

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{SHLN} , 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 expres-

sive Aboxes in databases. Before processing any queries, we create a *summary Abox* \mathcal{A}' (A.Fokoue *et al.* 2006b) 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 *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 *SHIN* expressivity, where we show dramatic reductions in space and time requirements for instance retrieval.

Background

We assume ontologies of SHIN expressiveness. SHIN is equivalent to OWL-DL without nominals and datatype reasoning, as shown in Table 1. In the semantic definition of SHIN, $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ refers to an interpretation where $\Delta^{\mathcal{I}}$ is a non-empty set (the domain of the interpretation), and the interpretation function $\cdot^{\mathcal{I}}$ maps every atomic concept C to a set $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, every atomic role R to a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and every individual a to $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. $\text{Trans}(R)$ refers to a transitive role R .

An RBox \mathcal{R} is a finite set of transitivity axioms of the form $\text{Trans}(R)$ and role inclusion axioms of the form $R \sqsubseteq P$ where R and P are roles. \sqsubseteq^* denotes the reflexive transitive closure of the \sqsubseteq relation on roles. A Tbox \mathcal{T} is a set of concept inclusion axioms of the form $C \sqsubseteq D$ where C and D are concept expressions. An ABox \mathcal{A} is a set of axioms of the form $a : C$, $R(a, b)$, and $a \neq b$.

An interpretation \mathcal{I} is a model of an ABox \mathcal{A} w.r.t. a Tbox \mathcal{T} and a Rbox \mathcal{R} iff it satisfies all the axioms in \mathcal{A} , \mathcal{R} , and \mathcal{T} (see Table 2). An ABox \mathcal{A} is said to be consistent w.r.t. a Tbox \mathcal{T} and a Rbox \mathcal{R} iff there is a model of \mathcal{A} w.r.t. \mathcal{T} and \mathcal{R} . If there is no ambiguity from the context, we simply say that \mathcal{A} is consistent.

A key feature of our approach is the construction of a summary ABox \mathcal{A}' corresponding to the ABox \mathcal{A} (A.Fokoue *et al.* 2006b). 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 *SHIN* 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 $a \neq b \in \mathcal{A}$ then $\mathbf{f}(a) \neq \mathbf{f}(b) \in \mathcal{A}'$

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} . 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 are a and b in \mathcal{A} such that $a' = \mathbf{f}(a)$, $b' = \mathbf{f}(b)$ and $R(a, b) \in \mathcal{A}$.

Definitions	Semantics
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$\exists R.C$	$\{x \mid \exists y. \langle x, y \rangle \in R^{\mathcal{I}}, y \in C^{\mathcal{I}}\}$
$\forall R.C$	$\{x \mid \forall y. \langle x, y \rangle \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}}\}$
$\leq nR$	$\{x \mid \{ \langle x, y \rangle \in R^{\mathcal{I}} \} \leq n\}$
$\geq nR$	$\{x \mid \{ \langle x, y \rangle \in R^{\mathcal{I}} \} \geq n\}$
R^-	$\{ \langle x, y \rangle \mid \langle y, x \rangle \in R^{\mathcal{I}} \}$

Table 1: SHIN Description Logic Constructors

Axioms	Satisfiability conditions
$\text{Trans}(R)$	$(R^{\mathcal{I}})^+ = R^{\mathcal{I}}$
$R \sqsubseteq P$	$\langle x, y \rangle \in R^{\mathcal{I}} \Rightarrow \langle x, y \rangle \in P^{\mathcal{I}}$
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$a : C$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
$R(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$
$a \neq b$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$

Table 2: SHIN Description Logic Axioms

- (5) If for all $x \in \mathcal{A}$, $a \neq x \notin \mathcal{A}$, $b \neq x \notin \mathcal{A}$, and $\mathcal{L}(a) = \mathcal{L}(b)$, then $\mathbf{f}(a) = \mathbf{f}(b)$.
- (6) $\mathbf{f}(a) \neq \mathbf{f}(b) \in \mathcal{A}'$ implies a is the only individual in \mathcal{A} mapped to $\mathbf{f}(a)$ (same for b).

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.

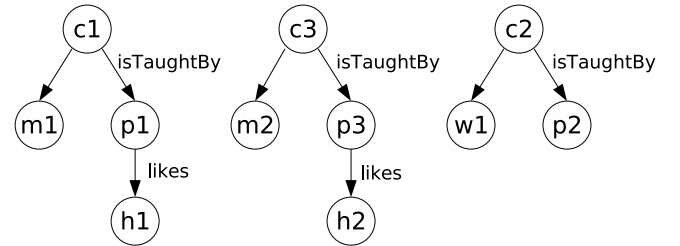


Figure 1: Example ABox

Consider the query for the concept *PeopleWithHobby*, which is defined as $\text{Person} \sqcap \geq 1 \text{likes}$. 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

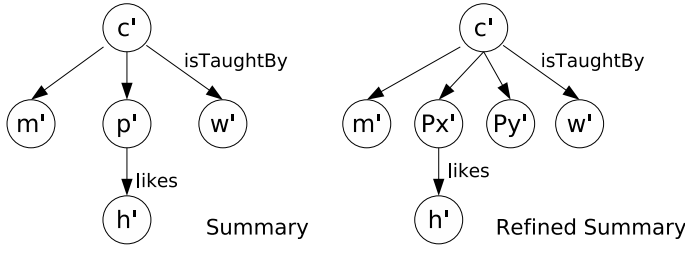


Figure 2: Summary Abox

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 $s \in S$, $s : \neg Q \notin \mathcal{A}$.¹ 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, $tested(\mathcal{A}, Q, S) = \mathcal{A} \cup \{s : \neg Q \mid s \in S\}$.

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

¹Note 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.

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 $tested(\mathcal{A}', Q, S)$. 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 $t \in H$, $s \neq t \in H$ (resp. $t \neq s \in H$) 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 $a \neq b \in \mathcal{A}$ (resp. $b \neq a \in \mathcal{A}$); and
3. There is an individual $a \in \mathcal{A}$ such that $\mathbf{f}(a) = s$; and
4. $s : C \in H - \{x : \neg Q \mid x \in S\}$ 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.

```

 $S \leftarrow \{x \mid x \in \text{individuals in } \mathcal{A}' \text{ and } x : \neg Q \notin \mathcal{A}'\}$ ;
 $R \leftarrow \mathcal{A}'$ ;
 $Results \leftarrow \emptyset$ ;
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 \mathcal{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 I'_R be the individuals in \mathcal{A}'_R . Let s be an individual in \mathcal{A}' . We say that \mathcal{A}'_R is a refinement of \mathcal{A}' w.r.t. s iff there are individuals $s_1 \dots s_n$, $n > 1$, in \mathcal{A}'_R such that:

1. $I'_R = (I' - \{s\}) \cup \{s_1 \dots s_n\}$ where $s_1 \dots s_n \notin I'$
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: $c' : Course$, $m' : Man$, $w' : Woman$, $isTaughtBy(c', m')$, $isTaughtBy(c', w')$. 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 as quickly as possible.
- We give higher priority to justifications that pertain to the query (i.e. those that contain a tested individual in the justification.) In making the summary more precise by refining query-pertinent justifications, spurious inconsistencies may disappear.

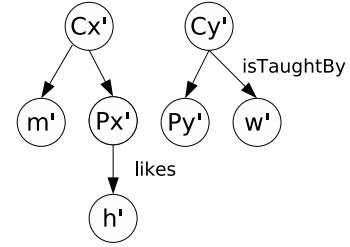


Figure 3: ABox after second refinement

Computing and Selecting Justifications

In order to compute justifications efficiently, we resort to a technique known as *tableau tracing*. The idea, initially proposed in (Baader & Hollunder 1993) for the logic *ALC*, and later extended to *SHIN* (Kalyanpur 2006), involves modifying the internals of a tableau reasoner by keeping track of the axioms responsible for each of its operations and events. Briefly, the process works as follows (the details are beyond the scope of this report, see (Kalyanpur 2006)): DL-based tableau reasoners check consistency of an ABox by building an abstraction of a model for it, known as a *completion graph*, which is constructed by repeatedly applying a set of *expansion rules*. If the ontology is inconsistent, it has no model, and thus the completion graph contains *clashes* or contradictions. In our problem, the goal is no longer constructing a model for the input, but identifying which axioms in the input ABox are responsible for the contradictions that prevent the model from being built. We do this by keeping track of the axioms responsible for firing each of the expansion rules (while building the graph), and finally examine the axiom trace of the clash event to obtain a justification for the inconsistent ABox.

However, while the above technique helps find a single justification easily, finding additional justifications is not straightforward. This is because the non-deterministic tableau algorithm typically terminates when an inconsistency-revealing clash is found, whereas we need it to explore any remaining choices and fire additional applicable expansion rules in order to reveal more clashes (thereby

exposing more justifications) inherent in the ABox. This, in effect, causes all the reasoner optimizations to be turned off, which is computationally very expensive. In order to resolve this problem, we use a workaround that involves the classical Reiter's Hitting Set Tree (HST) algorithm (Reiter 1987).

The basic principle behind the workaround strategy is to iteratively discover additional justifications by starting with a single justification, removing each of the axioms in the justification from the original ABox and then finding any new justifications in the modified ABox (using the tableau tracing procedure again). This process is repeated recursively till no new justifications are found. By drawing an analogy with Reiter's HST algorithm, we can benefit from many of the optimizations present therein (for details see (Kalyanpur 2006)).

The above technique provides a definite solution for finding all justifications. However, there is a trade-off between the time taken to compute all the justifications, which can be large due to the exponential nature of Reiter's search, versus the reduction in refinement time achieved by finding all justifications at each iteration of the refinement process. To address this trade-off, we impose a user-definable cut-off point (upper-threshold) for the subroutine which computes justifications, and perform an empirical analysis to fix a threshold value that works well in practice.

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 the following Lemma 1 : \square

Lemma 1. *Let \mathbf{f} be a summary function mapping an Abox \mathcal{A} to its summary \mathcal{A}' . Let Q be a concept expression and let*

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

If L is a non-empty acyclic precise subset of $\text{tested}(\mathcal{A}', Q, S)$, then, for each individual u of L and a of \mathcal{A} such that $\mathbf{f}(a) = u$, there exists a pair (I, ρ) such that I is a subset of \mathcal{A} and ρ is a total mapping from $\text{Indiv}(L)$ to $\text{Indiv}(I)$, where $\text{Indiv}(X)$ denotes the set of individuals in an Abox X . Furthermore, (I, ρ) satisfies the following properties for all individuals r and s in L :

- (a) $\rho(u) = a$
- (b) if $R(r, s) \in L$ then $R(\rho(r), \rho(s)) \in I$
- (c) if $r \neq s \in L$ then $\rho(r) \neq \rho(s) \in I$
- (d) if $s : D \in (L - \{x : \neg Q \mid x \in S\})$ then $\rho(s) : D \in I$

Proof. We prove Lemma 1 by induction on the number of individuals in L .

First, we consider the case where L has a single individual u . Let a be an individual of \mathcal{A} such that $\mathbf{f}(a) = u$. We define I as $I = \{a : D \mid u : D \in (L - \{x : \neg Q \mid x \in S\})\}$ and ρ is the mapping from $\{u\}$ to $\{a\}$ such that $\rho(u) = a$. The pair (I, ρ) satisfies the properties (b) and (c) because, since L is acyclic, $u \neq u \notin L$ and $R(u, u) \notin L$. It satisfies (a) by definition of ρ . It satisfies (d) and I is a subset of \mathcal{A} as a direct consequence of the fact that u is precise (4th property in the definition of a precise individual) and $\mathbf{f}(a) = u$.

Next, we assume that if the number of individuals in L is less than or equal to n (for n an integer such that $n \geq 1$), then, for any $(u, a) \in L \times \mathcal{A}$ such that $\mathbf{f}(a) = u$, there is a pair (I, ρ) satisfying all the requirements of Lemma 1. We prove that if L has $n + 1$ individuals such pairs still exist.

Let L be a non-empty acyclic subset of $\text{tested}(\mathcal{A}', Q, S)$ such that all its $n + 1$ individuals are precise w.r.t. L . Let u be an individual in L and a an individual in \mathcal{A} such that $\mathbf{f}(a) = u$.

The proof consists of three important steps. First, we establish properties of individuals of L directly "connected" to u (we will relate them to individuals in \mathcal{A} directly "connected" to a in the same way). Second, after removing u from L and its connections to other individuals in L (i.e. all role and differentFrom assertions relating u to another individual), we end up with disjoint subsets of L containing less than $n + 1$ individuals so that, by using the induction hypothesis, we can show that there are pairs (I', ρ') satisfying all the requirements of Lemma 1 on these subsets of L . Finally, using these pairs (I', ρ') , we show the existence of a pair (I, ρ) satisfying all the requirements of Lemma 1 on L .

First, we consider the set Con_u of individuals of L connected to u in L by a role or differentFrom assertion $Con_u = \{x \mid R(u, x) \in L \text{ or } R(x, u) \in L \text{ or } u \neq x \in L \text{ or } x \neq u \in L\}$. For an element $v \in Con_u$, since L is acyclic, there is only one connection, denoted $[u, v]$, between u and v (this connection is one of : $R(u, v)$, $R(v, u)$, $u \neq v$, or $v \neq u$). Regardless of the exact nature of the connection ($R(u, v)$, $R(v, u)$, $u \neq v$, or $v \neq u$), the fact that u is precise in L implies that there is an individual b in \mathcal{A} such that $\mathbf{f}(b) = v$ and b is connected to a in the same way v is connected to u (e.g. if $R(u, v) \in L$, then $R(a, b) \in \mathcal{A}$). We define a mapping β from Con_u to

$Indiv(\mathcal{A})$ as follows: for an element v in Con_u , $\beta(v)$ is chosen to be an individual b in \mathcal{A} such that $\mathbf{f}(b) = v$ and the connection between a and b in \mathcal{A} is of the same nature as the connection between u and v in L .

Now, we consider the subset L' of L obtained after removing u and all connections relating u to another individual in L . Formally, $L' = L - (\{R(u, x) \in L \mid x \in Con_u\} \cup \{R(x, u) \in L \mid x \in Con_u\} \cup \{u \neq x \in L \mid x \in Con_u\} \cup \{x \neq u \in L \mid x \in Con_u\} \cup \{u : D \mid u : D \in L\})$. For an element $v \in Con_u$, we define $Reach_v$ as the set of elements of L' that contains v and all elements reachable from v in L' . Since L is acyclic, for v and w elements of Con_u , $v \neq w$ implies that $Reach_v$ and $Reach_w$ are disjoint. Furthermore, by definition of L' and $Reach_v$, for all $v \in Con_u$, $u \notin Reach_v$. For an individual $v \in Con_u$, L'_v denotes the greatest subset of L' containing only individuals in $Reach_v$. For all $(v, w) \in Con_u^2$ such that $v \neq w$, since $Reach_v$ and $Reach_w$ are disjoint, it follows that L'_v and L'_w are also disjoint. Finally, let $Rest'$ be the subset of L' defined as $Rest' = L' - \bigcup_{v \in Con_u} L'_v$. By its definition, $Rest'$ cannot contain u and is disjoint with any L'_v .

For an element $v \in Con_u$, since L'_v is a non-empty acyclic subset of $tested(\mathcal{A}', Q, S)$ such that all its k individuals (with $k \leq n$) are precise w.r.t. L'_v , $\beta(v) \in \mathcal{A}$ and $\mathbf{f}(\beta(v)) = v$, it follows, from the induction hypothesis, that there is a pair (I_v, ρ_v) , where I_v is a subset of \mathcal{A} and ρ_v is a total mapping from $Indiv(L'_v)$ to $Indiv(I_v)$, satisfying $\rho_v(v) = \beta(v)$ and all the properties (b) to (d) of Lemma 1. Assuming $Rest'$ is not empty, we choose, randomly, an individual denoted z in $Rest'$ (since z is precise, we know that there must be an individual $za \in \mathcal{A}$ such that $\mathbf{f}(za) = z$). Likewise, there is a pair (I_z, ρ_z) , where I_z is a subset of \mathcal{A} and ρ_z is a total mapping from $Indiv(Rest')$ to $Indiv(I_z)$, satisfying $\rho_z(z) = za$ and all the properties (b) to (d) of Lemma 1.

From the pairs (I_x, ρ_x) obtained on all the disjoint subsets of L' , we can now define the pair (I, ρ) satisfying all the requirements of Lemma 1 for L . We define I as follows:

$$I = \begin{cases} A & \text{if } Rest' = \emptyset \\ A \cup I_z & \text{otherwise} \end{cases}$$

where A is defined as:

$$A = \begin{aligned} & \{a : D \mid u : D \in L - \{x : \neg Q \mid x \in S\}\} \\ & \bigcup \{R(a, \beta(v)) \mid R(u, v) \in L\} \\ & \bigcup \{R(\beta(v), a) \mid R(v, u) \in L\} \\ & \bigcup \{a \neq \beta(v) \mid u \neq v \in L\} \cup \{\beta(v) \neq a \mid v \neq u \in L\} \\ & \bigcup_{v \in Con_u} I_v \end{aligned}$$

ρ is defined as the mapping from $Indiv(L)$ to $Indiv(I)$ as follows: for an individual x in L ,

$$\rho(x) = \begin{cases} a & \text{if } x = u \\ \rho_v(x) & \text{if } x \in Reach_v \text{ for some } v \in Con_u \\ \rho_z(x) & \text{otherwise (i.e. } x \in Rest') \end{cases}$$

It is obvious that, due to the properties satisfied by the pairs (I_x, ρ_x) obtained on the disjoint subsets of L' and by the definition of β , (I, ρ) defined previously satisfies all the requirements of Lemma 1. \square

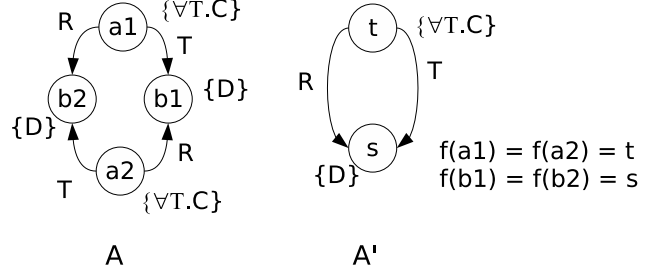


Figure 4: Example with Cycle

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 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 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 \mathbf{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. $imL = \{a : C \in \mathcal{A} \mid \mathbf{f}(a) \text{ is in } L\} \cup \{R(a, b) \in \mathcal{A} \mid \mathbf{f}(a) \text{ and } \mathbf{f}(b) \text{ are individuals in } L\} \cup \{a \neq b \in \mathcal{A} \mid \mathbf{f}(a) \text{ and } \mathbf{f}(b) \text{ are individuals in } L\}$). 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. First, we observe that applying a deterministic tableau expansion rule on a Abox yields an equivalent Abox.

Assuming that L is precise, we show that the application of a deterministic tableau rule to an individual s in L yields an abox \widehat{L} which is a precise summary of the Abox obtained

by applying the same deterministic rule on all the individuals of imL mapped to s (i.e. individuals a such that $\mathbf{f}(a) = s$).

\exists -rule: Let assume that the \exists -rule is applied to s in order to satisfy the constraint $s : \exists R.C \in L$. The resulting Abox \widehat{L} is such that $\widehat{L} = L \cup \{R(s, t), t : C\}$ where t is a new individual not present in L .

By definition of the applicability of the \exists -rule, there are no R -neighbors y of s in L such that $y : C \in L$. Let a be an individual of imL such that $\mathbf{f}(a) = s$. Since s is precise w.r.t. L , $a : \exists R.C \in imL$. Since L is a summary of imL , there are no R -neighbors b of a in imL such that $b : C \in imL$. So we can apply the \exists -rule on a to satisfy $a : \exists R.C \in imL$. This will result in the creation of a new individual, denote $\beta(a)$, such that $\beta(a) : C$ and $R(a, \beta(a))$. Let \widehat{imL} be the result of the application of the \exists -rule to all such individuals a . $\widehat{imL} = imL \cup \{\beta(a) : C | \mathbf{f}(a) = s\} \cup \{R(a, \beta(a)) | \mathbf{f}(a) = s\}$. It follows that \widehat{L} is a precise summary of \widehat{imL} with the summary function $\widehat{\mathbf{f}}$ defined as follows: for all individual a in \widehat{imL} ,

$$\widehat{\mathbf{f}}(a) = \begin{cases} \mathbf{f}(a) & \text{if } a \in \text{Indiv}(imL) \\ t & \text{otherwise} \end{cases}$$

\geq -rule: Similar to \exists -rule.

\forall -rule: Let assume that the \forall -rule is applied to s in order to satisfy the constraints $s : \forall R.C \in L$ and t is a R -neighbor of s (for simplicity of the presentation, in the remainder of the proof, we assume that t is a R -neighbor of s because $S(s, t) \in L$ with $S \sqsubseteq^* R$). The resulting Abox \widehat{L} is such that $\widehat{L} = L \cup \{t : C\}$.

Let \widehat{imL} be the Abox obtained from imL by applying the \forall -rule to satisfy the constraint $a : \forall R.C \in imL$ to all individuals a and b in imL such that $\mathbf{f}(a) = s$, $\mathbf{f}(b) = t$ and $S(a, b)$ (Note that such $a : \forall R.C \in imL$ because s is precise w.r.t. L and $s : \forall R.C \in L$).

Clearly that $\widehat{imL} = imL \cup \{b : C | S(a, b) \in imL \text{ and } \mathbf{f}(a) = s \text{ and } \mathbf{f}(b) = t\}$. For an individual b in imL such that $\mathbf{f}(b) = t$, since t is precise in L and $S(s, t) \in L$, there exists an individual a such that $\mathbf{f}(a) = s$ and $S(a, b) \in imL$. The application of \forall -rule on a ($a : \forall R.C$ must be in imL since s is precise w.r.t. L) insures that $b : C \in \widehat{imL}$, which establishes that \widehat{L} is a precise summary of \widehat{imL} with \mathbf{f} as its summary function.

\forall_+ -rule: Similar to \forall -rule.

Unfol-rule: Similar to \sqcap -rule.

\leq -rule: Let assume that the \leq -rule is applied to s in order to satisfy the constraints $s : \leq 1R \in L$ and r and t are R -neighbors of s such that $r \neq t \notin L$ and $t \neq r \notin L$. It results in the merger of t into r (for simplicity of the presentation, in the remainder of the proof, we assume, without loss of generality, that r and t are R -neighbors of s because $S(s, r) \in L$ and $S(s, t) \in L$ with $S \sqsubseteq^* R$). The resulting Abox \widehat{L} is such that $\widehat{L} = (L - \text{Rem}) \cup \text{Add}$ where,

$$\begin{aligned} \text{Rem} &= \{t : C \in L\} \cup \{T(x, t) \in L | \text{for a role } T\} \cup \\ &\{T(t, x) \in L | \text{for a role } T\} \cup \{x \neq t \in L\} \cup \{t \neq x \in L\} \\ \text{Add} &= \{r : C | t : C \in L\} \cup \{T(x, r) | T(x, t) \in L\} \cup \\ &\{T(r, x) | T(t, x) \in L\} \cup \{x \neq r | x \neq t \in L\} \cup \{r \neq x | t \neq x \in L\} \end{aligned}$$

Let \widehat{imL} be the Abox obtained from imL by applying the standard \leq -rule to satisfy the constraint $a : \leq 1R \in imL$ to all individuals a, b, c in imL such that $\mathbf{f}(a) = s$, $\mathbf{f}(b) = r$, $\mathbf{f}(c) = t$, $S(a, b)$ and $S(a, c)$ (Note that $a : \leq 1R \in imL$ because s is precise w.r.t. L and $s : \leq 1R \in L$. Also b and c cannot be asserted to be distinct because $\mathbf{f}(b)$ and $\mathbf{f}(c)$ are not asserted to be distinct in the summary. So \leq -rule is applicable and its application merges c and b).

Let $\widehat{\mathbf{f}}$ be the mapping from \widehat{imL} to \widehat{L} defined as follows: for all individuals x in \widehat{imL} ,

$$\widehat{\mathbf{f}}(x) = \begin{cases} \mathbf{f}(x) & \text{if } \mathbf{f}(x) \neq t \\ r & \text{otherwise (i.e. } \mathbf{f}(x) = t) \end{cases}$$

\widehat{L} is a precise summary of \widehat{imL} with $\widehat{\mathbf{f}}$ as its summary function as a direct consequence of the following facts:

– For an individual c in imL such that $\mathbf{f}(c) = t$, since t is precise in L and $S(s, t) \in L$, there exists an individual a such that $\mathbf{f}(a) = s$ and $S(a, c) \in imL$. Furthermore, since s is precise in L , $s : \leq 1R \in L$ and $S(s, r)$, there exists an individual b such that $\mathbf{f}(b) = r$, $a : \leq 1R \in imL$ and $S(a, b) \in imL$ (once again, c and b cannot be asserted to be distinct). Therefore, the application of the \leq -rule would have resulted in the merger of c into b .

– Likewise, for an individual b in imL such that $\mathbf{f}(b) = r$, there are a and c in imL such that $\mathbf{f}(c) = t$, $S(a, b) \in imL$ and $S(a, c) \in imL$. Therefore, the application of the \leq -rule would have resulted in the merger of c into b .

\sqcap -rule: If the \sqcap -rule is applied to s to satisfy $s : C \sqcap D \in L$, then $\widehat{L} = L \cup \{s : C, s : D\}$. Since s is precise w.r.t. L , all individuals a in imL such that $\mathbf{f}(a) = s$ are such that $a : C \sqcap D$. Since L is a summary of imL , the one of the assertions $a : C$ or $a : D$ is not in imL . So \sqcap -rule is applicable to a . Applying the same \sqcap -rule on all these individuals a yields $\widehat{imL} = imL \cup \{a : C | \mathbf{f}(a) = s\} \cup \{a : D | \mathbf{f}(a) = s\}$. \widehat{L} is obviously a precise summary of \widehat{imL} with \mathbf{f} as its summary function.

GCI-rule: Similar to \sqcap -rule.

□

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

Corollary 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 a precise subset L of $\text{tested}(\mathcal{A}', Q, S)$, and
2. the Abox \widehat{L}' obtained after the application of deterministic tableau expansion rules on $L' = L - \{x : \neg Q \mid x \in S\}$ is such that $\text{tested}(\widehat{L}', Q, S)$ has an acyclic² inconsistent subset IC , and
3. $\{s \in S \mid s : \neg Q \in IC\} = \{t\}$

Proof. Direct consequence of theorems 2 and 1 \square

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 a_1 and a_2 are instances of Q .

Theorem 3. 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 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 $\text{tested}(\mathcal{A}', Q, S)$, and
2. IC is precise, and
3. $\{s \in S \mid s : \neg Q \in IC\} = \{t\}$, and
4. 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 $(\mathcal{A} \cup \{x : D\}, \mathcal{T}, \mathcal{R}) \models a : Q$

IC_1 has an acyclic inconsistent subset H . If we consider a new query $Q' = \neg(\neg Q \sqcap C)$, then IC_1 is a precise subset of $\text{tested}(\mathcal{A}', Q', S)$ (direct consequence of the fact that it is a precise subset of $\text{tested}(\mathcal{A}', Q, S)$). Since H is a precise acyclic subset of $\text{tested}(\mathcal{A}', Q', S)$, Lemma 1 implies that, for each a in \mathcal{A} such that $\mathbf{f}(a) = t$, there is an instance I_a of the pattern $H - \{t : \neg Q'\}$ in \mathcal{A} where a is mapped to t . Since H is inconsistent, it follows that $I_a \cup \{a : \neg Q'\}$ is also inconsistent. This establishes that, for each a in \mathcal{A} such that $\mathbf{f}(a) = t$, $(\mathcal{A} \cup \{a : C\}, \mathcal{T}, \mathcal{R}) \models a : Q$. Likewise, for each a in \mathcal{A} such that $\mathbf{f}(a) = t$, $(\mathcal{A} \cup \{a : D\}, \mathcal{T}, \mathcal{R}) \models$

$a : Q$. Therefore, $(\mathcal{A} \cup \{a : C \sqcup D\}, \mathcal{T}, \mathcal{R}) \models a : Q$, which establishes $(\mathcal{A}, \mathcal{T}, \mathcal{R}) \models a : Q$ because $a : C \sqcup D \in \mathcal{A}$. \square

Note that condition (4) of Theorem 3 cannot be relaxed so as to apply to any individual r in IC because, in general, in the Abox, the solution a_1 that would be obtained from the instance of the acyclic inconsistent subset of $IC_1 = IC \cup \{r : C\}$ might not be identical to the solution a_2 that would be obtained from the instance of the acyclic inconsistent subset of $IC_2 = IC \cup \{r : D\}$

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 $\mathcal{J} \cup \{t : \forall R.\forall T^{-1}.B \sqcap \neg B\}$ don't have any acyclic inconsistent subset. In fact, a_1 and a_2 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.

Optimizations

As described earlier, for query answering we start by creating the summary Abox \mathcal{A}' of the original Abox in memory. The summary Abox is dramatically smaller than the original Abox, but we can reduce the size of the summary Abox built in memory much further by employing effective filtering techniques described in (A.Fokoue *et al.* 2006b). The basic idea is that we filter out role assertions that cannot be responsible for the detection of an inconsistency in the Abox, either because they cannot be used to propagate a concept assertion, or because they cannot be involved in the detection of an inconsistency due to a merger of Abox individuals. For the SHIN sub-language of DL, these are role assertions where the roles are not specified in any universal restriction or a maximum cardinality restriction in $\text{clos}(\mathcal{A})$. Note that the $\text{clos}(\mathcal{A})$ includes the negated query, because the queried concept is effectively a part of the knowledge base. This filtering step reduces the size of the knowledge base that is used as a starting point for query processing. As described in earlier section, we iteratively apply refinement to selectively increase the size of the knowledge base. At each step of refinement, the refined knowledge base is checked for consistency, to check if a conclusive inconsistency exists. Prior to each consistency check, we also apply filtering based on a static tableau algorithm described in (A.Fokoue *et al.* 2006a). This filtering step considers whether a role assertion can be used in the detection of an inconsistency based on the a conservative estimate of all concepts that can ever be added to a given individual's concept set. This conservative estimate of the concept set of an individual is obtained by the application of a modified static tableau as described in the paper, and the step further reduces the size of the knowledge base that is checked for consistency at each refinement step.

²Once again, acyclicity is defined w.r.t. to the undirected graph induced by role and differentFrom assertions

Concept Simplification

One of the key optimizations in our query answering algorithm, which helps simplify the summary even further, is to replace concepts (types) present in the labels of the summary individuals, that are found to be irrelevant for answering a particular query, by their more generic super-types. We refer to this process as *concept simplification*.

Concept simplification is a two-step process. In the first step, we expand the label of each summary individual by unfolding concepts present in them recursively. For example, if a concept A is present in the label of a summary individual x , and the axioms $\{A \sqsubseteq B, B \sqsubseteq C\} \in T_u$ (the unfoldable fragment of the TBox), then we set the label $\mathcal{L}(x) \leftarrow \{A, B, C\}$. In the second step, we examine concepts across all the expanded labels and replace any atomic concept(s) whose negation does not appear in the *closure* of the ABox (including the negation of the query) by \top . Note that this replacement of atomic concepts by \top is also performed on nested complex expressions where the atomic concepts appear. We describe the notion of ABox closure (previously defined in (A.Fokoue *et al.* 2006b)) and show how it is extended here.

Given an ABox \mathcal{A} , an RBox \mathcal{R} , and a TBox $\mathcal{T} = \mathcal{T}_u \cup \mathcal{T}_g$, where all the concepts appearing in \mathcal{T} and \mathcal{A} are in negation normal form (NNF), the closure of a concept expression C (also in NNF) is the smallest set X containing C closed under concept sub-expression, such that for a named concept A , (1) if $A \in X$ and $A \sqsubseteq D \in \mathcal{T}_u$, then $D \in X$; (2) if $\neg A \in X$ and $\neg A \sqsubseteq D \in \mathcal{T}_u$, then $D \in X$, and (3) if $\forall P.C \in X$ and there is a role R with $R \sqsubseteq^* P$ and $\text{Trans}(R)$, then $\forall R.C \in X$. We define the closure of \mathcal{A} w.r.t. \mathcal{T} and \mathcal{R} , denoted $\text{clos}(\mathcal{A}, \mathcal{T}, \mathcal{R})$, as $\bigcup_{a:C \in \mathcal{A}} \text{clos}(C, \mathcal{T}, \mathcal{R}) \cup \bigcup_{C \sqsubseteq D \in \mathcal{T}_g} \text{clos}(\text{NNF}(\neg C \sqcup D), \mathcal{T}, \mathcal{R})$.

Given a particular query Q under consideration, we extend the closure of the ABox to include $\neg Q$. Let us call the resultant concept set $\text{clos}(\mathcal{A}, Q)$, i.e., $\text{clos}(\mathcal{A}, Q) \leftarrow \text{clos}(\mathcal{A}, \mathcal{T}, \mathcal{R}) \cup \{\neg Q\}$. In the second step, we check whether a atomic concept in the expanded summary individuals' labels (i.e., expanded after unfolding in the first step) and its negation are present in the set $\text{clos}(\mathcal{A}, Q)$. The rationale here is that query answering is done by a standard proof-by-refutation technique, which involves building an inconsistent ABox, and thus both, a concept and its negation, have to be present in the extended closure $\text{clos}(\mathcal{A}, Q)$ for them to cause a contradiction. The absence of either one (concept or its negation) implies that the concept is irrelevant from this particular query answering point of view, and thus can be safely removed from the summary ABox without affecting the soundness or completeness of the algorithm.

Computing Obvious Answers and Non-answers

We avoid testing individuals in the summary ABox that are obvious answers or non-answers of the queried concept Q . An obvious non-answer of Q is an individual t in the summary \mathcal{A}' already explicitly asserted to be an instance of $\neg Q$ (i.e. $t : \neg Q \in \mathcal{A}'$). An obvious answer of Q is an individual in the summary \mathcal{A}' such that the intersection of all its explicit concepts is subsumed by Q . An obvious answer al-

ways has a role-free justification. Without nominals, finding obvious answer is a pure TBox computation. All individuals in the original ABox \mathcal{A} mapped to an obvious answer are instances of Q because our summarization process does not introduce extraneous concept assertions to the summary (i.e. if $t : C \in \mathcal{A}'$, then, for all $a \in \mathcal{A}$ such that $\mathbf{f}(a) = t$, $a : C \in \mathcal{A}$). Likewise, all individuals in \mathcal{A} mapped to an obvious non-answer cannot be inferred to be instances of Q .

Treatment of Leaves in a Justification

The notion of precise justifications of a summary used in practice is less restrictive than Definition 2. A precise subset L of a summary \mathcal{A}' is such that all its non-leaf individuals and its tested leaves are precise w.r.t. L . Non-tested leaf individuals are not required to be precise w.r.t. L .

All the results presented in this paper remain valid with this less restrictive definition. In fact, if $\text{im}L$ is the image of a precise, in the sense of the relaxed definition, subset L of a summary and H is the set of individuals in $\text{im}L$ that make leaf individuals imprecise (w.r.t. Definition 2), then, because our summarization does not produce extraneous concept assertions, elements of H are all isolated nodes in $\text{im}L$. Furthermore, since our summarization does not produce extraneous role assertions (i.e. if $R(s, t) \in \mathcal{A}'$, then there is at least one pair of individuals (a, b) in \mathcal{A} such that $R(a, b) \in \mathcal{A}$, $\mathbf{f}(a) = s$ and $\mathbf{f}(b) = t$), it follows that L is a precise summary of $\text{im}L - H$ in the sense of Definition 2.

Partitioning

When the summary ABox contains islands, i.e., disconnected sub-graphs, we partition the summary and consider each isolated sub-graph separately. Since our system currently works for the description logic *SHIN*, which does not contain nominals, it is safe to partition the ABox without affecting soundness and completeness of the query answering algorithm. Note that individuals in disconnected partitions can only interact via axioms in the TBox by using nominals.

We found that the partitioning strategy works well in a lot of realistic large ontologies where the class hierarchy is spread out, typically observed when dealing with separate domains or specializing in numerous areas. In such cases, there exist a lot of disconnections between sub-ABoxes that are tied into separate class hierarchies.

Partitioning also presents a great opportunity for parallelization since the query answering algorithm can be executed on each separate partition simultaneously with the results being combined at the end.

Individual Selection Strategy

Using our technique of query answering, one can either take the approach of adding a negated query to a single individual in the summary ABox, and test it for consistency, or add it to all individuals in the summary ABox and test it for consistency. The advantage of adding it to all individuals in the summary ABox is that we can try to find a large number of justifications that are present in the summary ABox, and target all individuals which are part of this justification for

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 3: Dataset Statistics

further refinement. Because refinement is I/O intensive, it is useful to refine as many individuals as possible in a single pass.

Justification Patterns

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 (or even a straightforward pattern matcher) 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 this 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.

Evaluation

Our algorithms are implemented in a system called SHER, which includes additional optimizations (blind anon 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 3. Table 4 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.

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 4: Runtimes (sec)

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; C.Patel; E.Schonberg; and K.Srinivas. 2006a. Using abstract evaluation in abox reasoning. *Proceedings of the Second Int. Workshop in Scalable Semantic Web Knowledge Base Systems*.
- A.Fokoue; A.Kershenbaum; L.Ma; E.Schonberg; and K.Srinivas. 2006b. 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*.
- blind anon. 2007. Technical report: Scalable semantic retrieval through summarization and refinement.
- 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 Hender, 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*.

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)*.

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.

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial Intelligence* 32:57–95.

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