Scalable Semantic Retrieval Through Summarization and Refinement

Julian Dolby, Achille Fokoue, Aditya Kalyanpur,
Aaron Kershenbaum, **Edith Schonberg**, **Kavitha Srinivas**
IBM Watson Research Center
P.O.Box 704, Yorktown Heights, NY 10598, USA
dolby, achille,adityakal, aaronk, ediths, ksrinivs@us.ibm.com

Li Ma
IBM China Research Lab
Beijing 100094, China
malli@ cn.ibm.com

Abstract

Query processing of OWL-DL ontologies is intractable in the worst case, but we present a novel technique that in practice allows for efficient querying of ontologies with large Aboxes in secondary storage. We focus on the processing of *instance retrieval* queries, i.e., queries that retrieve individuals in the Abox which are instances of a given concept C. Our technique uses summarization and refinement to reduce instance retrieval to a small relevant subset of the original Abox. We demonstrate the effectiveness of this technique in Aboxes with up to 7 million assertions. Our results are applicable to the very expressive description logic $\mathcal{SHIN}$, which corresponds to OWL-DL minus nominals and datatypes.

keywords: Reasoning, Description Logic, Ontology.

Introduction

Semantic retrieval is one of the important applications of ontologies and reasoning. Using reasoning to answer a query, information which is not explicitly stored in a knowledge base can be inferred, thus improving recall. Reasoning algorithms that can be scaled to realistic databases are a key enabling technology for semantic retrieval.

We focus in this paper on the scalable processing of *instance retrieval* queries for OWL-DL knowledge bases in secondary storage (excluding nominals and datatypes). An OWL-DL knowledge base consists conceptually of three components: the Tbox which contains terminological assertions about concepts, the Rbox which contains assertions about roles and role hierarchies, and the Abox which contains membership assertions and role assertions between individuals. An instance query retrieves the individuals in the Abox which are instances of a given concept C, w.r.t. information in the Tbox and Rbox.

It is well known that all queries over expressive-DL ontologies can be reduced to consistency detection (Horrocks & Tessaris 2002), which is usually checked with a tableau algorithm. As an example, a simple algorithm for instance retrieval can be realized by testing if the addition of an assertion $a : \neg C$ for a given individual a results in an inconsistency. If the resulting Abox is inconsistent, then a is an instance of C. Of course, it is not practical to apply this simple approach to every individual.

We propose a novel approach that uses summarization and refinement to scale instance retrieval to large expressive Aboxes in databases. Before processing any queries, we create a *summary Abox* $\mathcal{A}'$ (A.Fokoue *et al.* 2006) from an original Abox $\mathcal{A}$ and store it in a database. A summary Abox is created by aggregating individuals which are members of the same concepts. Query processing is performed on $\mathcal{A}'$ rather than $\mathcal{A}$. By testing an individual in the summary Abox, all individuals mapped to the summary are effectively tested at the same time.

However, there is a catch. For a tested individual s in $\mathcal{A}'$, if the summary is found to be consistent, then we know that all individuals mapped to that summary individual s are not solutions. But if the summary is found to be inconsistent, it is possible that either (a) a subset of individuals mapped to the summarized individual s are instances of the query or (b) the inconsistency is a spurious effect of the summarization. We determine the answer through *refinement*, which selectively expands the summary Abox to make it more precise.

Refinement is an iterative process that partitions the set of individuals mapped to a single summary individual and remaps each partition to a new summary individual. The iteration ends when either the expanded summary is consistent, or it can be shown that all individuals mapped to the tested summary individual are solutions. Significantly, convergence on the solution is based only on the structure of the refined summary, without testing individuals in $\mathcal{A}$.

With this summarize-and-refine technique, it is critical to have an effective refinement strategy, which limits both the number of refined individuals and the number of iterations. Our refinement strategy is based on identifying *justifications* for the inconsistency in the summary Abox, which is a minimal inconsistent subset of the summary (Kalyanpur 2006), and selectively applying refinement to individuals in justifications. We test multiple individuals in the summary at the same time, and process multiple justifications at each refinement step. This approach proved effective on the UOBM benchmark ontology (Ma *et al.* 2006), where we demonstrate that we process Abox queries with up to 7.4 million assertions efficiently, whereas the state of the art reasoners could not scale to this size.

In addition to guiding refinement, justifications are helpful for users to understand query results. Since our explanations are at a summarized level, the information is more

useful than detailed information about each individual in an Abox. Another key point is that our summarize-and-refine technique can be treated as an optimization that any tableau reasoner can employ to achieve scalable ABox reasoning.

The key contributions of this paper are as follows: (a) We present a novel, tableau-based technique to use summarization and refinement for efficient instance retrieval for large $\mathcal{SHIN}$ Aboxes in secondary storage. (b) We describe the use of optimization techniques to selectively target parts of the summary for refinement. (c) We show the application of these techniques to the UOBM benchmark ontology with $\mathcal{SHIN}$ expressivity, where we show dramatic reductions in space and time requirements for instance retrieval.

Background

A key feature of our approach is the construction of a summary Abox $\mathcal{A}'$ corresponding to the Abox $\mathcal{A}$. An individual in $\mathcal{A}'$ represents individuals in $\mathcal{A}$ which are members of the same concepts. Formally, an Abox $\mathcal{A}'$ is a summary Abox of a $\mathcal{SHIN}$ [1] Abox $\mathcal{A}$ if there is a mapping function $\mathbf{f}$ that satisfies the following constraints:

(1) if $a : C \in \mathcal{A}$ then $\mathbf{f}(a) : C \in \mathcal{A}'$

(2) if $R(a, b) \in \mathcal{A}$ then $R(\mathbf{f}(a), \mathbf{f}(b)) \in \mathcal{A}'$

(3) if $a \dot{\neq} b \in \mathcal{A}$ then $\mathbf{f}(a) \dot{\neq} \mathbf{f}(b) \in \mathcal{A}'$

The *image* of an individual a in $\mathcal{A}'$ is the set Im of individuals in $\mathcal{A}$ such that $\mathbf{f}(b) = a, b \in Im$. If the summary Abox $\mathcal{A}'$ obtained by applying the mapping function $\mathbf{f}$ to $\mathcal{A}$ is consistent w.r.t. a Tbox $\mathcal{T}$ and a Rbox $\mathcal{R}$, then $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$ and $\mathcal{R}$ (A.Fokoue *et al.* 2006). However, the converse does not hold.

Let $\mathcal{L}$ be a mapping from each individual in $\mathcal{A}$ to a set of concepts, such that $a : C \in \mathcal{A}$ iff $C \in \mathcal{L}(a)$. We call $\mathcal{L}(a)$ the *concept set* of a. In practice, we use a *canonical function* $\mathbf{f}$ to create a summary Abox, which maps all non-distinct individuals that have identical concept sets to the same individual in $\mathcal{A}'$. More precisely, the converse of constraints (1) and (3) hold for the *canonical summary*, and:

(4) If $R(a', b') \in \mathcal{A}'$ then there exists a pair of individuals (a, b) in $\mathcal{A}$ such that $a' = \mathbf{f}(a), b' = \mathbf{f}(b)$ and $R(a, b) \in \mathcal{A}$.

(5) If for all $x \in \mathcal{A}$, $(a \dot{\neq} x) \notin \mathcal{A}$, $b \dot{\neq} x \notin \mathcal{A}$, and $\mathcal{L}(a) = \mathcal{L}(b)$, then $\mathbf{f}(a) = \mathbf{f}(b)$.

(6) $\mathbf{f}(a) \dot{\neq} \mathbf{f}(b) \in \mathcal{A}'$ implies a is the only individual in $\mathcal{A}$ mapped to $\mathbf{f}(a)$ (same for b).

In general, the canonical summary Abox $\mathcal{A}'$ is dramatically smaller than the original Abox $\mathcal{A}$. It can be constructed efficiently from $\mathcal{A}$ using conventional relational database queries. It only needs to be computed once, persisted, and then reused in subsequent queries. It is easily updated incrementally and is thus resilient to changes in $\mathcal{A}$.

Overview

We motivate our instance retrieval algorithm using an example based on the UOBM ontology with an Abox $\mathcal{A}$ shown in Figure 1. The individuals $c1$, $c2$ and $c3$ are instances of the

concept $Course$; $p1$, $p2$ and $p3$ are instances of $Person$; $m1$ and $m2$ are instances of Man; $w1$ is an instance of $Woman$, and $h1$ and $h2$ are instances of $Hobby$. For now, we assume that the Tbox contains only one axiom stating that Man and $Woman$ are disjoint concepts and the Rbox is empty.
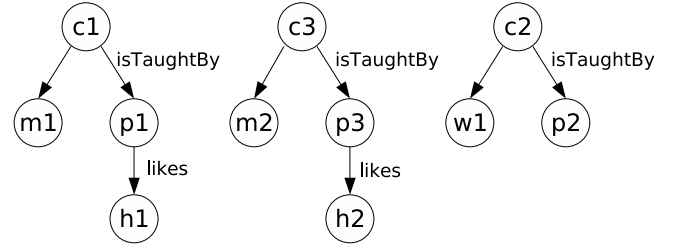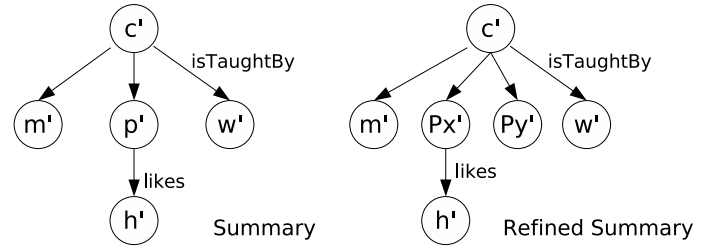

Figure 1: Example Abox

Figure 2: Summary Abox

Consider the query for the concept $PeopleWithHobby$, which is defined as $Person \sqcap \geq 1likes$. From Figure 1, it is clear that the answer consists of the individuals $p1$ and $p3$, which are the only $Person$ individuals in $\mathcal{A}$ which are related to at least one $Hobby$ individual by the $likes$ role.

Instead of reasoning on $\mathcal{A}$ directly, our algorithm reasons on the canonical summary Abox $\mathcal{A}'$ for $\mathcal{A}$. The canonical summary is shown in the left of Figure 2, where the single individuals $c1$, $c2$, $c3$ are mapped to c' in $\mathcal{A}'$, $m1$ and $m2$ are mapped to m' in $\mathcal{A}'$, and so on. Since the Rbox is empty (i.e., $isTaughtBy$ is not functional), $\mathcal{A}'$ is consistent. By testing a summary individual, our goal is to draw conclusions about all of the actual individuals in $\mathcal{A}$ mapped to it. An individual s in $\mathcal{A}'$ is tested by adding the assertion $s : \neg PeopleWithHobby$ to $\mathcal{A}'$, and checking for consistency using a tableau reasoner. To achieve scalability, we test multiple individuals in the summary graph at the same time.

Definition 1. *Let $\mathcal{A}$ be an Abox. Let Q be a concept expression. Let S be a subset of individuals in $\mathcal{A}$ such that for all $s \in S$, $s : \neg Q \notin \mathcal{A}$.* [2] *We define the tested Abox w.r.t. $\mathcal{A}$, Q and S, denoted* tested$(\mathcal{A}, Q, S)$*, to be the Abox obtained from $\mathcal{A}$ by adding the assertion $s : \neg Q$ for each $s \in S$. Formally,* tested$(\mathcal{A}, Q, S) = \mathcal{A} \cup \{s : \neg Q | s \in S\}$*.*

---

[1] We assume without loss of generality that $\mathcal{A}$ does not contain an assertion of the form $a \dot{=} b$

[2] Note that we exclude individuals s such that $s : \neg Q$ belongs to $\mathcal{A}'$ because they are obviously not solutions and it also allows us to easily track assertions that were actually added to the tested summary.

If the result of testing a single individual s is consistent, then we know that none of the individuals in the image of s is a query solution. However, if the result is inconsistent, then we cannot conclude anything about individuals in the image of s. This situation arises because individuals are aggregated based only on the similarity of their concepts, not relationships.

In the example, adding $c' : \neg PeopleWithHobby$ is consistent. Therefore, $c1$, $c2$, and $c3$ in $\mathcal{A}$ can be excluded. Similarly, $m1$, $m2$, $w1$, $h1$ and $h2$ can be immediately excluded. When testing p', the result of adding $p' : \neg PeopleWithHobby$ to the summary is inconsistent. In this case, not all individuals in the image of p' satisfy the query, since they differ in their relationships.

Our approach for resolving summary Abox inconsistencies is to iteratively *refine* the summary. Refinement increases the size and precision of the summary, and preserves the summary Abox properties(1)-(3) defined in the previous section. Our strategy is to refine only individuals that are part of a summary Abox *justification*, where a justification is a minimal set of assertions which, when taken together, imply a logical contradiction, thus making the entire Abox inconsistent. In some cases, inconsistencies disappear through refinement. Otherwise, when a justification $\mathcal{J}$ is *precise* (as defined below in Definition 2) we typically know that we have converged on a solution. That is, there is a tested individual s in $\mathcal{J}$, such that all of the individuals in the image of s are instances of the query. We say that a tested individual s is *tested in $\mathcal{J}$* for query Q if $s : \neg Q$ is an assertion in $\mathcal{J}$. (The topic of drawing a conclusions on precise justifications is discussed in a later section.)

Definition 2. *Let $\mathcal{A}'$ be a summary Abox of an Abox $\mathcal{A}$ obtained through the summary mapping $\mathbf{f}$. Let Q be a queried concept, S be a subset of individuals in $\mathcal{A}'$ such that for all $x \in S$, $x : \neg Q \notin \mathcal{A}'$ and let H be a subset of $\text{tested}(\mathcal{A}', Q, S)$. We say that an individual $s \in H$ is precise w.r.t. H iff the following conditions are satisfied:*

1. *for all individuals $t \in H$ and for all roles R, $R(s,t) \in H$ (resp. $R(t,s) \in H$) implies that, for all individuals $a \in \mathcal{A}$ such that $\mathbf{f}(a) = s$, there is an individual $b \in \mathcal{A}$ such that $\mathbf{f}(b) = t$ and $R(a,b) \in \mathcal{A}$ (resp. $R(b,a) \in \mathcal{A}$); and*

2. *for all individuals $t \in H$, $s \dot{\neq} t \in H$ (resp. $t \dot{\neq} s \in H$) implies that, for all individuals $a \in \mathcal{A}$ such that $\mathbf{f}(a) = s$, there is an individual $b \in \mathcal{A}$ such that $\mathbf{f}(b) = t$ and $a \dot{\neq} b \in \mathcal{A}$ (resp. $b \dot{\neq} a \in \mathcal{A}$); and*

3. *There is an individual $a \in \mathcal{A}$ such that $\mathbf{f}(a) = s$; and*

4. *$s : C \in H - \{x : \neg Q | x \in S\}$ implies that, for all individuals $a \in \mathcal{A}$ such that $\mathbf{f}(a) = s$, $a : C \in \mathcal{A}$*

We say that H is precise iff all its individuals are precise w.r.t. H.

Continuing with the example, there is a justification $\mathcal{J}$ consisting of $p' : \neg PeopleWithHobby$, $h' : Hobby$, and $likes(p', h')$, and p' is tested in $\mathcal{J}$. Note that p' is not precise since it does not satisfy condition 1. Refinement replaces p' by two individuals p_x' and p_y', where the image of p_x' is $p1$ and $p3$, and the image of p_y' is $p2$. The result of this refinement is shown in the right of Figure 2. The tested refined summary is still inconsistent, and there is now a precise justification $p_x' : \neg PeopleWithHobby$, $h' : Hobby$, and $likes(p_x', h')$. Every $Person$ in the image of p_x' likes a $Hobby$. Thus, the image of p_x', namely $p1$ and $p3$, are solutions.

A high level outline of our algorithm is shown below. More details are given in subsequent sections.

$S \leftarrow \{x | x \in individuals\ in\ \mathcal{A}'\ and\ x : \neg Q \notin \mathcal{A}'\}$;
$R \leftarrow \mathcal{A}'$;
$Results \leftarrow \emptyset$;
while $S \neq \emptyset$ **do**
    $R_T \leftarrow tested(R, Q, S)$ (see Definition 1.) ;
    if *consistent(R_T)* **then**
        return $Results$;
    end
    Find $Justifications$ in R_T;
    $T \leftarrow$ individuals tested in precise $Justifications$;
    $Results \leftarrow Results \cup Image(T)$;
    $S \leftarrow S - T$;
    Execute refinement strategy on R ;
end
return $Results$;

Refinement

This section describes justification-based refinement, and some of the issues in deciding justification refinement order.

Definition 3. *Let $\mathcal{A}'$ and A_R' be summary Aboxes of an abox $\mathcal{A}$, with resp. mapping functions $\mathbf{f}$ and $\mathbf{f_R}$. Let I' be the set of individuals in $\mathcal{A}'$, and $\mathbf{I}_R'$ be the individuals in A_R'. Let s be an individual in $\mathcal{A}'$. We say that A_R' is a refinement of $\mathcal{A}'$ w.r.t. s iff there are individuals $s_1 \ldots s_n$, $n > 1$, in $\mathcal{A}'_R$ such that:*

1. *$I_R' = (I' - \{s\}) \cup \{s_1 \ldots s_n\}$ where $s_1 \ldots s_n \notin I'$*
2. *if $\mathbf{f}(a) \neq s$, then $\mathbf{f_R}(a) = \mathbf{f}(a)$*
3. *if $\mathbf{f}(a) = s$, then $\mathbf{f_R}(a) = s_i$, for some $1 \leq i \leq n$.*
4. *for each $1 \leq i \leq n$, there is at least one individual a in $\mathcal{A}$ such that $\mathbf{f_R}(a) = s_i$.*

In the worst case, iterative refinement can expand a summary Abox into the original Abox, so an effective refinement strategy is critical. The refinement step for an individual s in a justification $\mathcal{J}$ is as follows. For each a in the image of s, define $key(a)$ w.r.t. $\mathcal{J}$ to be the set of role assertions in $\mathcal{J}$ for which a has a corresponding role assertion in the original $\mathcal{A}$. That is, $key(a) =$

$$\left\{ R(t,s) \left| \begin{array}{l} \mathbf{f}(a) = s \wedge \\ R(t,s) \in \mathcal{J} \wedge \\ \exists b \text{ in } \mathcal{A} \text{ s.t.} \\ R(b,a) \in \mathcal{A} \wedge \\ \mathbf{f}(b) = t \end{array} \right. \right\} \cup \left\{ R(s,t) \left| \begin{array}{l} \mathbf{f}(a) = s \wedge \\ R(s,t) \in \mathcal{J} \wedge \\ \exists b \text{ in } \mathcal{A} \text{ s.t.} \\ R(a,b) \in \mathcal{A} \wedge \\ \mathbf{f}(b) = t \end{array} \right. \right\}$$

To refine s, we partition its image so that all individuals in a partition have the same key w.r.t. $\mathcal{J}$. Each partition is mapped to a new summary individual, creating a refined summary Abox. Conversely, if all individuals in $\mathcal{A}$ mapped to a summary individual s have the same key w.r.t. $\mathcal{J}$, then s is precise w.r.t. $\mathcal{J}$. Thus, justification-based refinement leads to precise justifications in subsequent iterations.

In general, there can be multiple justifications corresponding to different inconsistencies. For example, let us add a constraint to Figure 1 that the role $isTaughtBy$ is functional. The summary Abox in Figure 2 now contains a spurious inconsistency, because m' and w' will be inferred to be the same individual because of the functional property, but m' and w' are instances of the disjoint

concepts Man and $Woman$ respectively. The justification for this inconsistency is: $c' : Course$, $m' : Man$, $w' : Woman$, $isTaughtBy(c', m')$, $isTaugtBy(c', w')$. Figure 3 illustrates the application of a refinement step to the refined summary in Figure 2. The individual c' participates in multiple role assertions, so it is replaced by c_x' and c_y'. The individuals $c1$ and $c3$, which have the same key $\{isTaughtBy(c', m')\}$, are mapped to c_x'. The individual $c2$ is mapped to c_y' because it has a different key $\{isTaughtBy(c', w')\}$. After this refinement step, this spurious inconsistency disappears.

Refinement Strategy

The justification refinement order is important. Here are some sample heuristics:

- A single refinement candidate s may belong to multiple justifications. In such a case, we define its key to be the set of role assertions in all justifications that s belongs to. However, this can lead to a large number of key combinations, and to needless partitioning. We therefore give preference to justifications that have no overlap.

- Smaller justifications are given priority over larger justifications.

- If there are two tested individuals in $\mathcal{J}$, it is possible that the inconsistency is due to the interaction between two $\neg Q$ type assertions. We therefore delay the refinement of such justifications until no other justifications are left in the summary, when it is more efficient to test each of these individuals separately.

- Once a given $\mathcal{J}$ has been selected for refinement, we track its transformation in successive iterations to avoid recomputation overhead, and to reach a conclusion quickly.

- We give higher priority to justifications that pertain to the query, with the hope that spurious inconsistencies disappear during their refinement.
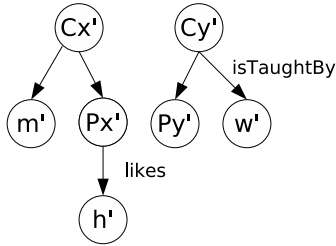

Figure 3: Abox after second refinement

Computing Justifications

To compute justifications efficiently, we use a technique called *tableau tracing* (initially defined for the logic $\mathcal{ALC}$ in (Baader & Hollunder 1993), and later extended to $\mathcal{SHIN}$ in (Kalyanpur 2006)). This technique involves extending a tableau reasoner to track the axioms responsible for the firing of each expansion rule. However, to implement our refinement strategy, it is desirable to find as many justifications

as possible at each refinement step. For this purpose we use Reiter's Hitting Set Tree (HST) algorithm as described in (Kalyanpur 2006), which recursively removes axioms from justifications so that new justifications can be found. However, since Reiter's approach is an exponential search algorithm, we have to impose a threshhold on the search. Therefore we find a subset of the justifications in a refinement step, and discover more in subsequent iterations.

As an optimization, we exploit similarities among justifications by forming justification patterns. Given a particular justification $\mathcal{J}$ for the inconsistent summary, we generalize it into a justification "pattern" by expressing it as a SPARQL query where individuals in are treated as variables. Note that we do not consider any of the Tbox or Rbox axioms in $\mathcal{J}$ while creating this query, only looking at assertions of the form $C(a), R(a, b), a \neq b$ present in $\mathcal{J}$. We execute this query against the summary ABox using a SPARQL engine to retrieve other "isomorphic" justifications, and then add the Tbox and Rbox axioms from $\mathcal{J}$ to each query result individually, to obtain valid new justifications. Since query pattern matching does not require any inferencing, the queries are fast. This optimization dramatically reduces the time taken to find additional similar justifications that would normally have been found one at a time as part of the exponential Reiter's search.

Drawing Conclusions on Justifications

In this section, we present a set of conditions that are sufficient to prove that all individuals mapped to a tested individual s in a precise justification $\mathcal{J}$ are solutions. These conditions depend only on the structure of $\mathcal{J}$.

Theorem 1. *Let $\mathbf{f}$ be a summary function mapping an Abox $\mathcal{A}$, consistent w.r.t. to its Tbox $\mathcal{T}$ and its Rbox $\mathcal{R}$, to its summary $\mathcal{A}'$. Let Q be a concept expression and let S be a subset of individuals in $\mathcal{A}'$ such that for all $s \in S$, $s : \neg Q \notin \mathcal{A}'$. For an individual $t \in S$, all individuals a such that $\mathbf{f}(a) = t$ are instances of Q (i.e. $(\mathcal{A}, \mathcal{T}, \mathcal{R}) \models a : Q$) if the following conditions hold:*

1. *there is an inconsistent subset IC of $\text{tested}(\mathcal{A}', Q, S)$, and*
2. *IC is precise, and*
3. *$\{s \in S \mid s : \neg Q \in IC\} = \{t\}$, and*
4. *IC is acyclic (i.e. the undirected graph induced by role and differentFrom assertions is acyclic)*

Proof. The main idea behind the proof is that, assuming the 4 conditions are satisfied and viewing $IC - \{t : \neg Q\}$ as a pattern, for each individual $a \in \mathcal{A}$ such that $\mathbf{f}(a) = t$, there is an instance I_a of the pattern $IC - \{t : \neg Q\}$ in $\mathcal{A}$ such that a is associated with t. Since IC is inconsistent, it follows that $I_a \cup \{a : \neg Q\}$ is inconsistent, hence a is an instance of Q (because I_a is a subset of $\mathcal{A}$). The existence of the instance I_a relies on Lemma 1 formulated in (Julian Dolby 2007). $\square$

In our algorithm, the subset IC of $tested(\mathcal{A}', Q, S)$ is always a minimal justification $\mathcal{J}$. Using the proof of Theorem 1, if $\mathcal{J}$ is precise and acyclic, then we can view it as a pattern, and conclude that there are corresponding patterns
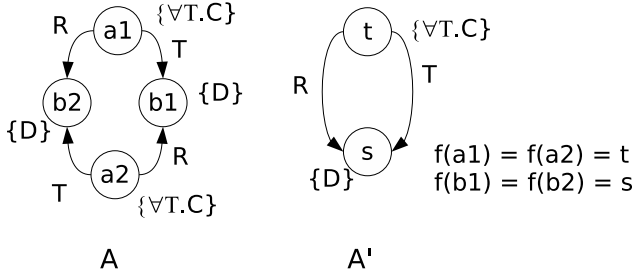
Figure 4: Example with Cycle

in $\mathcal{A}$ which match $\mathcal{J} - \{t : \neg Q\}$. However, if $\mathcal{J}$ is precise and cyclic, we cannot draw this conclusion. Consider the example in Figure 4, and let $Q = \exists R.C$ and $S = \{t\}$. There is a precise justification $\mathcal{J} = \{t : \forall T.C, R(t, s), T(t, s), t : \neg Q\}$, but there is no pattern in $\mathcal{A}$ which matches $\mathcal{J} - \{t : \neg Q\}$.

Fortunately, in many cases, we can efficiently transform part of the summary Abox so that Theorem 1 is still applicable. Theorem 2 allows us to apply deterministic tableau expansion rules (Horrocks, Sattler, & Tobies 2000) to a precise justification, and then check for an acyclic justification. More specifically, applying deterministic tableau expansion rules to a precise justification results in a new precise inconsistent subset of a summary of an Abox equivalent to the original Abox. To motivate this transformation, consider the image imL of $L = \mathcal{J} - \{t : \neg Q\}$ in $\mathcal{A}$. If we were to apply deterministic tableau rules to imL, the result would be $\widehat{imL} = imL \cup \{b1, b2 : C\}$, which is logically equivalent to imL. Now observe that $\widehat{L} = L \cup \{s : C\}$ is a precise summary of $\widehat{imL}$, and can be obtained by applying deterministic tableau expansion rules to L. Furthermore, $tested(\widehat{L}, S, Q)$ has a precise and acyclic justification $\widehat{\mathcal{J}} = \{R(t, s), s : C, t : \neg Q\}$, so that we can now conclude that $a1$ and $a2$ are instances of Q by directly using Theorem 1.

Theorem 2. *Let* **f** *be a summary function mapping an Abox* $\mathcal{A}$ *to its summary* $\mathcal{A}'$ *and* L *be a subset of* $\mathcal{A}'$. *Let* imL *denote the image of* L *in* $\mathcal{A}$ *(i.e.* $imL = \{a : C \in \mathcal{A}|\mathbf{f}(a)$ *is in* $L\} \cup \{R(a, b) \in \mathcal{A}|\mathbf{f}(a)$ *and* $\mathbf{f}(b)$ *are individuals in* $L\} \cup \{a \dot{\neq} b \in \mathcal{A}|\mathbf{f}(a)$ *and* $\mathbf{f}(b)$ *are individuals in* $L\}$). *Let* $\widehat{L}$ *denote the Abox obtained after the application of deterministic tableau expansion rules on* L. *If* L *is precise, then there is an Abox* $\widehat{imL}$ *equivalent to* imL *such that* $\widehat{L}$ *is a summary of* $\widehat{imL}$ *and* $\widehat{L}$ *is precise.*

Proof. See (Julian Dolby 2007) for more details. □

The following corollary generalizes the idea illustrated in the previous example:

Corollary 1. *Let* **f** *be a summary function mapping an Abox* $\mathcal{A}$, *consistent w.r.t. to its Tbox* $\mathcal{T}$ *and its Rbox* $\mathcal{R}$, *to its summary* $\mathcal{A}'$. *Let* Q *be a concept expression and let* S *be a*

subset of individuals in $\mathcal{A}'$ such that for all $s \in S$, $s : \neg Q \notin \mathcal{A}'$.

For an individual $t \in S$, all individuals a such that $\mathbf{f}(a) = t$ are instances of Q (i.e. $(\mathcal{A}, \mathcal{T}, \mathcal{R}) \models a : Q$) if the following conditions hold:

1. *there is a precise subset* L *of* $tested(\mathcal{A}', Q, S)$, *and*

2. *the Abox* $\widehat{L}'$ *obtained after the application of deterministic tableau expansion rules on* $L' = L - \{x : \neg Q | x \in S\}$ *is such that* $tested(\widehat{L}', Q, S)$ *has an acyclic[3] inconsistent subset* IC, *and*

3. $\{s \in S \mid s : \neg Q \in IC\} = \{t\}$

Proof. Direct consequence of theorems 2 and 1 □

In general, applying deterministic rules on a subset L of a summary of a consistent Abox $\mathcal{A}$ may still be insufficient to directly find solutions of the query in $\mathcal{A}$. Consider the example in Figure 4, and let $Q = \exists R.D \sqcap \exists T.D$ and $S = \{t\}$. $\mathcal{J} = \{s : D, R(t, s), T(t, s), t : \neg Q\}$ is a precise justification. No deterministic rule is applicable to $\mathcal{J} - \{t : \neg Q\}$. However, the two Aboxes $\mathcal{J}_1 = \mathcal{J} \cup \{t : \forall R.\neg D\}$ and $\mathcal{J}_2 = \mathcal{J} \cup \{t : \forall T.\neg D\}$ corresponding to the two branches resulting from the application of the non-deterministic $\sqcup$-rule to satisfy $t : \neg Q$ have acyclic precise inconsistent subsets, which, according to the following Theorem 3, is enough to conclude that $a1$ and $a2$ are instances of Q.

Theorem 3. *Let* **f** *be a summary function mapping an Abox* $\mathcal{A}$, *consistent w.r.t. to its Tbox* $\mathcal{T}$ *and its Rbox* $\mathcal{R}$, *to its summary* $\mathcal{A}'$. *Let* Q *be a concept expression and let* S *be a subset of individuals in* $\mathcal{A}'$ *such that for all* $s \in S$, $s : \neg Q \notin \mathcal{A}'$. *For an individual* $t \in S$, *all individuals* a *in* $\mathcal{A}$ *such that* $\mathbf{f}(a) = t$ *are instances of* Q *(i.e.* $(\mathcal{A}, \mathcal{T}, \mathcal{R}) \models a : Q$*) if the following conditions hold:*

1. *there is an inconsistent subset* IC *of* $tested(\mathcal{A}', Q, S)$, *and*

2. IC *is precise, and*

3. $\{s \in S \mid s : \neg Q \in IC\} = \{t\}$, *and*

4. *there are concepts* C *and* D *such that* $t : C \sqcup D \in IC$, $t : C \notin IC$ *and* $t : D \notin IC$), *and*

5. *Each of the Aboxes* $IC_1 = IC \cup \{t : C\}$ *and* $IC_2 = IC \cup \{t : D\}$ *has at least one acyclic inconsistent subset.*

Proof. The proof relies on the simple observation that, for an Abox $\mathcal{A}$, $(\mathcal{A} \cup \{x : C \sqcup D\}, \mathcal{T}, \mathcal{R}) \models a : Q$ is equivalent to $(\mathcal{A} \cup \{x : C\}, \mathcal{T}, \mathcal{R}) \models a : Q$ and $(\mathcal{A} \cup \{x : D\}, \mathcal{T}, \mathcal{R}) \models a : Q$. See (Julian Dolby 2007) for more details. □

Unfortunately, despite all the techniques presented in this section, there are precise cyclic justifications that remain inconclusive. For example, in Figure 4, let $Q = \neg((\forall R.\forall T^{-1}.A \sqcap \neg A) \sqcup (\forall R.\forall T^{-1}.B \sqcap \neg B))$ and $S = \{t\}$. There is a precise cyclic justification $\mathcal{J} = \{R(t, s), T(t, s), t : \neg Q\}$. The previous technique does not work because $\mathcal{J} \cup \{t : \forall R.\forall T^{-1}.A \sqcap \neg A\}$ and $\mathcal{J} \cup \{t :$

---

[3]Once again, acyclicity is defined w.r.t. to the undirected graph induced by role and differentFrom assertions

Dataset	type assertions	role assertions
UOBM-1	25,453	214,177
UOBM-10	224,879	1,816,153
UOBM-30	709,159	6,494,950

Table 1: Dataset Statistics

Reasoner	Dataset	Avg. Time	St.Dev	Range
KAON2	UOBM-1	20.7	1.2	18-37
KAON2	UOBM-10	447.6	23.3	414.8-530
SHER	UOBM-1	4.2	3.8	2.4-23.8
SHER	UOBM-10	15.4	25.6	6.4-191.1
SHER	UOBM-30	34.7	63.5	11.6-391.1

Table 2: Runtimes (sec)

$\forall R.\forall T^{-1}.B \sqcap \neg B\}$ don't have any acyclic inconsistent subset. In fact, $a1$ and $a2$ are not solutions. In such cases, we refine an individual x in $\mathcal{J}$ by dividing the image set of x arbitrarily into two new summary graph individuals. In all of the queries that we have processed so far, none have required this fall-back.

Evaluation

Our algorithms are implemented in a system called SHER, which includes additional optimizations (Julian Dolby 2007). We evaluated it on the UOBM benchmark which was modified to $\mathcal{SHIN}$ expressivity. We issued instance retrieval queries for the 112 concepts in the ontology. The results are reported for 1, 10, and 30 universities, which are referred to as UOBM-1, UOBM-10 and UOBM-30. We compared our results against KAON2 (Hustadt, Motik, & Sattler 2004). (Pellet (Sirin & Parsia 2004) did not scale to even one university.) For KAON2, we set all maximum cardinality restrictions to one because of KAON2 limitations. The runs were made on a 64 bit AMD dual processor 8G RAM Linux machine. The Abox was stored in DB2 for SHER and MySQL for KAON2.

The size of the datasets are given in Table 1. Table 2 summarizes the times taken (in seconds) by KAON2 and SHER solely for query answering, i.e., in both cases, the times do not include the knowledge base pre-processing and setup costs. KAON2 ran out of memory on UOBM-30. In 111 out of 112 queries SHER and KAON2 had 100% agreement. The difference in the one remaining was due to differences in the constraints used. As can be seen, the average runtimes for SHER are significantly lower, usually by an order of magnitude, than those for KAON2. For this particular example, SHER scaled in a sublinear fashion.

Related Work and Conclusions

Optimized tableau algorithms exist for Aboxes in secondary storage, but they either assume role-free Aboxes (Horrocks *et al.* 2004), relatively inexpressive-DLs (Calvanese *et al.* 2005), or require pre-processing of the Abox to make it role-free (Li 2004). KAON2, which we included in our evaluation, is a non-tableau based approach that relies on translating Description Logic to disjunctive datalog (Hustadt,

Motik, & Sattler 2004). Our summarization-and-refinement strategy is an improvement over the divide-and-conquer (binary instance retrieval) approach (Haarslev & Moller 2002) implemented in state-of-the-art tableau reasoners to test potential solutions to the query. Our approach provides a better partitioning of tested individuals through summarization and refinement, the ability to conclude directly that all tested individuals are solutions without necessarily testing each of them in isolation, and explanations for solutions.

References

A.Fokoue; A.Kershenbaum; L.Ma; E.Schonberg; and K.Srinivas. 2006. The summary abox: Cutting ontologies down to size. *Proc. of the Int. Semantic Web Conf. (ISWC 2006)* 136–145.

Baader, F., and Hollunder, B. 1993. Embedding defaults into terminological knowledge representation formalisms. In *Technical Report RR-93-20*.

Calvanese, D.; Giacomo, G. D.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2005. Dl-lite: Tractable description logics for ontologies. *Proc. of AAAI*.

Haarslev, V., and Moller, R. 2002. Optimization strategies for instance retrieval. *Proceedings of the international workshop on description logics (DL 2002)*.

Horrocks, I., and Tessaris, S. 2002. Querying the semantic web: a formal approach. In Horrocks, I., and Hendler, J., eds., *Proc. of the 1st Int. Semantic Web Conf. (ISWC 2002)*, number 2342 in Lecture Notes in Computer Science, 177–191. Springer-Verlag.

Horrocks, I.; Li, L.; Turi, D.; and Bechhofer, S. 2004. The instance store: Dl reasoning with large numbers of individuals. *Proceedings of the 2004 Description Logic Workshop*.

Horrocks, I.; Sattler, U.; and Tobies, S. 2000. Reasoning with individuals for the description logic SHIQ*. *Proc. of 17th Int.Conf. on Automated Deduction* 482–496.

Hustadt, U.; Motik, B.; and Sattler, U. 2004. Reducing shiq - description logic to disjunctive datalog programs. *Proc. of the 9th Int. Conf. on Knowledge Representation and Reasoning (KR 2004)*.

Julian Dolby, e. a. 2007. Technical report: Scalable semantic retrieval through summarization and refinement. In *http://www.research.ibm.com/iaa/techReport2007.pdf*.

Kalyanpur, A. 2006. *Debugging and Repair of OWL-DL Ontologies*. Ph.D. Dissertation, University of Maryland, https://drum.umd.edu/dspace/bitstream/1903/3820/1/umi-umd-3665.pdf.

Li, L. 2004. Reasoning with large numbers of individuals moves on: extending the instance store.

Ma, L.; Yang, Y.; Qiu, Z.; Xie, G.; and Pan, Y. 2006. Towards a complete owl ontology benchmark. In *Proc. of the third European Semantic Web Conf.(ESWC 2006)*, 124–139.

Sirin, E., and Parsia, B. 2004. Pellet: An owl dl reasoner. In *Description Logics*.