Island Reasoning for $\mathcal{ALCHI}$ Ontologies

Sebastian WANDELT and Ralf MOELLER
Hamburg University of Technology,
21079 Hamburg,
Germany

Abstract. In the last years, the vision of the Semantic Web fostered the interest in reasoning over ever larger sets of assertional statements in ontologies. It is easily conjectured that, soon, real-world ontologies will not fit into main memory anymore. If this was the case, state-of-the-art description logic reasoning systems cannot deal with these ontologies any longer, since they rely on in-memory structures.

We propose a way to overcome this problem by reducing instance checking for an individual in an ontology to a (usually small) relevant subset of assertional axioms. This subset can then be processed by state-of-the-art description logic reasoning systems to perform sound and complete instance checks for the given individual. We think that this technique will support description logic systems to deal with the upcoming large amounts of assertional data.

Keywords. Ontologies, Instance checking, Scalability

Introduction

As the Semantic Web evolves, scalability of inference techniques becomes increasingly important. Even for basic description logic-based inference techniques, e.g. concept satisfiability, it is only recently understood on how to perform reasoning on large ABoxes in an efficient way. This is not yet the case for problems that are too large to fit into main memory.

In this paper we present an approach to execute efficient instance retrieval tests on ontologies, which do not fit into main memory. Existing tableau-based description logic reasoning systems, e.g. Racer [HM01], do not perform well in such scenarios since the implementation of tableau-algorithms is usually built based on efficient in-memory structures. Our contribution is concerned with the following main objective: given an individual a and a concept description C, we want to identify a relevant subset of assertional statements, which are sufficient to decide, whether a is an instance of C or not. The situation is depicted in Figure 1. In the left part there is a graph representing the assertional facts. On the right side a relevant set of assertions is identified for reasoning about individual a. Once we obtained such a (small) subset, called *island*, it can be loaded into a description logic reasoning system for instance checking. Thus, description logic reasoners of any kind can benefit from our proposal.

The intuition is to identify a small island of assertions, s.t. no "new and complex" information can be propagated between the island and the remaining assertional part of the ontology. Although the idea seems straightforward, to the best of our knowledge we

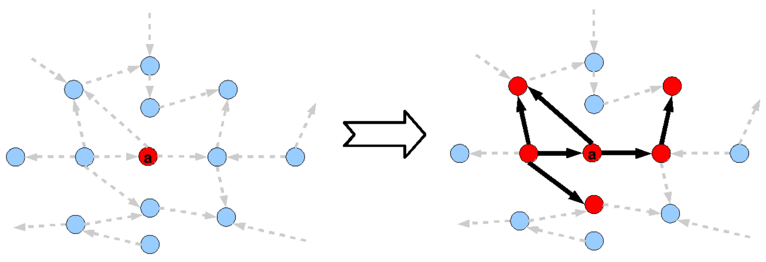


Figure 1. Example: connected subgraph relevant for reasoning about individual a

are the first to propose such an algorithm and evaluate it with respect to large ontologies. There exists previous work on partitioning/modularizing ontologies (e.g. [GH06]) and also on summarization techniques (e.g. [FKM$^+$06]). We will explain below why we think that these techniques are not sufficient in our setting. In a nutshell, the major reasons are:

1. We extract a small subset of assertions, which are worst-case relevant for individual a and concept C.
2. Our approach does not require a precomputation process depending on the ABox. Thus, it is directly applicable to ontologies where the assertional information changes over time. This can be seen as an important step towards dealing with streams of assertional information.

The remaining part of the paper is structured as follows. Section 1 provides the formal background for description logics and also presents some related work. In Section 2 we introduce an example ontology, which will be used throughout the paper. Section 3 analyzes the TBox of given ontologies and Section 4 shows how to use the result of the analysis to rewrite assertional parts of ontologies. Furthermore we we show how instance tests can be restricted to relevant subsets of the assertional information. In Section 5 we present preliminary evaluation of the proposed algorithm and provide further ideas for improvements. We conclude with Section 6.

1. Foundations

1.1. Description Logics

In the following part we will define mathematical notions, which are relevant for the remaining paper. We briefly recall syntax and semantics of the description logic $\mathcal{ALCHI}$. For the details, please refer to [BCM$^+$07]. We assume a collection of disjoint sets: a set of *concept names* N_{CN} , a set of *role names* N_{RN} and a set of *individual names* N_I. The *set of roles* N_R is $N_{RN} \cup \{R^- | R \in N_{RN}\}$. The set of *$\mathcal{ALCHI}$-concept descriptions* is given by the following grammar:

$$C, D ::= \top | \bot | A | \neg C | C \sqcap D | C \sqcup D | \forall R.C | \exists R.C$$

where $A \in N_{CN}$ and $R \in N_R$. We say that a concept description is *atomic*, if it is a concept name. With N_C we denote all atomic concepts. For defining the semantics of concept descriptions and roles we consider *interpretations* $\mathcal{I}$ that consist of a non-empty set $\Delta^{\mathcal{I}}$, the domain, and an interpretation function $\cdot^{\mathcal{I}}$, which assigns to every atomic concept description A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and to every role R a set $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. For complex concept descriptions the interpretation function is extended as shown in [BCM$^+$07]. The semantics of description logics is based on the notion of satisfiability. An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *satisfies* a concept description C if $C^{\mathcal{I}} \neq \emptyset$. In this case, $\mathcal{I}$ is called a *model* for C.

A *TBox* is a set of so-called *generalized concept inclusions*(GCIs) $C \sqsubseteq D$. An interpretation $\mathcal{I}$ *satisfies* a generalized concept inclusion $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. An interpretation is a *model* of a TBox $\mathcal{T}$ if it satisfies all generalized concept inclusions in $\mathcal{T}$. A *RBox* is a set of so-called *role inclusions* $R \sqsubseteq S$ and *role equalities assertion* $R \doteq S$. An interpretation $\mathcal{I}$ *satisfies* a role inclusion $R \sqsubseteq S$ if $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$. An interpretation $\mathcal{I}$ *satisfies* a role equality assertion $R \doteq S$ if $R^{\mathcal{I}} = S^{\mathcal{I}}$. An *ABox* is a set of so-called *concept and role assertions* $a : C$ and $R(a, b)$. An interpretation $\mathcal{I}$ *satisfies* a concept assertion $a : C$ (role assertion $R(a, b)$) if $a \in C^{\mathcal{I}}$ ($(a, b) \in R^{\mathcal{I}}$).

A *ontology* $\mathcal{O}$ consists of a 3-tuple $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox, $\mathcal{R}$ is a RBox and $\mathcal{A}$ is a ABox. We restrict the concept assertions in $\mathcal{A}$ in such a way that each concept description is an atomic concept or a negated atomic concept. This is a common assumption, e.g. in [GH06], when dealing with large assertional datasets in ontologies. With $Ind(\mathcal{A})$ we denote the set of individuals occurring in $\mathcal{A}$. We say that $\mathcal{O}$ is *inconsistent*, denoted with $INC(\mathcal{O})$, if there exists no model for $\mathcal{O}$. We say that $\mathcal{O}$ is *consistent*, denoted with $CON(\mathcal{O})$, if there exists at least one model for $\mathcal{O}$. Given an individual a and an atomic concept C, we have $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models a : C$ if and only if $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \cup \{a : \neg C\} \rangle)$.

In the following we define some additional notions, which will be used throughout the remaining part of the paper. A $\exists$-*constraint* is a concept description of the shape $\exists R.C$, s.t. C is an arbitrary concept description. A $\forall$-*constraint* is a concept description of the shape $\forall R.C$, s.t. C is an arbitrary concept description. A concept description is in *negation normal form* if negation occurs only in front of concept names. We assume the standard transformation $nnf(...)$ into negation normal form[BCM$^+$07].

The *subsumption hierarchy* of parents and children for each concept name can be obtained by classification. For $\mathcal{ALCHI}$ ontology it is possible to compute the subsumption hierarchy in advance given only the TBox $\mathcal{T}$ and RBox $\mathcal{R}$, i.e. without the ABox $\mathcal{A}$. This is possible since $\mathcal{ALCHI}$ does not allow the use of nominals. With $\sqsubseteq_{\mathcal{T}}: N_C \times N_C$ we denote the precomputed subsumption hierarchy obtained by classification, e.g. we have $\sqsubseteq_{\mathcal{T}} (C, D)$ iff $\mathcal{O} \models C \sqsubseteq D$ for atomic concepts C and D. The role hierarchy of an $\mathcal{ALCHI}$-ontology can be computed in advance given the RBox $\mathcal{R}$ only. With $\sqsubseteq_{\mathcal{R}}: N_R \times N_R$ we denote the precomputed role hierarchy, e.g. we have $(R, S) \in \sqsubseteq_{\mathcal{R}}$ iff $\mathcal{O} \models R \sqsubseteq S$ for roles R and S.

Because of space limitations we do not introduce the notion of tableau proofs w.r.t. description logics, but refer to [BCM$^+$07].

1.2. Related Work

In the following, we discuss selected previously published work related to approximate reasoning and query answering optimization for description logics. Recently, an approach for partitioning large OWL ontologies has been presented in [GH06]. The idea is to partition a large ABox into smaller ABoxes, s.t. reasoning on the smaller assertional subsets is complete, but possibly unsound. Although the authors report impressive results for the increase in performance, we see some problems:

1. To perform instance checking for a particular individual, one has to evaluate all existing partitions. Thus, one ends up loading the whole ontology into memory step-by-step.
2. In fact, the reported average partition size can be orders of magnitudes larger in real-world conditions. Some of these conditions are shown in [Wan08].
3. The approach is not subject to updateable ontologies, i.e. the recomputation of the partitions takes up to several hours/days.

In [FKM$^+$06], the authors propose a method to reduce the number of individuals in an ABox for complete but possibly unsound reasoning. Afterwards, a filtering algorithm is applied to obtain soundness. The idea is to join/summarize similar individuals into one individual and then perform reasoning. This approach is excellent for working with a compact representation of the whole ontology. However, in our setting we are only concerned with these parts of an ontology, which are relevant for a particular (given) individual. In a similar way as the approach given in [GH06], a Summary ABox has to be build in a precomputation step, which depends on the actual ABox. Thus, the approach is not per-se applicable to updateable ontologies.

There is different related work on scalability of query answering by approximation. However, since our work does not involve approximation, we do not discuss these approaches here. After all, our work can be seen as complementary to other work. For more information refer to Section 5. Finally, we should mention the work on QuOnto[ACG$^+$05], which has been the first description logic reasoner that does not use main memory at all to perform ABox reasoning. Their approach rests on the reduced expressiveness of the description logic $DL - Lite$ and can transform queries over ontologies into equivalent and more efficient queries over databases.

2. Guiding example

In the following we define an example ontology, which is used throughout the remaining part of the paper. The ontology is inspired by LUBM [GPH05], a benchmark-ontology in the setting of universities. Although this is a synthetic benchmark, several (if not most) papers on scalability of ontological reasoning consider it as a base reference. We take a particular a snapshot from the LUBM-ontology (TBox, RBox and ABox) and adapt it for presentation purposes. Please note that we do not claim that our snapshot is representative for LUBM. We evaluate our approach w.r.t. to “full” LUBM in Section 5.

Example 1. *Let* $\mathcal{O}_{EX} = \langle \mathcal{T}_{EX}, \mathcal{R}_{EX}, \mathcal{A}_{EX} \rangle$, *s.t.*

$$
\begin{aligned}
\mathcal{T}_{EX} =\{ & \\
& Chair \equiv \exists headOf.Department \sqcap Person, Professor \sqsubseteq Faculty, Book \sqsubseteq Publication, \\
& GraduateStudent \sqsubseteq Student, Student \equiv Person \sqcap \exists takesCourse.Course, \\
& \top \sqsubseteq \forall teacherOf.Course, \exists teacherOf.\top \sqsubseteq Faculty, Faculty \sqsubseteq Person, \\
& \top \sqsubseteq \forall publicationAuthor^{-}.(Book \sqcup ConferencePaper) \\
& \} \\
\mathcal{R}_{EX} =\{ & headOf \sqsubseteq worksFor, worksFor \sqsubseteq memberOf, memberOf \doteq member^{-}\} \\
\mathcal{A}_{EX} = & \texttt{see Figure 2}
\end{aligned}
$$

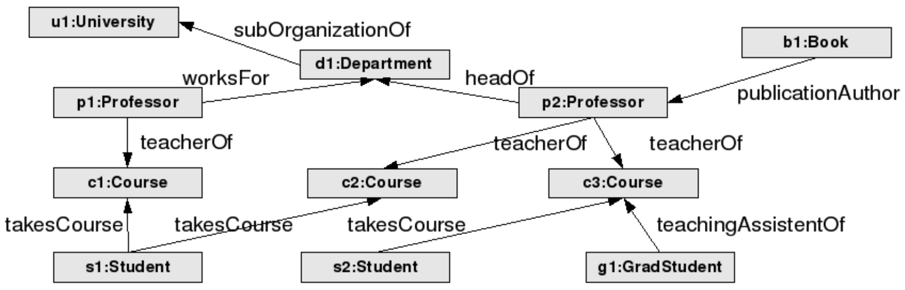


Figure 2. Guiding Example: ABox $\mathcal{A}_{EX}$ for ontology $\mathcal{O}_{EX}$

3. Identification of $\forall$-constraint patterns

In the following part we identify a superset of concepts, which might be propagated over roles during the application of a tableau algorithm to an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$. Please note that we do not claim to find a minimal set of such constraints, but rather a set which allows for worst-case considerations. Although a minimal set would allow a more fine-grained analysis of the ontology $\mathcal{O}$, we think that computing such a minimal set is equivalent to precise reasoning on $\mathcal{O}$, something we assumed as not feasible before. Furthermore, the computation of a minimal set would require us to use information from the ABox in a preprocessing step - something we want to avoid for being able to deal with dynamic assertional information.

First of all, we know that $\forall$-constraints cannot come directly from the ABox, since we only allow for concept assertions of the kind $a : C$, where C is an atomic concept or its negation. Thus, $\forall$-constraints can only be derived from the TBox. Unfortunately, the shape of TBox axioms, i.e. $C \sqsubseteq D$, does not allow to easily read off "possible" $\forall$-constraints. This is due to the implicit presence of negation in the concept C. We propose some kind of normal form, which allows for extraction of a superset of $\forall$-constraints, which can possibly be used in a tableau algorithm.

Definition 1. *A concept description C is in* Shallow Normal Form *(SNF), if it has the shape $C = C_1 \sqcup C_2 \sqcup ... \sqcup C_n$, s.t. each C_i is either*

- *an atomic concept,*
- *a negated atomic concept,*
- *an $\exists$-constraint $\exists R.D$, s.t. D is an arbitrary concept description in negation normal form*
- *a $\forall$-constraint $\forall R.D$, s.t. D is an arbitrary concept description in negation normal form*

Please note that we do not enforce anything on concepts "hidden" behind $\forall/\exists$-constraints, but that they are in negation normal form. Thus the name *shallow* normal form.

Lemma 1. *Each GCI $C \sqsubseteq D$ can be converted into a set S of equivalent concept descriptions in SNF. Here, equivalent means that $C \sqsubseteq D$ is unsatisfiable iff the conjunction of the formulas in S is unsatisfiable.*

Proof. Sketch: We know that $(C \sqsubseteq D)$ is unsatisfiable if and only if $nnf(\neg C \sqcup D)$ is unsatisfiable. Starting with $nnf(\neg C \sqcup D)$, we can obtain a set of concept descriptions by applying equivalence preserving rules to bring conjunctions to the "outside" and break formulas up into SNF. $\square$

With $Shallow(\mathcal{T})$ we denote the set of concept descriptions, which are equivalent to the TBox inclusions in $\mathcal{T}$. Although the transformation into any conjunctive normal form can yield an exponential blow-up in the worst case, we claim that it is feasible to convert a TBox for our purposes. For details please refer to Section 5. Using Lemma 1, in the following we will assume w.l.o.g that a TBox is given as a set of concept descriptions in SNF. Next, we give a set of concept descriptions in SNF for $\mathcal{T}_{EX}$ from Example 1.

Example 2. *The TBox $\mathcal{T}_{EX}$ in SNF is as follows:*

$$
\begin{aligned}
Shallow(\mathcal{T}_{EX}) = \{ & \\
& \neg Chair \sqcup \exists headOf.Department, \neg Chair \sqcup Person, \\
& \forall headOf.\neg Department \sqcup \neg Person \sqcup Chair, \neg Professor \sqcup Faculty, \\
& \neg Book \sqcup Publication, \neg GraduateStudent \sqcup Student, \neg Student \sqcup Person, \\
& \neg Student \sqcup \exists takesCourse.Course, \neg Person \sqcup \forall takesCourse.\neg Course \sqcup Student, \\
& \forall teacherOf.Course, \forall teacherOf.\bot \sqcup Faculty, \neg Faculty \sqcup Person, \\
& \forall publicationAuthor^{-}.(Book \sqcup ConferencePaper) \\
& \}
\end{aligned}
$$

In the following we define a structure for managing $\forall$-constrains in an $\mathcal{ALCHI}$-ontology.

Definition 2. *A* $\forall$-info structure *for TBox $\mathcal{T}$ is a function $f_{\mathcal{T}}^{\forall} : N_R \rightarrow \mathcal{P}(N_C \cup \{\neg A | A \in N_C\} \cup \{\bot\}) \cup \{*\}$, s.t. N_C (N_R) is a set of atomic concepts (roles) used in $\mathcal{T}$. The function $f_{\mathcal{T}}^{\forall}$ is used to manage the $\forall$-constraints, i.e. the function assigns to each role name in N_R one of the following entries:*

```
Function $build^{\forall}(C, f^{\forall}_{\mathcal{T}})$
Parameter: Concept description $C$ in SNF, $\forall$-info structure $f^{\forall}_{\mathcal{T}}$
        1. If $C = C_1 \sqcap ... \sqcap C_n$ or $C = C_1 \sqcup ... \sqcup C_n$ then
            (a) For $1 < i < n$ do $build^{\forall}(C_i, f^{\forall}_{\mathcal{T}})$
        2. Else If $C = \exists R.C_1$ then
            (a) $build^{\forall}(C_1, f^{\forall}_{\mathcal{T}})$
        3. Else If $C = \forall R.C_1$ then
            (a) If $C_1$ is an atomic concept or a negated atomic concept or $\bot$ then
                i. If $f^{\forall}_{\mathcal{T}}(R) \neq *$ then $f^{\forall}_{\mathcal{T}}(R) = f^{\forall}_{\mathcal{T}}(R) \cup \{C_1\}$
            (b) else
                i. $f^{\forall}_{\mathcal{T}}(R) = *$
                ii. $build(C_1, f^{\forall}_{\mathcal{T}})$
        4. Return

Function $build^{\forall}(\mathcal{T})$
Parameter: TBox $\mathcal{T}$
        For each $R \in N_R$ do initialize
            $f^{\forall}_{\mathcal{T}}(R) = \emptyset$
        For each $C \in \mathcal{T}$ (in SNF) do
            $build^{\forall}(C, f^{\forall}_{\mathcal{T}})$
        Return $f^{\forall}_{\mathcal{T}}(R)$
```

Figure 3. Building $f^{\forall}_{\mathcal{T}}$

- $\emptyset$, *if we know that there is no* $\forall$*-constraint for* R *in* $\mathcal{T}$
- *a subset* S *of* $N_C \cup \{\neg A | A \in N_C\} \cup \{\bot\}$, *s.t. there is no other concept but those in* S*, which occur* $\forall$*-bound (i.e. they are a subconcept of a* $\forall$*-constraint) on* R *in* $\mathcal{T}$
- $*$, *if there are arbitrary complex* $\forall$*-constraints on role* R *in* $\mathcal{T}$*, but we don't give additional information on the structure of these constraints.*

The intuition for $f^{\forall}_{\mathcal{T}}$ is defined below. In Figure 3 we propose an algorithm to compute $f^{\forall}_{\mathcal{T}}$. We do not explain the algorithm, since it is a straightforward computation of the closure of $\mathcal{T}$ in SNF. Next, we lift the notion of a $\forall$-*info structure* from a TBox to an ontology.

Definition 3. *A* $\forall$-info structure *for ontology* $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ *is a function* $f^{\forall}_{\mathcal{O}} : N_R \rightarrow \mathcal{P}(N_C \cup \{\neg A | A \in N_C\} \cup \{\bot\}) \cup \{*\}$, *s.t.*

$$f^{\forall}_{\mathcal{O}}(R) = \begin{cases} * & if \exists S \in N_R. \sqsubseteq_R (R, S) \wedge (f^{\forall}_{\mathcal{T}}(S) = *) \\ \bigcup_{R \sqsubseteq_{\mathcal{R}} S} f^{\forall}_{\mathcal{T}}(S) & else \end{cases}$$

Let us look at our ontology $\mathcal{O}_{EX}$ again to give an example for a $\forall$-info structure.

Example 3. *The* ∀-info structure *for ontology $\mathcal{O}_{EX}$ is as follows:*

$$f^{\forall}_{\mathcal{O}_{EX}}(R) = \begin{cases} \{\neg Department\} & \textit{if } R = headOf \\ \{\neg Course\} & \textit{if } R = takesCourse \\ \{\bot, Course\} & \textit{if } R = teacherOf \\ * & \textit{if } R = publicationAuthor^{-} \\ \{\} & \textit{else} \end{cases}$$

The main result of this section is presented in the following lemma:

Lemma 2. *Given an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, the following holds: if $f^{\forall}_{\mathcal{O}}(R) \neq *$, then for each valid tableau proof P of $\mathcal{O}$ and for each application of a ∀-rule (on R and subconcept C) in P, we have that $C \in f^{\forall}_{\mathcal{O}}(R)$.*

Proof. Sketch: By contradiction. W.l.o.g. assume that there exists a tableau proof P, s.t. during the proof the ∀-rule is applied to individual a, which is labeled with $\forall R.C$ and $C \notin f^{\forall}_{\mathcal{O}}(R)$. Since we only allow ABox assertions for atomic concepts, the ∀-constraint must come from the TBox. Thus $\forall R.C$ must be a subconcept of $\mathcal{T}$ in SNF. Since $f^{\forall}_{\mathcal{O}}$ is build by computing the closure of the $\mathcal{T}$, we have either $f^{\forall}_{\mathcal{O}}(R) = *$ or $C \in f^{\forall}_{\mathcal{O}}(R)$. Contradiction. □

4. ABox Rewriting and Island Computation

Given the notions in the previous section, in the following we will derive means to rewrite an ontology $\mathcal{O}$, s.t. inconsistency is preserved, i.e, $INC(\mathcal{O})$ if and only if $INC(\mathcal{O}_{rewritten})$. Inconsistency tests are important for instance checking, since it holds that $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \cup \{a : \neg C\}\rangle)$ if and only if $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models a : C$. Please note that we are not going to rewrite the ABoxes in practice, but rather use rewriting for proving soundness and completeness of our algorithm for island computation. The following definition of $\mathcal{O}$-separability is used to determine the importance of role assertions in a given ABox. Informally speaking, the idea is that $\mathcal{O}$-separable assertions will never be used to propagate "complex and new information" (see below) via role assertions.

Definition 4. *Given an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, a role assertion $R(a, b)$ is called $\mathcal{O}$-*separable*, if we have $INC(\mathcal{O}) \iff INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}_2\}\rangle)$, where*

$$\mathcal{A}_2 = \mathcal{A} \setminus \{R(a, b)\} \cup \{R(a, i_1), R(i_2, b)\} \cup \{i_1 : C | b : C \in \mathcal{A}\} \cup \{i_2 : C | a : C \in \mathcal{A}\},$$

s.t. i_1 and i_2 are fresh individual names.

Given the above definitions, we propose a formal criterion on role assertions w.r.t. an ontology $\mathcal{O}$ to distinguish, whether they are $\mathcal{O}$-separable.

Lemma 3. *Given an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ and a role assertion $R(a, b) \in \mathcal{A}$, it holds that $R(a, b)$ is $\mathcal{O}$-separable, if we have*

1. *For each $C \in f^{\forall}_{\mathcal{O}}(R)$*

   (a) *$C = \bot$ or*
   (b) *we can find a concept description $D \in \{E|b : E \in \mathcal{A}\}$, s.t. we have $D \sqsubseteq_{\mathcal{T}} C$*

2. *For each $C \in f^{\forall}_{\mathcal{O}}(R^-)$*
   (a) *$C = \bot$ or*
   (b) *we can find a concept description $D \in \{E|a : E \in \mathcal{A}\}$, s.t. we have $D \sqsubseteq_{\mathcal{T}} C$*

Proof. We have to show that $INC(\mathcal{O}) \iff INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}_2\}\rangle)$, where

$$\mathcal{A}_2 = \mathcal{A} \setminus \{R(a,b)\} \cup \{R(a,i_1), R(i_2,b)\} \cup \{i_1 : C|b : C \in \mathcal{A}\} \cup \{i_2 : C|a : C \in \mathcal{A}\},$$

s.t. i_1 and i_2 are fresh individual names. The proof can be done in two steps (directions):

"$\Leftarrow$" To show: $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}_2\}\rangle) \Rightarrow INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}\}\rangle)$.
From $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}_2\rangle)$ we know that there exists a tableau proof P, which returns a closed tableau. We can apply the same tableau proof to $\langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$, by applying each tableau-rule involving individual i_1 to b and applying each tableau-rule involving individual i_2 to a. Again, we obtain a closed tableau and it holds that $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}\}\rangle)$.

"$\Rightarrow$" To show: $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle) \Rightarrow INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}_2\rangle)$.
Sketch: The idea is that each closed tableau proof P on $\langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$ can be rewritten to a closed tableau proof P_2 on $\langle \mathcal{T}, \mathcal{R}, \mathcal{A}_2\rangle$. This is due to the fact that only implicitly known information and immediate clashs are propagated via the split role assertion. This can be shown by induction on $ALCHI$-tableau rules.

□

Let us consider an example for $\mathcal{O}$-separability w.r.t. $\mathcal{O}_{EX}$ from Example 1:

Example 4. *For instance the ABox assertion $teacherOf(p2, c3)$ in Example 1 is $\mathcal{O}_{EX}$-separable, since we have*

- $f^{\forall}_{\mathcal{O}_{EX}}(teacherOf) = \{\bot, Course\}$ *and* $c3 : Course \in \mathcal{A}_{EX}$
- $f^{\forall}_{\mathcal{O}_{EX}}(teacherOf^-) = \{\}$

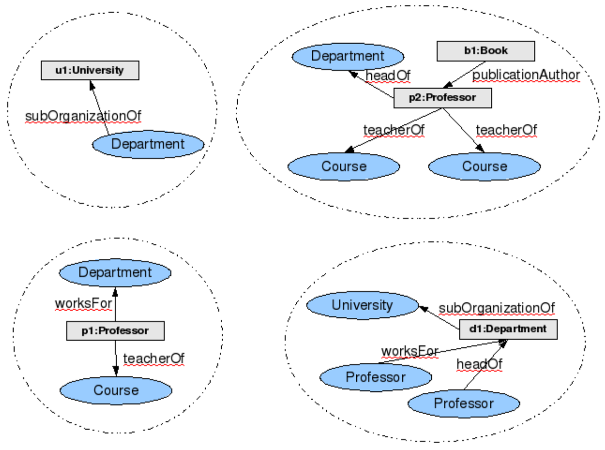
Definition 5. *Given an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$, let $RED(\mathcal{A})$ be the ABox computed from $\mathcal{A}$ by replacing each $\mathcal{O}$-separable role assertion $R(a,b)$ by $\{R(a,i_1), R(i_2,b)\} \cup \{i_1 : C|b : C \in \mathcal{A}\} \cup \{i_2 : C|a : C \in \mathcal{A}\}$, s.t. i_1 and i_2 are fresh individual names.*

Lemma 4. *It holds that $INC(\langle \mathcal{T}, \mathcal{R}, \mathcal{A}\rangle)$ iff $INC(\langle \mathcal{T}, \mathcal{R}, RED(\mathcal{A})\rangle)$.*

Proof. Easy: By definition of $\mathcal{O}$-separability. □

Definition 6. *An* interconnection-based partitioning *for a ABox $\mathcal{A}$, denoted $P(\mathcal{A}) = \{\mathcal{A}_1, ...\mathcal{A}_n\}$, is built by role-connectedness, i.e. two individuals are in the same partition iff there exists an explicit role assertion between these two individuals.*

Example 5. *Some parts of the interconnection-based partitioning $P(RED(\mathcal{A}_{EX}))$ for $RED(\mathcal{A}_{EX})$ are shown below (4 partitions). Rectangular nodes denote individuals from $\mathcal{A}_{EX}$ and elliptic nodes denote fresh (unnamed) individuals.*



Lemma 5. *We have $INC(\langle\mathcal{T},\mathcal{R},\mathcal{A}\rangle)$ iff $\exists\mathcal{A}_i \in P(RED(\mathcal{A})).INC(\langle\mathcal{T},\mathcal{R},\mathcal{A}_i\rangle)$.*

Proof. Easy: If all partitions are disconnected, then there is no interaction between the partitions. □

Definition 7. *Given an ABox $\mathcal{A}$, let $P_a(\mathcal{A})$ be the partition in $P(RED(\mathcal{A}))$, which contains individual a.*

Lemma 6. *Given $CON(\langle\mathcal{T},\mathcal{R},\mathcal{A}\rangle)$, we have that $INC(\langle\mathcal{T},\mathcal{R},\mathcal{A}\cup\{a : C\}\rangle)$ iff $INC(\langle\mathcal{T},\mathcal{R},P_a(\mathcal{A})\cup\{a : C\}\rangle)$*

Proof. $\Leftarrow$:

Easy, since $P_a(\mathcal{A})\cup\{a : C\} \subseteq \mathcal{A}\cup\{a : C\}$ (modulo renaming).

$\Rightarrow$:

Let $P_1 = P(RED(\mathcal{A}))$ and $P_2 = P(RED(\mathcal{A}\cup\{a : C\}))$. By definition of RED and P it is clear, that $P_2 \setminus P_1 = \{P_a(RED(\mathcal{A}\cup\{a : C\}))\}$. Since all partitions in P_1 are consistent (by $CON(\langle\mathcal{T},\mathcal{R},\mathcal{A}\rangle)$), $P_a(RED(\mathcal{A}\cup\{a : C\}))$ has to be inconsistent (by Lemma 5).

□

Next, we propose an algorithm, which solves the following problem: Given an ontology $\mathcal{O} = \langle\mathcal{T},\mathcal{R},\mathcal{A}\rangle$, check, whether $\mathcal{O} \vDash a : C$ holds for a given individual a and a given concept C, without having to take the whole ontology into consideration. The idea is that we identify a subset S of $\mathcal{A}$, s.t. we have $\mathcal{O} \vDash a : C$ iff $\langle\mathcal{T},\mathcal{R},S\rangle \vDash a : C$. The set S is usually orders of magnitudes smaller than the initial ABox $\mathcal{A}$ (for detailed statistics see Section 5). Given Lemma 6, we already have a notion at hand, which identifies a relevant subset of ABox assertions necessary for reasoning about a given individual a. It is easy to define a function, which computes $P_a(\mathcal{A})$, given an individual a (see 4). Soundness and completeness of the algorithm is clear from the definition of $\mathcal{O}$-separability. Termination is ensured by use of a “seen individuals”-list, which avoids following cycles during application of the algorithm.

```
Function $build(a, seen)$
Parameter: Individual $a$, list of visited individuals $seen$
Returns: Set $S$ of relevant ABox assertions
Algorithm:
    1. If $a \in seen$ then Return $\emptyset$
    2. $seen = seen \cup \{s\}$
    3. $S = \{a : X \in \mathcal{A}\}$
    4. For $R(a, b) \in \mathcal{A}$ do
        (a) If $f^{\forall}_{\mathcal{O}}(R) = *$ or $f^{\forall}_{\mathcal{O}}(R^{-}) = *$ then
            $S = S \cup \{R(a, b)\} \cup build(b, seen)$
        (b) else if $f^{\forall}_{\mathcal{O}}(R) \neq \emptyset$ or $f^{\forall}_{\mathcal{O}}(R^{-}) \neq \emptyset$ then
            i. $found = true$
            ii. For $C \in f^{\forall}_{\mathcal{O}}(R)$ do
                If not($C = \bot$ or $(\exists D.b : D \in \mathcal{A} \wedge D \sqsubseteq_{\mathcal{T}} C)$ or $(b : nnf(\neg C) \in \mathcal{A})$) then
                  found=false
            iii. For $C \in f^{\forall}_{\mathcal{O}}(R^{-})$ do
                If not($C = \bot$ or $(\exists D.a : D \in \mathcal{A} \wedge D \sqsubseteq_{\mathcal{T}} C)$ or $(a : nnf(\neg C) \in \mathcal{A})$) then
                  found=false
        (c) If $found = true$ then
            $S = S \cup \{R(a, i_x)\}$, s.t. $i_x$ does neither occur in $Ind(\mathcal{A})$ nor in $S$
        (d) else
            $S = S \cup \{R(a, b)\} \cup build(b, seen)$
    5. For $R(b, a) \in \mathcal{A}$ do
        (a) If $f^{\forall}_{\mathcal{O}}(R) = *$ or $f^{\forall}_{\mathcal{O}}(R^{-}) = *$ then
            $S = S \cup \{R(b, a)\} \cup build(b, seen)$
        (b) else if $f^{\forall}_{\mathcal{O}}(R) \neq \emptyset$ or $f^{\forall}_{\mathcal{O}}(R^{-}) \neq \emptyset$ then
            i. $found = true$
            ii. For $C \in f^{\forall}_{\mathcal{O}}(R)$ do
                If not($C = \bot$ or $(\exists D.a : D \in \mathcal{A} \wedge D \sqsubseteq_{\mathcal{T}} C)$ or $(a : nnf(\neg C) \in \mathcal{A})$) then
                  found=false
            iii. For $C \in f^{\forall}_{\mathcal{O}}(R^{-})$ do
                If not($C = \bot$ or $(\exists D.b : D \in \mathcal{A} \wedge D \sqsubseteq_{\mathcal{T}} C)$ or $(b : nnf(\neg C) \in \mathcal{A})$) then
                  found=false
        (c) If $found = true$ then
            $S = S \cup \{R(i_x, a)\}$, s.t. $i_x$ does neither occur in $Ind(\mathcal{A})$ nor in $S$
        (d) else
            $S = S \cup \{R(b, a)\} \cup build(b, seen)$
    6. Return $S$
```

Figure 4. Build island for individual a

5. Evaluation and Suggestions for Further Improvements

In the following section we evaluate our proposal for island reasoning. We have implemented the proposed algorithms in Java. For ontology access we used a Java interface and implementation for the Ontology Web Language, called OWLAPI [BVL03]. Given the OWLAPI interface, implementing the above algorithms was straightforward.

Ontology	|Inclusions|	|Equivalences|	$\|N_{RN}\|$	Time for analysis (ms)
LUBM [GPH05]	75	6	25	4
Cyc [CYC05]	43541	2	4853	93
GODaily [HCI04]	28997	0	1	103
Galen [ALRP96]	3388	699	413	55
Pizza [oM08]	57	2	7	3

Figure 5. Extraction of $\forall$-info structures for several ontologies

First, we investigate, whether it is feasible to convert the TBox of a given ontology into Shallow Normal Form and extract the $\forall$-info structure $f_{\mathcal{O}}^{\forall}$. For this purpose we have selected five arbitrary ontologies. For lack of space please refer to the given references for a detailed description of the ontologies. The results of our investigation are shown in Figure 5. We think the numbers indicate that extracting a $\forall$-info structure for most ontologies should be feasible. Even ontologies whose TBox is considered large for current description logic systems do perform quite well.

Second, we look at the average size of extracted islands. Our tests w.r.t. LUBM are quite encouraging. The average size of an island is 29 nodes. The actual island size depends on the chosen individual. Some preliminary statistics on example islands are given in Figure 6. Please note that the island size does not depend on the number of universities, i.e. for LUBM our approach promises quite good scalability.

The big island for $Dep0.Uni0/FullProfessor7$ can be explained as follows. LUBM imposes several $\forall$-constraints on the role $headOf$. Due to the definitions of $Chair$, $Dean$ and $Director$, the atomic concept descriptions $\neg Department$, $\neg College$ and $\neg Program$ can be propagated via role $headOf$. Since all individuals with an incoming $headOf$-edge are of type $Department$, we have to take them completely into account for building the islands. If there were further constraints in the TBox, e.g. disjointness of $Department$, $College$ and $Program$, we could further reduce the size of islands for individuals of type $FullProfessor$.

Individual	**Island size**	**Time for island computation**
$Dep0.Uni0/GraduateStudent128$	9	0 ms
$Dep0.Uni0/Publication2$	4	1 ms
$Dep0.Uni0/FullProfessor7$	93	2 ms
$Dep0.Uni0/Course4$	37	0 ms

Figure 6. Statistics for islands in LUBM

In the following we discuss several ideas for further improvement of island reasoning. With respect to other inference tasks, e.g. instance retrieval, a naive application of island loading can be doomed to failure. For instance, if there are several million named individuals in the ABox of $\mathcal{O}$, then one needs to load one island for each named individual and check, whether this individual is an instance of the concept in question. Thus, while the atomic operation (instance checking) can be performed quite efficiently, the overall run time can be still slow. We propose the following improvements to overcome such problems:

"Preselection" by incomplete/unsound reasoning approaches The idea here is to restrict the set of possible answers to a query by a fast algorithm, which neglects either soundness or completeness. Let $\phi_{sound} : N_C \rightarrow \mathcal{P}(N_A)$ be a sound instance retrieval function, i.e. we have $a \in \phi_{sound}(A) \implies \mathcal{O} \models a : A$. Furthermore, let $\phi_{complete} : N_C \rightarrow \mathcal{P}(N_A)$ be a complete instance retrieval function, i.e. we have $\mathcal{O} \models a : A \implies a \in \phi_{complete}(A)$.
If we want to perform instance retrieval for an atomic concept A, we only need to check for all individuals $a \in \phi_{complete}(A) \setminus \phi_{sound}(A)$, whether we have $\mathcal{O} \models a : A$. For all remaining individuals we do not have to perform explicit instance checking. Please note that the quality of such an improvement depends on the quality of ϕ_{sound} and $\phi_{complete}$. Possible suggestions for such functions are:

$\phi_{complete}$: Initial summary ABox [FKM$^+$06], role condensates [WM07]
ϕ_{sound}: Any precompletion technique [BCM$^+$07]

To sum up, we think that the combination of different scalability-notions will be of importance to cope with the increasing amount of assertional data.

Precomputation of islands in case of static ABoxes Throughout the paper we emphasize the advantages of our proposal for dynamic assertional information. However, if one knows in advance that the ABox will not change at all, our approach can be improved further. For instance, one could precompute the islands for each named individual in advance (offline). Then, whenever the island for individual a is needed, one only needs to load the precomputed island. The approach is even more promising, when you distribute these precomputed island among several computers.

Grouping individuals for "similarity"-islands Let S be a set of individuals for which we want to perform instance checking. In a naive way, we would have to iterate over all individuals in S. However, one could load the *group of islands* for S, by considering $\bigcup_{s \in S} build(s, \emptyset)$. It is easy to see that this group of islands allows for sound and complete reasoning for all individuals in S. Please note that the actual improvement depends on the choice of individuals in S. Usually one would like to group closely-related individuals together. A detailed investigation of this is part of future work.

6. Conclusions and Future Work

We have introduced means to reason over ontologies, which have large amounts of assertional information. Given an individual of interest, island reasoning allows state-of-the-art description logic reasoners to load only relevant subsets of the ABox to perform sound and complete reasoning. In particular, we have proposed a preprocessing step which can be easily performed offline (computation of $\forall$-info structures) and an actual island computation algorithm, which can be run on demand.

To analyze the scalability of our approach, we have investigated several commonly used ontologies. We think that our evaluation shows a clear advantage, since the identified island for a given individual is usually quite small. Furthermore we have proposed additional improvements which are subject to future work. Additionally, we will investigate the applicability of our proposal to more expressive description logics, e.g. SHIQ. The extension for transitive roles is straightforward. The incorporation of min/max-cardinality constraints in a naive way can be done as well. However, it has to be investigated, whether the average island size is still small enough to be feasible in practice.

References

[ACG$^+$05] Andrea Acciarri, Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Mattia Palmieri, and Riccardo Rosati. Quonto: Querying ontologies. In Manuela M. Veloso and Subbarao Kambhampati, editors, *AAAI*, pages 1670–1671. AAAI Press / The MIT Press, 2005.

[ALRP96] J. E. Rogers A. L. Rector and P. A. Pole. The GALEN High Level Ontology. In *Proceedings MIE 96*, pages 174–178. IOS Press, jan 1996.

[BCM+07] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider. *The Description Logic Handbook*. Cambridge University Press, New York, NY, USA, 2007.

[BVL03] S. Bechhofer, R. Volz, and P. Lord. Cooking the Semantic Web with the OWL API, 2003.

[CYC05] CYC. Cyc ontology (researchcyc). visited on, February 2005. retrieved from http://www.cyc.com/.

[FKM+06] Achille Fokoue, Aaron Kershenbaum, Li Ma, Chintan Patel, Edith Schonberg, and Kavitha Srinivas. Using Abstract Evaluation in ABox Reasoning. In *SSWS 2006*, pages 61–74, Athens, GA, USA, November 2006.

[GH06] Yuanbo Guo and Jeff Heflin. A Scalable Approach for Partitioning OWL Knowledge Bases. In *SSWS 2006*, Athens, GA, USA, November 2006.

[GPH05] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. Lubm: A benchmark for owl knowledge base systems. *J. Web Sem.*, 3(2-3):158–182, 2005.

[HCI04] M. A. Harris, J. Clark, and A. et. al. Ireland. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res*, 32(Database issue), January 2004.

[HM01] V. Haarslev and R. Möller. Description of the RACER System and its Applications. In *Proceedings International Workshop on Description Logics (DL-2001), Stanford, USA, 1.-3. August*, pages 131–141, 2001.

[oM08] The University of Manchester. Pizza ontology. visited on, April 2008. retrieved from http://www.co-ode.org/ontologies/pizza/2007/02/12/.

[Wan08] S. Wandelt. Partitioning OWL Knowledge Bases - Revisited and Revised. In *Proc. International Workshop on Description Logics*, 2008.

[WM07] S. Wandelt and R. Möller. Scalability of OWL Reasoning: Role condensates. In *Proc. International Workshop on Scalable Semantic Web Systems*, 2007.