr}_{\mathbb{K}_n}(L^{ii})$ (This follows from Lemma 18 since both L and L^{ii} are expressible in terms of M_n). From Lemma 18 we know that $\mathrm{pr}_{\mathbb{K}_n}(L)''^n = \mathrm{pr}_{\mathbb{K}_n}(L^{ii})$. Thus $\mathrm{pr}_{\mathbb{K}_n}(L)$ is not an intent of $\mathbb{K}_n$. Since $\{P_k \to P_k''^n \mid 1 \leq k \leq n\}$ is complete for $\mathbb{K}_n$ by Lemma 5 there must be some $k \in \{1, \ldots, n\}$ such that $P_k \subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$, $P_k''^n \not\subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$. Thus $L \sqsubseteq \bigsqcap P_k$, $L \not\sqsubseteq \bigsqcap P_k''^n$ by Lemmas 16 and 18. Also from Lemma 18 we obtain $\bigsqcap P_k''^n \equiv (\bigsqcap P_k)^{ii}$ and thus $L \not\sqsubseteq \bigsqcap P_k^{ii}$. □

The next result finally ties the previous ones together. It finally enables us to prove completeness of $\{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$.

Lemma 11 *Let $\mathcal{C}$ be a set of $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions such that for every $L \in \Lambda_i$ it holds that either*

- *there is some $C \in \mathcal{C}$ such that $L \sqsubseteq C$, $L \not\sqsubseteq C^{ii}$, or*
- *$L \equiv L^{ii}$.*

Then $\{C \to C^{ii} \mid C \in \mathcal{C}\}$ is complete for i.

Proof: Assume that there is a concept descriptions $L \in \Lambda_i$ such that $L \to L^{ii}$ does not follow from $\{C \to C^{ii} \mid C \in \mathcal{C}\}$. Since Λ_i is finite we can assume without loss of generality that L is minimal with this property. L cannot be an i-intent for then $L \to L^{ii} \equiv L$ would follow trivially from any set of GCIs. Hence, the preconditions imply that there is some $C \in \mathcal{C}$ such that $L \sqsubseteq C$, $L \not\sqsubseteq C^{ii}$. Then $L \to L \sqcap C^{ii}$ follows from $C \to C^{ii}$. Furthermore $L \sqcap C^{ii} \sqsubset L$ is strictly subsumed, because of $L \not\sqsubseteq C^{ii}$. Also $L \sqcap C^{ii} \in \Lambda_i$ because of Lemma 4. We have assumed that L is minimal among all $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions for which $L \to L^{ii}$ does not follow from $\{C \to C^{ii} \mid C \in \mathcal{C}\}$. Therefore $L \sqcap C^{ii} \to (L \sqcap C^{ii})^{ii} \equiv L^{ii}$ follows from $\{C \to C^{ii} \mid C \in \mathcal{C}\}$. Thus both $L \to L \sqcap C^{ii}$ and $L \sqcap C^{ii} \to L^{ii}$ follow from $\{C \to C^{ii} \mid C \in \mathcal{C}\}$. This contradicts the assumption that $L \to L^{ii}$ does not follow from $\{C \to C^{ii} \mid C \in \mathcal{C}\}$. Therefore the assumption is false, i. e. for every $L \in \Lambda_i$ the GCI $L \to L^{ii}$ follows from $\{C \to C^{ii} \mid C \in \mathcal{C}\}$. Since $\{L \to L^{ii} \mid L \in \Lambda_i\}$ is complete, $\{C \to C^{ii} \mid C \in \mathcal{C}\}$ must also be complete. □

Now let us look at the final context $\mathbb{K}_n$ once more. From Lemma 8 we know that for every role name $r \in \mathcal{N}_r$ and every subset $U \subseteq M_n$ there is a concept description $C \in M_n$ such that $C \equiv \exists r.(\bigsqcap U)^{ii}$. Hence we can apply Lemma 9 which yields that all concept descriptions from Λ_i can be expressed in terms of M_n. This is exploited in the proof of Lemma 10 which shows that every $L \in \Lambda_i$ is either an i-intent or there is some $P_k \in \Pi_n$ such that $L \sqsubseteq \bigsqcap P_k$, $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$. Finally, Lemma 11 yields completeness of $\{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$.

Theorem 3 *Assume that Algorithm 3 terminates after the nth iteration of the while-loop. Then the set of GCIs*

$$\{\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$$

is complete for i.

4.2 Using background knowledge

The method for computing a basis used in Algorithm 3 does not take into account the subsumption hierarchy of the attributes in $\mathbb{K}_n$. Since the attributes from M_n are all $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions, it can happen that one attribute C is subsumed by another attribute D. In this case $C \rightarrow D$ holds in any model j. Thus adding $C \rightarrow D$ to the GCI basis would be redundant. In order for the FCA-based method to take into account and reduce this sort of redundancies, we can add background knowledge. This background knowledge can be represented in the form of implications $\{C\} \rightarrow \{D\}$, where $C \sqsubseteq D$. With the background knowledge in this form Stumme's approach can be applied (and its extension to a changing set of attributes, as in Algorithm 2).

Algorithm 4 Exploration for computing a basis for a given model i using background knowledge

Input: model $i = (\Delta_i, \cdot^i)$
// Initialisation
$M_0 = \mathcal{N}_p$, $\mathcal{S}_0 = \emptyset$, $\mathbb{K}_0 =$ the context induced by M_0 and i
$\Pi_0 = \emptyset$, $P_0 = \emptyset$, $k = 0$
while $P_k \neq \mathsf{null}$ **do**
  $\Pi_{k+1} = \Pi_k \cup \{P_k\}$
  $M_{k+1} = M_k \cup \{\exists r.(\bigsqcap P_k)^{ii} \mid r \in \mathcal{N}_r\}$
  $\mathcal{S}_{k+1} = \{\{C\} \rightarrow \{D\} \mid C, D \in M_{k+1}, C \sqsubseteq D\}$
  $k = k + 1$
  if $M_k = M_{k-1} = P_k$ **then**
    $P_k = \mathsf{null}$
  else
    $P_k =$ lectically next set of attributes that respects all implications from $\{P_j \rightarrow P_j''^{k} \mid 1 \leq j < k\}$ and $\mathcal{S}_k$
  end if
end while

We initiate Algorithm 2 with the context $\mathbb{K}_0 = (\Delta_i, \mathcal{N}_p, I_0)$ where $\mathbb{K}_0$ is the context induced by $\mathcal{N}_p$ and i. When Algorithm 2 finds a premise P_k it asks for a new context with an extended attribute set M_{k+1}. Just like in the version without background knowledge that has been presented in the previous Section 4.1 M_{k+1}

is obtained by adding to M_k all attributes of the form $\exists r.(\bigsqcap P_k)^{ii}$ where r is some role name. In addition to asking for a new set of attributes Algorithm 2 will also ask for a set of background implications. As background knowledge, define $\mathcal{S}_k = \{\{C\} \to \{D\} \mid C, D \in M_k, C \sqsubseteq D\}$ (in DL terms one would say that we classify M_k in each step). The resulting algorithm is presented as Algorithm 4.

Again, the algorithm will terminate, because there are only finitely many attributes that can be added. Assume that Algorithm 2 with contexts $\mathbb{K}_k$ and background knowledge $\mathcal{S}_k$ defined as above terminates after step n. We obtain a set $\Pi_n \subseteq M_n$. This set Π_n gives rise to a set of implications

$$\mathcal{B} = \{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}.$$

Since $\mathcal{B}$ is a subset of $\mathcal{B}_i$, we know that $\mathcal{B}$ is sound and finite. The proof for completeness is almost identical to the proof of completeness from Section 4.1.

Lemma 12 *Assume that Algorithm 2 with contexts $\mathbb{K}_k$ and background knowledge $\mathcal{S}_k$ defined as above terminates after the n-th iteration of the while loop. Then for every set $U \subseteq M_n$ and for every $r \in \mathcal{N}_r$ there is a description $C \in M_n$ such that $C \equiv \exists r.(\bigsqcap U)^{ii}$.*

Proof: The proof is analogous to the proof of Lemma 8. Let $U \subseteq M_n$ be some set of attributes of the final context $\mathbb{K}_n$. Then $V = \mathrm{pr}_{\mathbb{K}_n}((\bigsqcap U)^{ii})$ is an intent in $\mathbb{K}_n$ by Lemma 20. As an intent it is also quasi-closed and closed with respect to $\mathcal{S}_n$. Therefore Lemma 7 shows that there is some $P_k \in \Pi_n$ such that $P_k \subseteq V$ and $P_k^{\prime\prime_n} = V^{\prime\prime_n} = V$. In the k-th iteration of the algorithm the attributes $\exists r.(\bigsqcap P_k)^{ii}$ are added to the set of attributes. $\exists r.(\bigsqcap P_k)^{ii} \equiv \exists r.(\bigsqcap U)^{ii}$ is follows from Lemma 21. We have thus proved that for every set $U \subseteq M_n$ and for every $r \in \mathcal{N}_r$ there is a description $C \in M_n$ such that $C \equiv \exists r.(\bigsqcap U)^{ii}$. □

Since we know that $\mathcal{N}_p \in M_n$ this means that we can apply Lemma 9. Lemma 9 yields that every concept description from Λ_i can be expressed in terms of M_n.

Lemma 13 *Let n be the number of the iteration for which Algorithm 2 terminates with the contexts $\mathbb{K}_k$ and the implication sets $\mathcal{S}_k$ defined as above. Let $L \in \Lambda_i$ be a premise of the GCI set $\mathcal{B}_i$. Then either L is an i-intent or there is some $P_k \in \Pi_n$ such that $L \sqsubseteq \bigsqcap P_k$, $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$.*

Proof: Assume that L is not an i-intent, i. e. $L \not\equiv L^{ii}$. Then $\mathrm{pr}_{\mathbb{K}_n}(L) \neq \mathrm{pr}_{\mathbb{K}_n}(L^{ii})$ (This follows from Lemma 18 since both L and L^{ii} are expressible in terms of M_n). From Lemma 18 we know that $\mathrm{pr}_{\mathbb{K}_n}(L)^{\prime\prime_n} = \mathrm{pr}_{\mathbb{K}_n}(L^{ii})$. Thus $\mathrm{pr}_{\mathbb{K}_n}(L)$ is not an intent of $\mathbb{K}_n$. Since $\mathcal{S} \cup \{P_k \to P_k^{\prime\prime_n} \mid 1 \leq k \leq n\}$ is complete for $\mathbb{K}_n$ by Lemma 7 there must be some $k \in \{1, \ldots, n\}$ such that $P_k \subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$, $P_n^{\prime\prime_n} \not\subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$ or some implication $\{C\} \to \{D\} \in \mathcal{S}_k$ such that $\{C\} \subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$, $\{D\} \not\subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$. For such an implication $\{C\} \to \{D\}$ it holds that $C \sqsubseteq D$ by definition of $\mathcal{S}_k$. $C \in \mathrm{pr}_{\mathbb{K}_n}(L)$ means that $L \sqsubseteq C$. But then also $L \sqsubseteq D$ and thus $D \in \mathrm{pr}_{\mathbb{K}}(L)$. This contradiction proves that an implication $\{C\} \to \{D\} \in \mathcal{S}_k$ such that $\{C\} \subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$, $\{D\} \not\subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$ cannot exist. So there must be some $k \in \{1, \ldots, n\}$ such that $P_k \subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$, $P_n''^{n} \not\subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$. Thus $L \sqsubseteq \bigsqcap P_k$, $L \not\sqsubseteq \bigsqcap P_k''^{n}$ by Lemmas 16 and 18. Also from Lemma 18 we obtain $\bigsqcap P_k''^{n} \equiv (\bigsqcap P_k)^{ii}$ and thus $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$. $\square$

We have thus shown that every $L \in \Lambda_i$ is either an i-intent or there is some $P_k \in \Pi_n$ such that $L \sqsubseteq \bigsqcap P_k$, $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$. This allows us to apply Lemma 11 which states that $\{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$ is complete for i. Note that we do not need to add the background knowledge here, because this is already implicitly contained in the logic.

Theorem 4 *Assume that Algorithm 2 terminates after the nth iteration of the while-loop. Then the set of GCIs*

$$\{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$$

is complete for i.

5 The exploration algorithm

5.1 General Setting

In classical FCA, attribute exploration is a knowledge acquisition formalism based on expert interaction. The expert is assumed to have complete knowledge about the domain. In the standard FCA setting, one would typically start with a formal context. That formal context uses a fixed set of attributes and a set of objects that is extended during the course of the exploration. These objects serve as counterexamples for potential implications between sets of attributes. The system successively detects implications of the form $A \to B$ for which there is no counterexample. It then asks the expert whether the implication $A \to B$ holds in the domain of the exploration. If the expert accepts it the implication is added to the basis. Otherwise the expert is asked to provide a counterexample that is then added to the context. That way both the set of implications and the set of objects grow until every implication is either refuted by some counterexample or follows from the set of previously computed implications.

When transfering the idea of an exploration process to a Description Logics setting there are two choices for the underlying data structure. One can either use

ABoxes which allow for incomplete knowledge or models (or Kripke structures) which are required to contain knowledge, which is complete in a certain sense (In what sense shall become clearer soon). In our setting, we choose to represent data as a DL model and thus call the process *model exploration*. We would like to acquire knowledge in the form of GCIs in the sense of Definition 5. Just like in FCA, we assume that there is an expert who has complete knowledge about the domain of the exploration, the so-called *real world*, which in our case is a finite interpretation. The primary goal of the exploration formalism is to find a basis for the set of all GCIs that hold in this domain. The reason why we only deal with finite models is that for an infinite model there need not be a finite basis of the GCIs. In such a setting an exploration algorithm could not terminate. We call i *the background model*.

We assume, however, that at the beginning of the exploration process only some part of the model i is given to the exploration system, i. e. there is some model i_0 such that $\Delta_{i_0} \subseteq \Delta_i$. We call i_0 the working model as opposed to the background model i. At this point, we allow i_0 to be incomplete, but only in the sense that some of the potential counterexamples *are not contained* in Δ_{i_0}. We do, however, require to have complete knowledge about the objects that *are contained* in Δ_{i_0}. Now, what do we mean when we say complete knowledge? In this setting complete knowledge means that for every $x \in \Delta_{i_0}$ and for every $\mathcal{EL}_{\text{gfp}}$-concept description C it holds that $x \in C^{i_0}$ iff $x \in C^i$. Informally, this means that we know everything about x that can be expressed using $\mathcal{EL}_{\text{gfp}}$.

In order to fulfill this requirement we demand that i_0 has the following properties.

- $\Delta_{i_0} \subseteq \Delta_i$
- $P^{i_0} = P^i \cap \Delta_{i_0}$ for all $P \in \mathcal{N}_p$
- $r^{i_0} = r^i \cap (\Delta_{i_0} \times \Delta_{i_0})$ for all $r \in \mathcal{N}_r$
- $x \in \Delta_{i_0}$, $(x, y) \in r^i$ for some $r \in \mathcal{N}_r$ implies $y \in \Delta_{i_0}$

This means that i_0 is a submodel of i which is closed with respect to role successors, i. e. for every object $x \in \Delta_{i_0}$ it also contains all objects that can be reached from x by following some edge in the model $\mathcal{EL}$-description graph 3. We call such models *connected submodels of i*. Using the characterizations Proposition 1 and Theorem 1 it is straightforward to prove that for every connected submodel i_0 of i and for every $\mathcal{EL}_{\text{gfp}}$-concept description C it holds that $x \in C^{i_0}$ iff $x \in C^i$.

Our model exploration algorithm is based on Algorithm 3. Remember that, when Algorithm 3 computes a new premise, the set of attributes will change. In the model exploration algorithm (presented as Algorithm 5) the set of attributes will change, and in addition, the expert may also be asked to extend the working model. Let i_n represent the working model after the n-th step. The GCIs that the formalism finds will be of the form $C \to D$, where C and D are concept descriptions for a fixed signature $\Sigma = (\mathcal{N}_r, \mathcal{N}_p)$. Just like in standard FCA the expert can answer by either confirming or refuting the GCI. In the first case the GCI is added to the basis and the working model remains unchanged. In the latter case the expert is asked to provide a new background model i_{n+1} that contains a counterexample for the GCI. By counterexample we mean some $x \in \Delta_i$ such that $x \in C^i$ but $x \notin D^i$. In order for x to be accepted as a counterexample, the expert has to provide a connected submodel i_{n+1} of i such that $\Delta_{i_n} \subseteq \Delta_{i_{n+1}}$.

In every step the set of undecided GCIs, i. e. GCIs that neither follow from the GCI basis nor are refuted by some counterexample becomes smaller. Eventually every GCI will be decided. How and why this works in detail is explained in the following section.

5.2 General model-exploration algorithm

The algorithms presented in Section 4 have been designed to be robust with respect to changes in the underlying model. And indeed, the exploration algorithm requires only minor changes to Algorithm 3. The exploration process is initiated with the working model i_0. The algorithm starts with the context $\mathbb{K}_0$ (obtained as the context induced by i_0 and $\mathcal{N}_p$), and an empty set of premises Π_0. Like in Algorithm 3 a new context $\mathbb{K}_n$, is computed in every step of the algorithm. The following things may happen.

1. A new premise P_k is found.
2. P_k is a subset of M_k. Compute the corresponding $\mathcal{EL}_{\text{gfp}}$-concept description $\bigsqcap P_k$.
3. The expert is asked whether $\bigsqcap P_k \to (\bigsqcap P_k)^{i_j i_j}$ holds in the background model
4. If the expert refutes the GCI she is asked for a new working model i_{j+1}.
5. Repeat from Step 3 until the expert accepts $\bigsqcap P_k \to (\bigsqcap P_k)^{i_j i_j}$.
6. The algorithm asks for a new context with an extended attribute set M_{k+1}. M_{k+1} is obtained by adding to M_k all attributes of the form $\exists r.(\bigsqcap P_k)^{ii}$ where r is some role name. (This is done only, if no equivalent concept description is already present in the attribute set.)
7. $\mathbb{K}_{k+1}$ is computed as the induced context by the new attribute set M_{k+1} and i.

The modification with respect to Algorithm 3 merely consists of adding a second while-loop to the algorithm. This inner loop is used to determine the proper conclusion $(\bigsqcap P_k)^{ii}$ for a given premise $\bigsqcap P_k$. Since i is not known $(\bigsqcap P_k)^{ii}$ cannot be computed directly, but only by interacting with the expert. This is done in the following way. The GCI $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ is presented to the expert. If the expert refutes the GCI then she is required to provide a counter-example, i. e. to provide a connected submodel i_{j+1} of i that extends i_j, and in which $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ does not hold. This is repeated until the expert states that $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ holds in i. One can proof that if $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ holds in i then $(\bigsqcap P_k)^{i_j i_j} \equiv (\bigsqcap P_k)^{ii}$. This way we can obtain $(\bigsqcap P_k)^{ii}$ by repeatedly questioning the expert.

Lemma 14 (Termination) *Algorithm 5 terminates after a finite number of steps.*

Proof: There are only finitely many attributes that can be added. Hence the outer while loop can be entered only finitely often. The inner while-loop can only be passed a finite number of times, since with each pass the model i_j is extended. Since i_j is a submodel of i and i is finite this can only happen finitely often. □

Just like for Algorithm 1 the set of GCIs

$$\{\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j} \mid P_k \in \Pi_n\}$$

must be finite and sound, as it is a subset of $\mathcal{B}_i$, which is a finite basis for i.

Theorem 5 (Completeness:) *Assume that Algorithm 5 terminates after the n-th iteration of the outer while loop. Then the set of GCIs of the form $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{ii}$, $0 \leq k \leq n$, is complete for i.*

Proof: We prove completeness by proving that Algorithm 5 with the working model i_0 as input finds exactly the same GCI set as Algorithm 3 with the complete background model i as input. This is done by induction over n.

Denote by $\bar{P}_k$, $\bar{M}_k$, the respective premises, and attributes found by Algorithm 5 in the k-th step.

Obviously $P_0 = \bar{P}_0 = \emptyset$, and $M_0 = \bar{M}_0 = \mathcal{N}_p$.

Assume that for all $k \leq k_0$ it holds that $P_k \equiv \bar{P}_k$, and $M_k \equiv \bar{M}_k$. Algorithm 5 does not return from the inner while-loop until the working model i_j is such that $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ holds in i.

As submodels i_j we only allow models for which $x \in \Delta_{i_j}$ and $x \in (\bigsqcap P_k)^{i_j}$ implies $x \in (\bigsqcap P_k)^{i}$. Therefore $(\bigsqcap P_k)^{i_j} \subseteq (\bigsqcap P_k)^{i}$ and thus $((\bigsqcap P_k)^{i_j})^{i} \sqsubseteq (\bigsqcap P_k)^{ii}$. One

Algorithm 5 Model-Exploration Algorithm without background knowledge

Input: working model i_0 (connected submodel of background model i)
$M_0 := \mathcal{N}_{\text{prim}}$, $\mathbb{K}_0 :=$ the context induced by M_0 and i_0,
$\Pi_0 := \emptyset$, $P_0 := \emptyset$, $k := 0$, $j := 0$
while $P_k \neq \mathsf{null}$ **do**
  while expert refutes $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ **do**
    $j := j + 1$
    Ask the expert for a new working model i_j that extends i_{j-1}, is a connected submodel of i, and contains a counterexample for $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$
  end while
  $\Pi_{k+1} := \Pi_k \cup \{P_k\}$
  $M_{k+1} := M_k \cup \{\exists r.P_k^{i_j i_j} \mid r \in \mathcal{N}_r\}$
  $k := k + 1$
  if $M_k = M_{k-1} = P_k$ **then**
    $P_k := \mathsf{null}$
  else
    $P_k :=$ lectically next set of attributes that respects all implications in $\{P_l \rightarrow P_l^{\prime\prime_k} \mid 1 \leq l < k\}$
  end if
end while

can show [1] that $((\bigsqcap P_k)^{i_j})^i \equiv (\bigsqcap P_k)^{i_j i_j}$ and therefore $(\bigsqcap P_k)^{i_j i_j} \sqsubseteq (\bigsqcap P_k)^{ii}$. The expert has stated that $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ holds in i. This means that $(\bigsqcap P_k)^i \subseteq ((\bigsqcap P_k)^{i_j i_j})^i$. But $(\bigsqcap P_k)^{ii}$ is defined to be the most specific concept for $(\bigsqcap P_k)^i$ in i. The definition of most specific concept implies $(\bigsqcap P_k)^{ii} \sqsubseteq (\bigsqcap P_k)^{i_j i_j}$. Thus $(\bigsqcap P_k)^{i_j i_j} \equiv (\bigsqcap P_k)^{ii} \equiv \bigsqcap \bar{P}_k^{ii}$.

(1) We prove $M_{k+1} = \bar{M}_{k+1}$. M_{k+1} is obtained from M_k by adding all $\mathcal{EL}_{\text{gfp}}$-concept descriptions of the form $\exists r.(\bigsqcap P_k)^{i_j i_j}$. Likewise $\bar{M}_{k+1}$ is obtained by adding all $\mathcal{EL}_{\text{gfp}}$-concept descriptions of the form $\exists r.(\bigsqcap \bar{P}_k)^{ii}$ to $\bar{M}_k$. Since $P_k = \bar{P}_k$ by the induction hypothesis and $P_k^{ii} = P_k^{i_j i_j}$ it follows that $M_{k+1} \equiv M_{k+1}$.

(2) We show $P_{k+1} = \bar{P}_{k+1}$. P_{k+1} is the lectically smallest subset of M_{k+1} that

- closed with respect to $\{P_l \rightarrow P_l^{\prime\prime_{k+1}} \mid l \leq k\}$, and
- lectically greater than P_k.

An analogous definition holds for $\bar{P}_{k+1}$. We already know that $P_l = \bar{P}_l$ for all $l \leq k$ and $M_{k+1} = \bar{M}_{k+1}$. So what remains to be shown is $P_l^{\prime\prime_{k+1}} = \bar{P}_l^{\prime\prime_{k+1}}$ for all $l \leq k$.

[1] By definition for every $x \in \Delta_{i_j}$ and every concept description C it holds that $x \in C^{i_k}$ if and only if $x \in C^i$. In other words, for every $U \subseteq \Delta_{i_k}$ it holds that $U \subseteq C^{i_k}$ if and only if $U \subseteq C^i$. Thus $U^{i_k} \equiv U^i$ for all $U \subseteq \Delta_{i_k}$ by definition of most specific concepts.

Algorithm 6 The Model-Exploration Algorithm using background knowledge

Input: working model i_0 (connected submodel of background model i)
$M_0 := \mathcal{N}_{\text{prim}}$, $\mathbb{K}_0 :=$ the context induced by M_0 and i_0, $\mathcal{S}_0 := \emptyset$
$\Pi_0 := \emptyset$, $P_0 := \emptyset$, $k := 0$, $j := 0$
while $P_k \neq \mathsf{null}$ **do**
  while expert refutes $\bigsqcap P_k \to (\bigsqcap P_k)^{i_j i_j}$ **do**
    $j := j + 1$
    Ask the expert for a new working model i_j that extends i_{j-1}, is a connected submodel of i, and contains a counterexample for $\bigsqcap P_k \to (\bigsqcap P_k)^{i_j i_j}$
  end while
  $\Pi_{k+1} := \Pi_k \cup \{P_k\}$
  $M_{k+1} := M_k \cup \{\exists r.P_k^{i_j i_j} \mid r \in \mathcal{N}_r\}$
  $\mathcal{S}_{k+1} := \{C \to D \mid C, D \in M_k, C \sqsubseteq D\}$
  $k := k + 1$
  if $M_k = M_{k-1} = P_k$ **then**
    $P_k := \mathsf{null}$
  else
    $P_k :=$ lectically next set of attributes that respects all implications in $\{P_l \to P_l^{\prime\prime_k} \mid 1 \leq l < k\}$ and $\mathcal{S}_k$
  end if
end while

From Lemma 20 we obtain $P_l^{\prime\prime_{k+1}} = \text{pr}_{\mathbb{K}_{k+1}}((\bigsqcap P_l)^{i_j i_j})$. Since we have shown that $(\bigsqcap P_l)^{i_j i_j} = (\bigsqcap P_l)^{ii}$ it follows that $P_l^{\prime\prime_{k+1}} = \text{pr}_{\mathbb{K}_{k+1}}((\bigsqcap P_l)^{ii}) = \text{pr}_{\bar{\mathbb{K}}_{k+1}}\big((\bigsqcap \bar{P}_l)^{ii}\big) = \bar{P}_l^{\prime\prime_{k+1}}$.

Now we have shown that the model-exploration algorithm 5 finds exactly the same premises as Algorithm 3. Therefore the set $\{P_k \to P_k^{i_j i_j} \mid P_k \in \Pi_n\}$ is complete. $\square$

Algorithm 5 can be extended to allow for background knowledge to be included. This yields algorithm 6. Correctness and completeness can be proved in analogy to Algorithm 5.

We have developed an expert based method that will find a finite basis for $\mathcal{EL}_{\text{gfp}}$-GCIs that hold in the background model. Since the use of standard $\mathcal{EL}$ is far more common than the use of $\mathcal{EL}_{\text{gfp}}$, it would be desirable to have a method that works with standard $\mathcal{EL}$ only. And indeed it can be shown that one can always obtain a finite basis of $\mathcal{EL}$-GCIs from a finite basis of $\mathcal{EL}_{\text{gfp}}$-GCIs [7]. Unfortunately, one has to know the size of the underlying model for this construction. But this means that we cannot use standard $\mathcal{EL}$-GCIs *during* the exploration process (at least not by applying this construction to the GCIs found by our algorithm). This is because the size of the working model can grow during the process. After termination the GCIs holding in the final working model i_n are just the GCIs holding in the background model i So the basis $\{\bigsqcap P_k \to (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}$ can be transformed into a basis of $\mathcal{EL}$-GCIs (*after* the exploration process has terminated). This is important, because this allows us to encode the GCIs as GCIs in the popular DL-language $\mathcal{EL}$++, which provides for GCIs but not greatest fixpoint semantics.

An example

We illustrate Algorithm 3 using the example from the introduction. The domain of the background model thus consists of six persons: John, Michelle and their daughter Mackenzie, as well as Paul, Linda and their son James.[2] As primitive concepts we use Male, Female, Father and Mother, and as role child. Let us assume that the initial working model i_0 contains only the first family, i.e., Δ_{i_0} consists of John, Michelle, and Mackenzie, and we have

$$\mathsf{Male}^{i_0} = \mathsf{Father}^{i_0} = \{\text{John}\}, \qquad \mathsf{Mother}^{i_0} = \{\text{Michelle}\},$$
$$\mathsf{Female}^{i_0} = \{\text{Michelle}, \text{Mackenzie}\}, \quad \mathsf{child}^{i_0} = \{(\text{Michelle}, \text{Mackenzie}), (\text{John}, \text{Mackenzie})\}.$$

1st Iteration: The algorithm starts with $P_0 = \emptyset$. We have $\bigsqcap P_0 = \top$. The first step of the algorithm would be to compute $\top^{i_0 i_0}$. We do this in a little more detail. Obviously all objects are in the extension of $\top$ and thus

$$\top^{i_0} = \{\text{John}, \text{Michelle}, \text{Mackenzie}\}.$$

A next step is to compute the model based most specific concept for each of the three sets {John}, {Michelle} and {Mackenzie}. This is a very simple procedure which is explained in [1]. The results are $\{\text{John}\}^{i_0} = \mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists \mathsf{child}.\mathsf{Female}$, $\{\text{Michelle}\}^{i_0} = \mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists \mathsf{child}.\mathsf{Female}$ and $\{\text{Mackenzie}\}^{i_0} = \mathsf{Female}$. $\top^{i_0 i_0}$ is the least common subsumer of $\{\text{John}\}^{i_0}$, $\{\text{Michelle}\}^{i_0}$ and $\{\text{Mackenzie}\}^{i_0}$. Since the only common subsumer of these three descriptions is $\top$ we obtain $\top^{i_0 i_0} = \top$. Thus the expert is asked whether the GCI $\top \to \top$ holds in i. Obviously, the answer must be "yes," and we continue by computing the new set of attributes Male_1 by adding $\exists r.\top$ to $\mathsf{Male}_0 = \mathcal{N}_{\text{prim}}$. The induced context $\mathbb{K}_1$ obtained this way is

	Father	Male	Mother	Female	$\exists$child.$\top$
John	X	X			X
Michelle			X	X	X
Mackenzie			X		

where we assume that the elements of Male_1 are ordered as listed in the table.

[2]Since this is a very simple model, it satisfies GCIs not holding in the "real world."

2nd Iteration: The lectically next set that is closed with respect to $\{\emptyset \rightarrow \emptyset''^{1}\} = \{\emptyset \rightarrow \emptyset\}$ is $\{\mathsf{Father}\}$. We have $\mathsf{Father}^{i_0 i_0} = \{\text{John}\}^{i_0} = \mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists\mathsf{child}.\mathsf{Female}$, which gives rise to the GCI $\mathsf{Father} \rightarrow \mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists\mathsf{child}.\mathsf{Female}$. Thus, the expert is presented with the question: "Is it true that every father is male and has a child that is female?". This is not true in the background model i since Paul is a father without daughter. The expert refutes the GCI by adding Paul as a counterexample. Note that she must also add James, because the new working model i_1 must be a connected submodel of i. Based on this model, the algorithm computes a new right-hand-side for the GCI: $\mathsf{Father}^{i_1 i_1} = \mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists\mathsf{child}.\top$. The new GCI $\mathsf{Father} \rightarrow \mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists\mathsf{child}.\top$ is presented to the expert, who accepts it. Consequently, the new attribute $\exists\mathsf{child}.(\mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists\mathsf{child}.\top)$ is added.

We do not look at the next iterations in as much detail as for the first two. The following GCIs are found:

1. $\mathsf{Mother} \rightarrow \mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists\mathsf{child}.\mathsf{Female}$ (Refuted, Linda added as counterexample)
2. $\mathsf{Mother} \rightarrow \mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists\mathsf{child}.\top$ (Accepted)
3. $\mathsf{Female} \sqcap \mathsf{Male} \rightarrow \mathsf{AllAttributes}$ (Accepted)
4. $\exists\mathsf{child}.\top \sqcap \mathsf{Male} \rightarrow \mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists\mathsf{child}.\top$ (Accepted)
5. $\exists\mathsf{child}.\top \sqcap \mathsf{Female} \rightarrow \mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists\mathsf{child}.\top$ (Accepted)
6. $\exists\mathsf{child}.\mathsf{Male} \sqcap \exists\mathsf{child}.\mathsf{Female} \rightarrow \mathsf{AllAttributes}$ (Accepted)
7. $\exists\mathsf{child}.\exists\mathsf{child}.\top \rightarrow \mathsf{AllAttributes}$ (Accepted)

Here $\mathsf{AllAttributes}$ ("all attributes") stands for the cyclic $\mathcal{EL}_{\text{gfp}}$-concept description $(\mathcal{T}, A)$ where $\mathcal{T} = \{A \equiv \mathsf{Male} \sqcap \mathsf{Female} \sqcap \mathsf{Mother} \sqcap \mathsf{Father} \sqcap \exists\mathsf{child}.A\}$. Note that $\mathsf{AllAttributes}$ is subsumed by any $\mathcal{EL}_{\text{gfp}}$-concept description that can be formulated using the primitive concepts Male, Female, Father, Mother and the role child. As such, it is the best approximation of the bottom concept that $\mathcal{EL}_{\text{gfp}}$ can come up with.

Interestingly, all the GCIs accepted during the exploration process, except for the last two (6. and 7.), hold in the "real world." The GCIs 6. and 7. are artefacts of the simple model i used for the exploration. They are due to the fact that, in i, there are no grandparents, and no one has both a son and a daughter.

Upon termination, the exploration algorithm will have added a total of 13 attributes, which are:

- $\exists \mathsf{child}.\top$
- $\exists \mathsf{child}.(\mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists \mathsf{child}.\top)$
- $\exists \mathsf{child}.\mathsf{Male}$
- $\exists \mathsf{child}.(\mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists \mathsf{child}.\top)$
- $\exists \mathsf{child}.\mathsf{Female}$
- $\exists \mathsf{child}.\mathsf{AllAttributes}$
- $\exists \mathsf{child}.\exists \mathsf{child}.\top$
- $\exists \mathsf{child}.\exists \mathsf{child}.\mathsf{Male}$
- $\exists \mathsf{child}.(\mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists \mathsf{child}.\mathsf{Male})$
- $\exists \mathsf{child}.(\mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists \mathsf{child}.\mathsf{Male})$
- $\exists \mathsf{child}.\exists \mathsf{child}.\mathsf{Female}$
- $\exists \mathsf{child}.(\mathsf{Father} \sqcap \mathsf{Male} \sqcap \exists \mathsf{child}.\mathsf{Female})$
- $\exists \mathsf{child}.(\mathsf{Mother} \sqcap \mathsf{Female} \sqcap \exists \mathsf{child}.\mathsf{Female})$

13 attributes may look like a lot for such a small example, but it is not. Just take into account that with 4 primitive concept names and 1 role name one can form up to 256 $\mathcal{EL}$-concept descriptions of at role depth less than or equal to 2 (and that is counting only those that cannot be written as a conjunction).

6 Related and future work

As mentioned before, the context induced by a finite model and a finite set of concept descriptions as attributes has been considered before (e.g., in [15, 16]). However, since this previous work did not make use of the most specific concept, the authors could not show and utilize the connections between the $\cdot^i$ operators in the model and the $\cdot'$ operators in the induced context. The work whose objectives is closest to ours is [16], where Rudolph considers attributes defined in the DL $\mathcal{FLE}$, which is more expressive than $\mathcal{EL}$. Given a finite $\mathcal{FLE}$-model, he considers an infinite family of induced contexts $\mathbb{K}_n$, where the finite attribute sets are obtained by considering all $\mathcal{FLE}$-concept descriptions (modulo equivalence) up to role depth n. He then applies classical attribute exploration to these induced contexts, in each step increasing the role depths until a certain termination condition applies. Rudolph shows that the implication bases of the contexts considered up to the last step contain enough information to decide, for any GCI between $\mathcal{FLE}$-concept descriptions, whether this GCI holds in the given model or not. However, these implication bases do not appear to yield a basis for all the GCIs holding in the given finite model, though it might be possible to modify Rudolph's approach such that it produces a basis in our sense. The main problem with this approach is, however, that the number of attributes grows very fast when the role depth grows (this number increases at least by one exponential in each step). In contrast to considering all concept descriptions up to a certain role depth, our approach only adds an attribute of the form $\exists r.(\bigsqcap P)^{ii}$ if P has been generated as the left-hand side of a GCI in our basis.

The main topic for future research is to show that the approach for using attribute exploration to complete DL knowledge bases introduced in [8] can be extended to the model exploration algorithm introduced in this paper.

A Induced Contexts

What we call induced contexts in this work are basically Formal Contexts whose attributes are DL-concept descriptions and whose set of objects is the domain of a DL-model. We say that an object x *has* an attribute C if x is in the extension of C for a given model i.

Similar contexts have been introduced in [16, 15]. This section serves to examine the connection between the $\cdot'$-Operators in the induced context and the $\cdot^i$-functions in the corresponding DL-model. Induced contexts will be useful since they establish the connection between the DL world and the FCA world which we need for our algorithms. But let us first give a formal definition.

Definition 16 (induced context) *Let i be a model for some DL-language $\mathcal{L}$. Let $\mathbb{K} = (G, M, I)$ be a formal context such that $G = \Delta_i$, M is a finite set of $\mathcal{L}$-concept descriptions and*

$$I = \{(x, C) \mid x \in C^i\}.$$

Then $\mathbb{K}$ is called the context induced by M and i*.*

In FCA an object is in the extension of a set of attributes U iff it has *all* the attributes from $\mathcal{C}$. In DL terms this means that x is in the extension of the conjunction over all elements of U. Therefore we can say that U corresponds to $\bigsqcap_{C \in U} C$.

On the other hand, we can approximate a concept description C by taking the set of all attributes $D \in M$ that subsume C. Since M in general contains only a small number of attributes this is really just an approximation.

Definition 17 *Let $\mathbb{K}$ be the* induced context *by M and i. Let C be some $\mathcal{L}$ concept description and $U \subseteq M$ a subset of M. Define*

$$\mathrm{pr}_{\mathbb{K}}(C) = \{D \in M \mid C \sqsubseteq D\},$$

the projection of C to $\mathbb{K}$*. Define*

$$\bigsqcap U = \bigsqcap_{D \in U} D,$$

the concept defined by U*.*

We have already mentioned that for a given set of attributes $U \subseteq M$ the extent U' corresponds to the extension of $\bigsqcap U$. More formally this is captured in the following Lemma.

Lemma 15 *Let $\mathbb{K}$ be the context induced by some set M and some model i. Let U some subset of M, and O some subset of Δ_i. Then*

1. $U' = (\bigsqcap U)^i$, *and*
2. $O' = \mathrm{pr}_{\mathbb{K}}(O^i)$.

Proof: (1) Consider $U \subseteq M$. Then

$$\begin{aligned}(\bigsqcap U)^i =& \left(\bigsqcap_{D \in U} D\right)^i \\ =& \left\{x \in \Delta_i \mid \forall D \in U : x \in D^i\right\} \\ =& U'\end{aligned}$$

(2) Let $O \subseteq \Delta_i$ be some set of objects in $\mathbb{K}$. Then it follows from the definition of O^i that

$$O^i \sqsubseteq D \Leftrightarrow O \subseteq D^i.$$

Thus

$$\begin{aligned}\mathrm{pr}_{\mathbb{K}}(O^i) =& \{D \in M \mid O^i \sqsubseteq D\} \\ =& \{D \in M \mid O \subseteq D^i\} \\ =& O'.\end{aligned}$$

□

We continue with some calculation rules for induced contexts, projections and concepts defined by attribute sets.

Lemma 16 *Let C and D be some concept descriptions such that $C \sqsubseteq D$. Then $\mathrm{pr}_{\mathbb{K}}(D) \subseteq \mathrm{pr}_{\mathbb{K}}(C)$. Let $U, V \subseteq M$ be such that $U \subseteq V$. Then $\bigsqcap V \sqsubseteq \bigsqcap U$.*

Proof: The first part of the lemma is trivial: Let $E \in \mathrm{pr}_{\mathbb{K}}(D)$ be some concept description. By definition $D \sqsubseteq E$ and thus also $C \sqsubseteq E$ which implies $E \in \mathrm{pr}_{\mathbb{K}}(C)$. The second part is not much harder than the first part: By definition

$$\bigsqcap V = \bigsqcap_{D \in V} D = \bigsqcap U \sqcap \bigsqcap_{D \in V \setminus U} D.$$

Therefore obviously $\bigsqcap V \sqsubseteq \bigsqcap U$.

Lemma 17 *Let $\mathbb{K}$ be the context induced by M and i. Let C be an $\mathcal{L}$-concept description, $U \subseteq M$ a set of attributes. Then the following statements hold:*

1. $C \sqsubseteq \bigsqcap \mathrm{pr}_{\mathbb{K}}(C)$
2. $\mathrm{pr}_{\mathbb{K}}(C)'' \subseteq \mathrm{pr}_{\mathbb{K}}(C^{ii})$
3. $U \subseteq \mathrm{pr}_{\mathbb{K}}(\bigsqcap U)$
4. $(\bigsqcap U)^{ii} \sqsubseteq \bigsqcap U''$

Proof: (1): It holds that $C \sqsubseteq D$ for all $D \in \mathrm{pr}_{\mathbb{K}}(C)$ and thus

$$C \sqsubseteq \bigsqcap_{D \in \mathrm{pr}_{\mathbb{K}}(C)} D \equiv \bigsqcap \mathrm{pr}_{\mathbb{K}}(C)\,.$$

(2): From Lemma 15 it follows that $\mathrm{pr}_{\mathbb{K}}(C^{ii}) = (C^i)'$. $C \sqsubseteq D$ for all $D \in \mathrm{pr}_{\mathbb{K}}(C)$ hold by definition. Thus

$$C^i \subseteq \bigcap_{D \in \mathrm{pr}_{\mathbb{K}}(C)} D^i = \{x \in \Delta_i \mid \forall D \in \mathrm{pr}_{\mathbb{K}}(C) : x \in D^i\} = \mathrm{pr}_{\mathbb{K}}(C)'\,.$$

Therefore $\mathrm{pr}_{\mathbb{K}}(C)'' \subseteq (C^i)' = \mathrm{pr}_{\mathbb{K}}(C^{ii})$.

(3): It holds that $\bigsqcap U \sqsubseteq F$ for all $F \in U$. Thus $F \in \mathrm{pr}_{\mathbb{K}}(\bigsqcap U)$ for all $F \in U$. Hence $\mathrm{pr}_{\mathbb{K}}(\bigsqcap U) \supseteq U$.

(4): From Lemma 15 obtain

$$\bigsqcap U'' \equiv \bigsqcap \Big((\bigsqcap U)^i \Big)' \equiv \bigsqcap \mathrm{pr}_{\mathbb{K}}\Big((\bigsqcap U)^{ii} \Big)\,.$$

Then

$$(\bigsqcap U)^{ii} \sqsubseteq \bigsqcap \mathrm{pr}_{\mathbb{K}}\Big((\bigsqcap U)^{ii} \Big) \equiv \bigsqcap U''$$

follows from (1). □

The reason why we have only subsumption or subset relations in Lemma 17 is that not everything that can be expressed in the form of an $\mathcal{L}$-concept description can also be expressed as a subset of M. Since there are usually infinitely many $\mathcal{L}$-concept descriptions, but only finitely many subsets of M something is lost in the conversion. We now consider the case where M is large enough to express at least some of the relevant concepts.

Definition 18 *Let M be a set of $\mathcal{L}$ concept descriptions and C an $\mathcal{L}$ concept description. We say that C can be expressed in terms of M iff there is some subset $N \subseteq M$ such that*

$$C \equiv \bigsqcap_{D \in N} D \equiv \bigsqcap N.$$

Lemma 18 *Let M be a set of $\mathcal{L}$ concept descriptions. Let C be an $\mathcal{L}$-concept description that can be expressed in terms of M. Let $U \subseteq M$ be a set of attributes such that $(\bigsqcap U)^{ii}$ can be expressed in terms of M. Then the following statements hold*

1. $C \equiv \bigsqcap \mathrm{pr}_{\mathbb{K}}(C)$
2. $\mathrm{pr}_{\mathbb{K}}(C^{ii}) = \mathrm{pr}_{\mathbb{K}}(C)''$
3. $\bigsqcap U'' \equiv (\bigsqcap U)^{ii}$

Proof: (1) C can be written as the conjunction of elements of M, say

$$C \equiv \bigsqcap_{D \in N} D,$$

for some set $N \subseteq M$. Obviously for every $D \in N$ it holds that $C \sqsubseteq D$ and thus $N \subseteq \mathrm{pr}_{\mathbb{K}}(C)$. Hence

$$\begin{aligned}\bigsqcap \mathrm{pr}_{\mathbb{K}}(C) &\equiv \bigsqcap_{D \in \mathrm{pr}_{\mathbb{K}}(C)} D \\ &\sqsubseteq \bigsqcap_{D \in N} D \equiv C\end{aligned}$$

With Lemma 17 it follows that $C \equiv \bigsqcap \mathrm{pr}_{\mathbb{K}}(C)$.

(2) From Lemma 15 it follows that $\mathrm{pr}_{\mathbb{K}}(C^{ii}) = (C^i)'$. Since C can be expressed in terms of M there must be some set $N \subseteq M$ such that

$$C \equiv \bigsqcap_{D \in N} D.$$

Then $N \subseteq \mathrm{pr}_{\mathbb{K}}(C)$ and thus

$$C^i = \{x \in \Delta_i \mid \forall D \in N : x \in D^i\} \supseteq \{x \in \Delta_i \mid \forall D \in \mathrm{pr}_{\mathbb{K}}(C) : x \in D^i\} = \mathrm{pr}_{\mathbb{K}}(C)'.$$

Thus $\mathrm{pr}_{\mathbb{K}}(C^{ii}) = (C^i)' \subseteq \mathrm{pr}_{\mathbb{K}}(C)''$.

(3) From Lemma 15 obtain

$$\bigsqcap U'' \equiv \bigsqcap \left((\bigsqcap U)^i\right)' \equiv \bigsqcap \mathrm{pr}_{\mathbb{K}}\left((\bigsqcap U)^{ii}\right).$$

Since $(\bigsqcap U)^{ii}$ can be expressed in terms of M $\bigsqcap U'' \equiv (\bigsqcap U)^{ii}$ follows from (1). $\square$

So we have now found a criterion for which the subset and subsumption relations of Lemma 17 (1), (2) and (4) became equality or equivalence. The only one that is missing is Lemma 17 (3). We prove that equivalence holds for concept intents.

Lemma 19 *Let U'' be a concept intent in the context $\mathbb{K}$ induced by M and i. Then*

$$U'' = \mathrm{pr}_{\mathbb{K}}\Big(\bigsqcap U''\Big)$$

Proof: By definition of the $\cdot'$ operator and the relation I we obtain

$$\begin{aligned} U'' &= \{D \in M \mid \forall x \in U' : x \in D^i\} \\ &= \{D \in M \mid U' \subseteq D^i\} \end{aligned}$$

Let $E \in M$ be a concept description such that $\bigsqcap U'' \sqsubseteq E$. This means that $U' \subseteq \bigcap_{U' \subseteq D^i} D^i = (\bigsqcap_{U' \subseteq D^i} D)^i = \bigsqcap U''^i \subseteq E^i$. Therefore $E \in U''$. On the other hand $\bigsqcap U'' \sqsubseteq D$ holds for all $D \in U''$ by definition of $\bigsqcap U''$. Thus

$$\begin{aligned} \mathrm{pr}_{\mathbb{K}}\Big(\bigsqcap U''\Big) &= \{E \in M \mid \bigsqcap U'' \sqsubseteq E\} \\ &= \{E \in M \mid E \in U''\} \\ &= U''. \end{aligned}$$

□

Lemma 20 *Let $U \subseteq M$ be any set of attributes in a context $\mathbb{K}$ induced by i. Then*

$$U'' = \mathrm{pr}_{\mathbb{K}}\Big((\bigsqcap U)^{ii}\Big).$$

Proof: Let $C \in M$ be a concept description. Then it holds that

$$\begin{aligned} C \in \mathrm{pr}_{\mathbb{K}}\Big((\bigsqcap U)^{ii}\Big) &\Leftrightarrow (\bigsqcap U)^{ii} \sqsubseteq C \\ &\Leftrightarrow (\bigsqcap U)^{i} \subseteq C^i \end{aligned}$$

From Lemma 15:

$$\begin{aligned} C \in \mathrm{pr}_{\mathbb{K}}\Big((\bigsqcap U)^{ii}\Big) &\Leftrightarrow U' \subseteq C^i \\ &\Leftrightarrow C \in U''. \end{aligned}$$

Therefore $\mathrm{pr}_{\mathbb{K}}((\bigsqcap U)^{ii}) = U''$.

Lemma 21 *Let $\mathbb{K}$ be a context induced by M and i. Let $U \subseteq M$ and $P \subseteq M$ be sets of attributes such that $P \subseteq U''$, $P'' = U''$. Then $(\bigsqcap U)^{ii} \equiv (\bigsqcap P)^{ii}$.*

Proof: (1) It holds that $(\bigsqcap U)^{ii} \sqsubseteq \bigsqcap \mathrm{pr}_{\mathbb{K}}((\bigsqcap U)^{ii}) \equiv \bigsqcap U''$ by Lemma 17 and Lemma 20. And thus $(\bigsqcap U)^{ii} \sqsubseteq (\bigsqcap U'')^{ii}$ follows from Lemma 1. On the other hand $U \subseteq U''$ and therefore $\bigsqcap U'' \sqsubseteq \bigsqcap U$ by Lemma 17. But then $(\bigsqcap U'')^{ii} \sqsubseteq (\bigsqcap U)^{ii}$ by Lemma 1. Thus $(\bigsqcap U'')^{ii} = (\bigsqcap U)^{ii}$. (2) Since $P \subseteq U''$ we obtain $(\bigsqcap U'')^{ii} \sqsubseteq (\bigsqcap P)^{ii}$ from Lemma 16 and Lemma 1. On the other hand $(\bigsqcap P)^{ii} \sqsubseteq \bigsqcap P''$ follows from Lemma 17 and thus $(\bigsqcap P)^{ii} \sqsubseteq \bigsqcap U''$ because $U'' = P''$. Lemma 1 shows that $(\bigsqcap P)^{ii} \sqsubseteq (\bigsqcap U'')^{ii}$. Hence $(\bigsqcap P)^{ii} \equiv (\bigsqcap U'')^{ii}$. (3) Results (1) and (2) together imply that $(\bigsqcap P)^{ii} \equiv (\bigsqcap U)^{ii}$.

References

[1] Franz Baader. Least common subsumers and most specific concepts in a description logic with existential restrictions and terminological cycles. In Georg Gottlob and Toby Walsh, editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 319–324. Morgan Kaufmann, 2003.

[2] Franz Baader. Terminological cycles in a description logic with existential restrictions. In Georg Gottlob and Toby Walsh, editors, *Proc. of the 18th Int. Joint Conf. on Artificial Intelligence (IJCAI 2003)*, pages 325–330, Acapulco, Mexico, 2003. Morgan Kaufmann, Los Altos.

[3] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, pages 364–369, Edinburgh (UK), 2005. Morgan Kaufmann, Los Altos.

[4] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[5] Franz Baader and Felix Distel. A finite basis for the set of $\mathcal{EL}$-implications holding in a finite model. Technical Report 07-02, Inst. für Theoretische Informatik, TU Dresden, Dresden, Germany, 2007.

[6] Franz Baader and Felix Distel. Exploring finite models in the description logic $\mathcal{EL}_{\mathrm{gfp}}$. Technical Report 08-05, Institute for theoretical computer science, TU Dresden, Dresden, Germany, 2008.

[7] Franz Baader and Felix Distel. A finite basis for the set of $\mathcal{EL}$-implications holding in a finite model. In Raoul Medina and Sergei Obiedkov, editors, *Proceedings of the 6th International Conference on Formal Concept Analysis (ICFCA '08)*, volume 4933 of *LNAI*, pages 46–61. Springer, 2008.

[8] Franz Baader, Bernhard Ganter, Ulrike Sattler, and Barış Sertkaya. Completing description logic knowledge bases using formal concept analysis. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007)*. AAAI Press/The MIT Press, 2007.

[9] Franz Baader, Carsten Lutz, and Bontawee Suntisrivaraporn. CEL—a polynomial-time reasoner for life science ontologies. In Ulrich Furbach and Natarajan Shankar, editors, *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 287–291. Springer-Verlag, 2006.

[10] Bernhard Ganter. Two basic algorithms in concept analysis. Preprint 831, Fachbereich Mathematik, TU Darmstadt, Darmstadt, Germany, 1984.

[11] Bernhard Ganter. Attribute exploration with background knowledge. *Theoretical Computer Science*, 217(2):215–233, 1999.

[12] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, New York, 1997.

[13] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[14] S. O. Kuznetsov. On the intractability of computing the duquenne-guigues base. *Journal of Universal Computer Science*, 10(8):927–933, 2004.

[15] Susanne Prediger. Logical scaling in formal concept analysis. In Dickson Lukose, Harry S. Delugach, Mary Keeler, Leroy Searle, and John F. Sowa, editors, *International Conference on Conceptual Structures (ICCS '97)*, volume 1257 of *LNCS*, pages 332–341. Springer, 1997.

[16] Sebastian Rudolph. *Relational Exploration: Combining Description Logics and Formal Concept Analysis for Knowledge Specification*. PhD thesis, Technische Universität Dresden, 2006.

[17] K.A. Spackman, K.E. Campbell, and R.A. Cote. SNOMED RT: A reference terminology for health care. *J. of the American Medical Informatics Association*, pages 640–644, 1997. Fall Symposium Supplement.

[18] Gerd Stumme. Attribute exploration with background implications and exceptions. In H.-H. Bock and W. Polasek, editors, *Data Analysis and Information Systems*, page 457ff, Berlin, 1996. Springer.

[19] Gerd Stumme, Technische Hochschule Darmstadt, and Fachbereich Mathematik. Exploration tools in formal concept analysis. In *Opitz (Eds.): Ordinal and Symbolic Data Analysis. Proc. OSDA'95. Studies in Classification, Data Analysis, and Knowledge Organization 8*, pages 31–44. Springer, 1996.

[20] The Gene Ontology Consortium. Gene Ontology: Tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.