RELATIONAL REPRESENTATION OF $\mathcal{ALN}$ KNOWLEDGE BASES

Thomas Studer[*]

ABSTRACT

The retrieval problem for a knowledge base O and a concept C is to find all individuals a such that O entails $C(a)$. We describe a method to represent a knowledge base O as an instance $\widehat{O}$ of a relational database. To answer a retrieval query over O, we can execute a corresponding database query over $\widehat{O}$. The main problem of such an embedding is that the semantics of knowledge bases is characterized by an open world assumption while traditional databases feature a closed world semantics. Our procedure is sound and complete for knowledge bases given in the description logic $\mathcal{ALN}$.

KEYWORDS

Database and knowledge base systems, description logic, retrieval queries.

1. INTRODUCTION

One of the main reasoning tasks for knowledge base systems is the *retrieval problem* (Baader & Nutt 2003, Vitoriá et al. 1995): given a knowledge base O and a concept description C, find all individuals a such that O entails $C(a)$. That is, given O and C, we are looking for all individuals a for which we can logically infer from O that a belongs to C. There is a trivial algorithm for this problem, namely to test for each individual occurring in O whether it is an instance of the concept C. Of course, such an algorithm is in general not feasible. Thus, special storage and reasoning techniques are needed in order to efficiently answer retrieval queries.

Description Logics are expressive languages for knowledge representation (Baader et al. 2003). They deal with unary predicates (concepts), representing collections of individuals, and binary predicates (roles). A description logic knowledge base consists of a terminology, which states relationships between concepts, and of a world description, which makes assertions about individuals. There are a number of systems available which support reasoning and persistent storage of such knowledge bases, for instance Sesame (Broekstra et al. 2002), DLDB (Pan & Heflin 2003) and knOWLer (Ciorascu et al. 2003). Often, such systems are implemented on a common RDBMS and a description logic reasoner like FaCT (Horrocks 1998) or Racer (Haarslev & Möller 2001). However, concerning retrieval queries, these systems either work with the generate and test method or do not yield complete answers.

We show that for a knowledge base O that is given in the description logic language $\mathcal{ALN}$, it is possible to compute an instance $\widehat{O}$ of a relational database which represents O with respect to the retrieval problem. That is, for every concept description C, we can run a database query on the instance $\widehat{O}$ which returns exactly the individuals belonging to C in O. Hence, we do not need a special description logic reasoning engine to answer retrieval queries. Reasoning only takes place in the initial computation of the database instance. This instance is built by performing a kind of completion algorithm.

The completion algorithm calculates the extensions of the base concepts which occur in the knowledge base. That means, for each base concept A, we determine the set of all individuals belonging to A. Thus, we can represent the extensions of the base concepts and roles of the knowledge base as instance of a relational database. The retrieval query for a complex concept description C can now be answered by a simple database

[*]Institut für Informatik und angewandte Mathematik, Neubrückstrasse 10, CH-3012 Bern, Switzerland, `tstuder@iam.unibe.ch`

query on this instance.

The main technical difficulty in representing a description logic knowledge base as relational database instance is that description logics are characterized by an *open world assumption* whereas databases work with a *closed world semantics*. As a consequence, absence of information in a database is interpreted as negative information while absence of information in description logics only means lack of knowledge. For instance, consider a knowledge base with a `hasChild` role that contains only one tuple, say `(Peter, Harry)`. Still, we are not allowed to infer $\leq 1.\texttt{hasChild}(\texttt{Peter})$ from this knowledge base because there are models where the interpretation of `hasChild` contains more tuples. We cannot identify negative information of our knowledge base with missing information in its relational representation; instead, we have to introduce special relations and terms in order to deal with negative information in the database instance.

Vitoriá et al. (1995) present an approach to the retrieval problem for $\mathcal{ALN}$ which is based on finite automata. Similar to our work, they perform a preprocessing step which produces a completion of the knowledge base. However, they build a finite automaton instead of constructing a database instance. This automaton can then be used to answer retrieval queries in a syntax-directed way.

Although $\mathcal{ALN}$ is not a very expressive description logic language, it still suffices for many applications. As Gasdoué and Rousset notice, ontology designers often do not even exploit the full expressive power of $\mathcal{ALN}$ to model their application domain. In the context of web ontology languages, Antoniou and van Harmelen (2004) also state that there are reasons to expect that most ontological knowledge will be of a rather simple nature. Of course, these points are arguable.

2. PRELIMINARIES

In this section, we first present syntax and semantics of the description logic $\mathcal{ALN}$. In a second part, we introduce $\mathcal{ALN}$ knowledge bases which consist of a terminology and a world description.

The syntax of $\mathcal{ALN}$ is given by the following rule, where A, B are used for *atomic concepts*, R denotes a *role* and C, D stand for *concept descriptions*:

$$\begin{array}{rll} C,D \rightarrow & A \mid & \text{(atomic concept)} \\ & \top \mid \bot \mid & \text{(top, bottom)} \\ & \neg A \mid & \text{(atomic negation)} \\ & C \sqcap D \mid & \text{(conjunction)} \\ & \forall R.C \mid & \text{(value restriction)} \\ & \leq n.R \mid \geq n.R & \text{(number restrictions).} \end{array}$$

The semantics of concept descriptions is given as follows. An *interpretation* I is a pair $(\Delta^I, \cdot^I)$ where Δ^I is a non-empty set of individuals called the domain of the interpretation and $\cdot^I$ is an interpretation function assigning to each atomic concept A a set $A^I \subset \Delta^I$ and to each role R a binary relation $R^I \subset \Delta^I \times \Delta^I$. The interpretation function is extended to concept descriptions by the following inductive definition:

$$\begin{array}{rcl} \top^I & := & \Delta^I, \quad \bot^I := \emptyset \\ (\neg A)^I & := & \Delta^I \setminus A^I \\ (C \sqcap D)^I & := & C^I \cap D^I \\ (\forall R.C)^I & := & \{a \in \Delta^I \mid \forall b((a,b) \in R^I \rightarrow b \in C^I)\} \\ (\leq n.R)^I & := & \{a \in \Delta^I \mid \sharp\{b \mid (a,b) \in R^I\} \leq n\} \\ (\geq n.R)^I & := & \{a \in \Delta^I \mid \sharp\{b \mid (a,b) \in R^I\} \geq n\} \end{array}$$

where $\sharp S$ denotes the cardinality of the set S.

A *knowledge base* O consists of a terminology $O_{\mathcal{T}}$ (called the TBox) and a world description $O_{\mathcal{A}}$ (called the ABox). In this paper we will only consider finite knowledge bases.

The *terminology* $O_{\mathcal{T}}$ contains concept definitions and restricted inclusion statements. A *concept definition* is an expression of the form $A := C$ where A is an atomic concept and C is a concept description. We assume that an atomic concept appears on the left hand side of at most one concept definition. That is, the set of definitions is unequivocal.

We divide the atomic concepts occurring in $\mathcal{O}_{\mathcal{T}}$ into two sets, the *name symbols* that occur on the left-hand side of some concept definition, and the *base symbols* that occur only on the right-hand side of the concept definitions.

Let A and B be atomic concepts. We say A depends on B if B appears in the concept definition of A. A set of concept definitions is called *acyclic* if there is no cycle in this dependency relation. For the rest of the paper we will only consider acyclic TBoxes.

An acyclic set of definitions can be unfolded by the following iterative process: each occurrence of a name symbol on the right-hand side of a definition is replaced by the concept that it stands for. Since there is no cycle in the set of definitions, this process eventually stops. We end up with a terminology consisting solely of definitions of the form $A := C'$ where C' contains only base symbols and no name symbols.

Since we work in $\mathcal{ALN}$, we need an additional constraint to make this unfolding possible. We assume that no name symbol occurs negatively in $\mathcal{O}$. Otherwise we could have a terminology consisting of $A := \neg B$ and $B := C \sqcap D$. This would result in $A := \neg(C \sqcap D)$ which is not an $\mathcal{ALN}$ concept description.

The allowed *inclusion statements* are of the form $A \sqsubseteq B$ and $A \sqcap B \sqsubseteq \bot$ where A and B are base symbols, that is atomic concepts for which no definition is included in the TBox. The inclusion statements of the TBox have to be acyclic. There is no chain $L_0, L_1, L_2, \ldots, L_n$ such that L_0 is the same concept as L_n and all statements $L_i \sqsubseteq L_{i+1}$ are included in the TBox.

An interpretation $\mathcal{I}$ satisfies $\mathcal{O}_{\mathcal{T}}$ if for every inclusion $C \sqsubseteq D$ in $\mathcal{O}_{\mathcal{T}}$ we have $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, and if for every definition $A := C$ in $\mathcal{O}_{\mathcal{T}}$ we have $A^{\mathcal{I}} = C^{\mathcal{I}}$.

A *world description* $\mathcal{O}_{\mathcal{A}}$ is a set of assertions about the extension of concepts and roles. In the ABox we introduce a set of individual constants $a, b, c, \ldots$ and we assert properties of these individuals. We can make the following two kinds of assertions in an ABox: $C(a)$ and $R(b, c)$ where C is a concept and R is a role. The first kind, called *concept assertion*, states that a belongs to (the interpretation of) C. The second kind, called *role assertion*, states that b is related to c by R.

We give semantics for the ABox by extending interpretations to individual constants as follows. Let $\mathcal{I}$ be $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where the interpretation function $\cdot^{\mathcal{I}}$ not only maps atomic concepts and roles to sets and relations, but in addition maps each individual constant a to an element $a^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$. We assume that distinct individual constants denote distinct objects in $\Delta^{\mathcal{I}}$. Therefore, the mapping $\cdot^{\mathcal{I}}$ respects the *unique name assumption*, that is if a and b are different individual constants, then we have $a^{\mathcal{I}} \neq b^{\mathcal{I}}$.

The interpretation $\mathcal{I}$ satisfies the concept assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and if $(b^{\mathcal{I}}, c^{\mathcal{I}}) \in R^{\mathcal{I}}$ then it satisfies the role assertion $R(b, c)$. An interpretation satisfies an ABox $\mathcal{O}_{\mathcal{A}}$ if it satisfies each assertion in it. We call an interpretation $\mathcal{I}$ a *model* of a knowledge base $\mathcal{O}$, if $\mathcal{I}$ satisfies $\mathcal{O}_{\mathcal{T}}$ and $\mathcal{O}_{\mathcal{A}}$. We will use the following notation: $\mathcal{I} \models \mathcal{O}$ means that $\mathcal{I}$ is a model of the knowledge base $\mathcal{O}$. We call a knowledge base $\mathcal{O}$ *satisfiable* if there exists an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{O}$. We say $\mathcal{O}$ *entails* $C(a)$, formally $\mathcal{O} \models C(a)$, if every model $\mathcal{I}$ of $\mathcal{O}$ satisfies $C(a)$. Similarly, $\mathcal{O} \models R(a, b)$ states that every model $\mathcal{I}$ of $\mathcal{O}$ satisfies $R(a, b)$. By $\mathcal{O} \not\models C(a)$ and $\mathcal{O} \not\models R(a, b)$ we mean that there exists a model $\mathcal{I}$ of $\mathcal{O}$ which does not satisfy $C(a)$ and $R(a, b)$, respectively.

Example 1. Let us have a look at two examples of entailment.

- Let $\mathcal{O}$ be a knowledge base which contains the assertions $\leq 1.R(a)$, $R(a, b)$ and $C(b)$. We find that $\mathcal{O} \models \forall R.C(a)$.
- Let $\mathcal{O}$ be a knowledge base which contains $\forall R.A(a)$, $\forall R.\neg B(a)$ and having $A \sqsubseteq B$ in its TBox. We find that $\mathcal{O} \models \leq 0.R(a)$.

3. BUILDING THE DATABASE INSTANCE

We are going to present the construction of the database instance representing an $\mathcal{ALN}$ knowledge base. We will consider *normalized* knowledge bases. The normalization of a knowledge base $\mathcal{O}$ is achieved by the following steps:

1. Add the inclusion statement $\neg B \sqsubseteq \neg A$ for each $A \sqsubseteq B$ in $\mathcal{O}$.
2. Replace each statement $A \sqcap B \sqsubseteq \bot$ by the two inclusion statements $A \sqsubseteq \neg B$ and $B \sqsubseteq \neg A$.

3. If there is a chain $L_0, L_1, L_2, \ldots, L_n$ of concepts such that L_n is the negation of L_0 and all statements $L_i \sqsubseteq L_{i+1}$ are included in $\mathcal{O}$, then add $L_0 \sqsubseteq \bot$.

4. If there is a concept L such that $L \sqsubseteq \bot$ is an inclusion statement of $\mathcal{O}$, then replace L with $\bot$ in $\mathcal{O}$.

5. Fully unfold the concept definitions of $\mathcal{O}$, so that no name symbol occurs on the right-hand side of a concept definition. Then we exhaustively apply the following normalization rule

$$\forall R.(C \sqcap D) \rightarrow \forall R.C \sqcap \forall R.D.$$

Now replace each assertion $C(a) \in \mathcal{O}$ by $C'(a)$ where C' is built from C by substituting each name symbol with its (unfolded and normalized) definition.

Observe that in a normalized knowledge base, all atomic concepts are base symbols. If an atomic concept has a definition, then it has been replaced by its definition during the process of normalization. Hence, the assertional part of a normalized knowledge base contains only concept descriptions built from base symbols. In the sequel, $\mathcal{O}$ will always denote such a normalized knowledge base.

Given a knowledge base $\mathcal{O}$, let $\mathcal{C}$ be the set of all individuals occurring in $\mathcal{O}$. Now we define a set $\widehat{\mathcal{C}}$ of individual constants which will be the domain of our database instance. We need some auxiliary notions. The *role depth* $\mathsf{rd}(C)$ of a concept C is given by: $\mathsf{rd}(\top) = \mathsf{rd}(\bot) = \mathsf{rd}(A) = \mathsf{rd}(\neg A) = 0$, $\mathsf{rd}(C \sqcap D) = \max(\mathsf{rd}(C), \mathsf{rd}(D))$, $\mathsf{rd}(\forall R.C) = \mathsf{rd}(C) + 1$ and $\mathsf{rd}(\leq n.R) = \mathsf{rd}(\geq n.R) = 1$.

Let $\mathsf{rd}(\mathcal{O})$ be the maximum of all $\mathsf{rd}(C)$ for concept expressions C occurring in $\mathcal{O}$. Let k be the biggest natural number n such that a number restriction $\leq n.R$ or $\geq n.R$ occurs in $\mathcal{O}$. The set $\widehat{\mathcal{C}}$ is given by:

- every individual constant $a \in \mathcal{C}$ is included in $\widehat{\mathcal{C}}$. We define $\mathsf{rd}(a) := 0$ for $a \in \mathcal{C}$.

- new constants $\exists_{R,a,i}$ are added to $\widehat{\mathcal{C}}$ for all role names R, for all i with $1 \leq i \leq k$ and for all $a \in \widehat{\mathcal{C}}$ with $\mathsf{rd}(a) < \mathsf{rd}(\mathcal{O})$. We define $\mathsf{rd}(\exists_{R,a,i}) := \mathsf{rd}(a) + 1$.

- new constants $\forall_{R,a}$ are added to $\widehat{\mathcal{C}}$ for all role names R and for all $a \in \widehat{\mathcal{C}}$ with $\mathsf{rd}(a) < \mathsf{rd}(\mathcal{O})$. We define $\mathsf{rd}(\forall_{R,a}) := \mathsf{rd}(a) + 1$.

Note that since $\mathcal{O}$ is finite, $\mathcal{C}$ is finite. Thus, $\widehat{\mathcal{C}}$ is also finite because of the role depth restrictions in its definition.

Next, we are going to define the relation $\mathcal{O} \vdash_{\alpha} s$ where s is an ABox assertion and α is a natural number. The intended meaning of $\mathcal{O} \vdash_{\alpha} s$ is "s follows from $\mathcal{O}$ in at most α steps". We will employ this relation to compute the atomic concept assertions which are entailed by $\mathcal{O}$. They will serve as a basis to build the database instance we are aiming at.

1. $\mathcal{O} \vdash_{0} C(a)$ if $C(a) \in \mathcal{O}$.

2. $\mathcal{O} \vdash_{0} \top(a)$, for all $a \in \widehat{\mathcal{C}}$.

3. $\mathcal{O} \vdash_{0} \neg A(a)$, for all $a \in \widehat{\mathcal{C}}$ if $A \sqsubseteq \bot \in \mathcal{O}$.

4. $\mathcal{O} \vdash_{0} R(a,b)$ if $R(a,b) \in \mathcal{O}$.

5. $\mathcal{O} \vdash_{0} R(a, \forall_{R,a})$ if $\forall_{R,a} \in \widehat{\mathcal{C}}$.

6. $\mathcal{O} \vdash_{\alpha+1} C(a)$ if $\mathcal{O} \vdash_{\alpha} C \sqcap D(a)$ or $\mathcal{O} \vdash_{\alpha} D \sqcap C(a)$.

7. $\mathcal{O} \vdash_{\alpha+1} C(a)$ if there exists a constant $b \in \widehat{\mathcal{C}}$ such that $\mathcal{O} \vdash_{\alpha} \forall R.C(b)$ and $\mathcal{O} \vdash_{\alpha} R(b,a)$.

8. $\mathcal{O} \vdash_{\alpha+1} C(a)$ if $\mathcal{O} \vdash_{\alpha} D(a)$ and $D \sqsubseteq C \in \mathcal{O}$.

9. $\mathcal{O} \vdash_{\alpha+1} R(a, \exists_{R,a,i})$ if the following two conditions hold:
   - $\mathcal{O} \vdash_{\alpha} \geq n.R(a)$

   - $1 \leq i \leq n - \sharp\{x \in \mathcal{C} \mid R(a,x) \in O\}$.

10. $O \vdash_{\alpha+1} \forall R.C(a)$ if $O \vdash_{\alpha} \forall R.D(a)$ as well as $D \sqsubseteq C \in O$.
11. $O \vdash_{\alpha+1} \leq 0.R(a)$ if we have $O \vdash_{\alpha} \forall R.A(a)$ and $O \vdash_{\alpha} \forall R.\neg A(a)$ for some base concept A.
12. $O \vdash_{\alpha+1} \leq 0.R(a)$ if we have $O \vdash_{\alpha} \forall R.{\leq} n.R_2(a)$ and $O \vdash_{\alpha} \forall R.{\geq} m.R_2(a)$ for some role R_2 and natural numbers n and m with $n < m$.
13. $O \vdash_{\alpha+1} \leq 0.R(a)$ if $O \vdash_{\alpha} \forall R.\bot(a)$.
14. $O \vdash_{\alpha+1} C(a)$ if $O \vdash_{\alpha} C(a)$.
15. $O \vdash_{\alpha+1} R(a,b)$ if $O \vdash_{\alpha} R(a,b)$.

We will write $O \vdash C(a)$ and $O \vdash R(a,b)$ for $\exists\alpha.O \vdash_{\alpha} C(a)$ and $\exists\alpha.O \vdash_{\alpha} R(a,b)$, respectively.
Let us briefly discuss the different cases in the definition of $O \vdash_{\alpha} s$.

- Cases 1 to 4 deal with assertions directly included in O.
- In case 5 we include special constants $\forall_{R,a}$ to the extension of a role R. If we later get that $O \vdash \forall R.C(a)$, then this implies $O \vdash C(\forall_{R,a})$ (see case 7). Our database instance will be defined such that if $\forall_{R,a}$ is in the answer to a query about the concept C, then O entails $\forall R.C(a)$.
- Cases 6 to 8 compute the immediate consequences of assertions of the form $C \sqcap D(a)$, $\forall R.C(a)$ and $C \sqsubseteq D$.
- Case 9 treats the consequences of $\geq n.R(a)$. Such an assertion demands the existence of a certain number of elements of the form (a,x) in R. However, it does not impose any condition on what x has to be. So we can simply include new individual constants $\exists_{R,a,i}$ in order to witness these existential commitments. We only have to take care that we do not include too many new elements, as this could contradict a statement of the form $\leq n.R(a) \in O$.
- Cases 10 to 13 deal with the relationship between statements of the form $\leq 0.R(a)$ and the constructor for value restriction $\forall R$ (see Example 1).
- Finally, cases 14 and 15 are included to make the definition monotone with respect to α.

Note that the algorithm does nothing in the case $\leq n.R(a) \in O$. An assertion of this form does not enforce the existence of more elements in the interpretation of R. However, it may imply facts of the form $\forall R.C(a)$. This will be reflected in Definition 3 where $(a, \forall_{R,a})$ may be included in the extension of the complement $\overline{R}$ of R.
The following theorem states that the computation of $O \vdash s$ terminates. It can easily be seen by a simple cardinality argument.

Theorem 2. *Assume O is finite. There exists a natural number α depending on O such that*

1. $O \vdash C(a) \Leftrightarrow O \vdash_{\alpha} C(a)$,
2. $O \vdash R(a,b) \Leftrightarrow O \vdash_{\alpha} R(a,b)$.

Definition 3. We define the relational database instance $\widehat{O}$ by:

- $A(a)$ is included in $\widehat{O}$ if $O \vdash A(a)$ for a base symbol A,
- $\overline{A}(a)$ is included in $\widehat{O}$ if $O \vdash \neg A(a)$ for a base symbol A,
- $R(a,b)$ is included in $\widehat{O}$ if $O \vdash R(a,b)$ and $b \neq \forall_{R,a}$,

- $\overline{R}(a,b)$ is included in $\widehat{O}$ if $O \vdash {\leq}n.R(a)$ and b is one of $s_1,\ldots,s_m$ which are defined as follows. Let $S := \{x \in \widehat{C} \mid O \nvdash R(a,x)\}$. Let $s_1,\ldots,s_l$ be an enumeration of S. Let k be the least natural number such that $O \vdash {\leq}k.R(a)$. The index m is defined as $\sharp\widehat{C} - k$. If the cardinality l of S is only $\sharp\widehat{C} - k - 1$, then we let s_m be the constant $\forall_{R,a}$.

The relations $\overline{A}$ and $\overline{R}$ in $\widehat{O}$ are needed in order to deal with negated concepts and negative information about roles.

4. SOUNDNESS AND COMPLETENESS

We introduce a special consequence relation $\widehat{O} \models_{\mathsf{DB}} C(a)$ for our database instance $\widehat{O}$ and $\mathcal{ALN}$ concept descriptions C. This gives us a translation of description logic retrieval queries to queries on our relational database instance.

Definition 4. The consequence relation $\models_{\mathsf{DB}}$ is defined by:

1. $\widehat{O} \models_{\mathsf{DB}} A(a)$ if $A(a) \in \widehat{O}$,
2. $\widehat{O} \models_{\mathsf{DB}} \neg A(a)$ if $\overline{A}(a) \in \widehat{O}$,
3. $\widehat{O} \models_{\mathsf{DB}} C \sqcap D(a)$ if $\widehat{O} \models_{\mathsf{DB}} C(a)$ and $\widehat{O} \models_{\mathsf{DB}} D(a)$,
4. $\widehat{O} \models_{\mathsf{DB}} \forall R.C(a)$ if $\widehat{O} \models_{\mathsf{DB}} C(\forall_{R,a})$ or $\overline{R}(a,\forall_{R,a}) \in \widehat{O} \wedge \forall x.(R(a,x) \in \widehat{O} \rightarrow \widehat{O} \models_{\mathsf{DB}} C(x))$,
5. $\widehat{O} \models_{\mathsf{DB}} {\geq}n.R(a)$ if $\sharp\{x \mid R(a,x) \in \widehat{O}\} \geq n$,
6. $\widehat{O} \models_{\mathsf{DB}} {\leq}n.R(a)$ if $\exists x \in \widehat{C}.\overline{R}(a,x) \in \widehat{O}$ and $\sharp\{x \mid \overline{R}(a,x) \in \widehat{O}\} \geq \sharp\widehat{C} - n$.

By a canonical model construction, we obtain that the relational representation is complete with respect to $\mathcal{ALN}$.

Theorem 5. *Let O be a satisfiable knowledge base. We have*

$$O \models C(a) \Rightarrow \widehat{O} \models_{\mathsf{DB}} C(a).$$

Now we are going to show soundness of our database instance. Every statement we can infer from the database instance is a consequence of the knowledge base. In order to prove soundness, we need a series of lemmas and definitions. All the lemmas are shown by induction on the length α of $O \vdash_{\overline{\alpha}}$ derivations.

Lemma 6. *Assume a,b are individuals of $\mathcal{C}$.*

1. $\forall\alpha(O \vdash_{\overline{\alpha}} C(a) \Rightarrow O \models C(a))$
2. $\forall\alpha(O \vdash_{\overline{\alpha}} R(a,b) \Rightarrow O \models R(a,b))$

We need some auxiliary definitions to cope with terms of the form $\forall_{R,a}$.

Definition 7. The * *unfolding* of concept assertions is given by:

- $C(s)^* := C(S)$ if s is not of the form $\forall_{R,t}$ for some R and t,
- $C(s)^* := (\forall R.C(t))^*$ if $s \equiv \forall_{R,t}$ for some R and t.

Definition 8. The function basis determines the individual constant occurring in the * unfolding of a concept assertion. It is defined by:

- $\mathsf{basis}(a) := a$ if $a \in \mathcal{C}$ or a is of the form $\exists_{R,b,i}$ for some R and b,
- $\mathsf{basis}(a) := \mathsf{basis}(b)$ if a is of the form $\forall_{R,b}$ for some R and b.

Lemma 9. *Assume* $\mathsf{basis}(\forall_{R,a}) \in \mathcal{C}$*. Then we have* $O \vdash C(\forall_{R,a}) \Rightarrow O \models C(\forall_{R,a})^*$.

Lemma 10. *Let* $\exists_{R,a,i}$ *be an element of* $\widehat{\mathcal{C}}$*. We have* $O \vdash C(\exists_{R,a,i}) \Rightarrow O \vdash C(\forall_{R,a})$.

Now we can state the soundness theorem for the database instance that we have constructed. We want to show $\widehat{O} \models_{\mathsf{DB}} C(a) \Rightarrow O \models C(a)$ for all $a \in \mathcal{C}$. In order to prove this statement, we have to formulate it more generally. This is done in the following theorem.

Theorem 11. *Assume* $\mathsf{basis}(a) \in \mathcal{C}$*. We have*

$$\widehat{O} \models_{\mathsf{DB}} C(a) \Rightarrow O \models C(a)^*.$$

Proof. We show the statement by induction on the structure of C. We have the following cases.

- If C is a base symbol, we get $C(a) \in \widehat{O}$. That is $O \vdash C(a)$. Using Lemmas 6 and 9, we obtain $O \models C(a)^*$.
- If C is of the form $\neg D$, then we find $O \vdash C(a)$. Again by Lemmas 6 and 9, we obtain $O \models C(a)^*$.
- Assume C is of the form $D \sqcap E$. By the induction hypothesis we obtain $O \models D(a)^*$ as well as $O \models E(a)^*$. Therefore $O \models D \sqcap E(a)^*$ holds.
- Assume C is of the form $\forall R.D$. We have the following two possibilities given by the definition of the $\models_{\mathsf{DB}}$ relation.
  - $\widehat{O} \models_{\mathsf{DB}} D(\forall_{R,a})$: We apply the induction hypothesis to infer $O \models D(\forall_{R,a})^*$. By the * unfolding we get $O \models (\forall R.D(a))^*$ which is the same as $O \models C(a)^*$.
  - $\widehat{O} \not\models_{\mathsf{DB}} D(\forall_{R,a})$: Then we have

$$\overline{R}(a, \forall_{R,a}) \in \widehat{O} \text{ and} \tag{1}$$

$$\forall x.(R(a,x) \in \widehat{O} \to \widehat{O} \models_{\mathsf{DB}} D(x)). \tag{2}$$

    By Definition 3 we infer from (1) that $\leq k.R(a) \in O$ where k is the least number n such that $\leq n.R(a) \in O$. Additionally, there exist k many individuals $b_j \in \widehat{\mathcal{C}}$ such that $R(a,b_j) \in \widehat{O}$. Since $\widehat{O} \not\models_{\mathsf{DB}} D(\forall_{R,a})$, we know by (2) that $b_j \neq \forall_{R,a}$ for all those b_j. Moreover, Lemma 10 implies by (2) that $b_j \neq \exists_{R,a,i}$. Therefore $b_j \in \mathcal{C}$ and hence $a \in \mathcal{C}$ by the definition of $O \vdash_{\overline{\alpha}} R(a,b)$. Summing up, we have $O \models {\leq} k.R(a)$ as well as $O \models R(a,b_j)$ and $O \models D(b_j)$ for the k many individuals $b_j \in \mathcal{C}$. Taking these facts together we finally obtain $O \models \forall R.D(a)$. That is $O \models C(a)^*$ since $a \in \mathcal{C}$.
- Assume C is of the form $\geq n.R$, again we have two cases:
  - Assume there are n many individuals $b_1, \dots, b_n \in \mathcal{C}$ such that $R(a,b_i) \in \widehat{O}$. Note that this implies $a \in \mathcal{C}$. Then we have $R(a,b_i) \in O$ for $1 \leq i \leq n$. Hence, we also have $O \models R(a,b_i)$ for $1 \leq i \leq n$. Therefore, we conclude $O \models {\geq} n.R(a)$ which is $O \models {\geq} n.R(a)^*$.
  - If there are only $m < n$ many individual constants $b_i \in \mathcal{C}$ with $R(a,b_i) \in \widehat{O}$, then $\widehat{O} \models_{\mathsf{DB}} {\geq} n.R$ implies that there are at least $n-m$ many constant $\exists_{R,a,i}$ such that $R(a, \exists_{R,a,i}) \in \widehat{O}$. This only is possible if $O \vdash {\geq} n.R(a)$ holds. We conclude $O \models {\geq} n.R(a)^*$ by Lemmas 6 and 9.
- Assume C is of the form $\leq n.R$. We have $\sharp\{x \mid \overline{R}(a,x) \in \widehat{O}\} \geq \sharp\widehat{\mathcal{C}} - n$ and there exists an individual x such that $\overline{R}(a,x) \in \widehat{O}$. Then by Definition 3 there exists a $k \leq n$ such that $O \vdash {\leq} k.R(a)$. This implies $O \models {\leq} k.R(a)^*$. Because $k \leq n$, we can finally conclude $O \models {\leq} n.R(a)^*$ which is $O \models C(a)^*$. □

Corollary 12. *Let O be a satisfiable knowledge base, C be a concept description of O, a be an individual of $\mathcal{C}$ and $\widehat{O}$ be the database instance as above. Then we have*

$$O \models C(a) \iff \widehat{O} \models_{\mathsf{DB}} C(a).$$

5. DISCUSSION AND CONCLUSION

We have considered the retrieval problem for knowledge bases which are formulated in the description logic language $\mathcal{ALN}$. We have shown that it is possible to represent such a knowledge base $\mathcal{O}$ as relational database instance $\widehat{\mathcal{O}}$. A description logic retrieval query can be translated to a corresponding relational database query over $\widehat{\mathcal{O}}$ which provides exactly the answers to the original query over $\mathcal{O}$. That is, our procedure is sound and complete.

Let us discuss the complexity of our relational knowledge base representation. First, we have defined the domain $\widehat{\mathcal{C}}$ of our database instance. We need as many new constants as the biggest number k which occurs in a number restriction. Given a binary encoding of numbers, this means that the size of our domain is exponential in the size of the original knowledge base. In Theorem 2, the closure ordinal α is polynomial in the size of $\mathcal{O}$ and $\widehat{\mathcal{C}}$. This can be seen by a simple cardinality argument. Donini et al. (1994) have shown that the retrieval problem for $\mathcal{ALN}$ is solvable in polynomial time. Our approach does not yield a polytime solution since $\widehat{\mathcal{C}}$ grows too fast. However, we need to build the database instance only once. Then several queries can be answered over this instance. Hence, the performance of query processing may be increased by using database caching mechanisms to reuse intermediate results. Of course, this needs to be experimentally verified.

We are looking for a relational representation in which we do need that many new individuals for the database domain. This could then yield a polytime solution to the $\mathcal{ALN}$ retrieval problem in our framework. Further directions for continuing our work are database representations of more expressive description logics as well as optimization techniques for retrieval queries. Last but not least, the impact of updating a knowledge base on its relational representation is an important point for future research.

References

G. Antoniou & F. van Harmelen (2004). *A Semantic Web Primer*. MIT Press.

F. Baader, et al. (2003). *The Description Logic Handbook*. Cambridge.

F. Baader & W. Nutt (2003). ‘Basic Description Logic’. In F. Baader, D. Calvanese, D. McGuiness, D. Nardi, & P. Patel-Schneider (eds.), *The Description Logic Handbook*, pp. 43–95. Cambridge.

J. Broekstra, et al. (2002). ‘Sesame: A generic architecture for storing and querying RDF and RDF schema’. In *The Semantic Web - ISWC 2002*, vol. 2342 of *LNCS*, pp. 54–68. Springer.

I. Ciorascu, et al. (2003). ‘Scalable Ontology Implementation Based on knOWLer’. In *Workshop on Practical and Scaleable Semantic Web Systems, ISWC 2003*.

F. Donini, et al. (1994). ‘Deduction in Concept Languages: From Subsumption to Instance Checking’. *Journal of Logic and Computation* **4**(4):423–452.

F. Gasdoué & M.-C. Rousset (to appear). ‘Answering Queries using Views: a KRDB Perspective for the Semantic Web’. *ACM Transactions on Internet Technology*.

V. Haarslev & R. Möller (2001). ‘RACER System Description’. In *Proc. of the Int. Joint Conf. on Automated Reasoning*, vol. 2083 of *LNCS*, pp. 701–705. Springer.

I. Horrocks (1998). ‘The FaCT System’. In *Proc. of the 2nd Int. Conf. on Analytic Tableaux and Related Methods*, vol. 1397 of *LNAI*, pp. 307–312. Springer.

Z. Pan & J. Heflin (2003). ‘DLDB: Extending Relational Databases to Support Semantic Web Queries’. In *Workshop on Practical and Scaleable Semantic Web Systems, ISWC 2003*, pp. 109–113.

A. Vitória, et al. (1995). ‘The retrieval problem in a concept language with number restrictions’. In C. Pinto & N. Mamede (eds.), *Proc. of the 7th Portuguese Conf. on Artificial Intelligence*, vol. 990 of *LNAI*. Springer.