The Summary Abox: Cutting Ontologies Down to Size

Achille Fokoue[1], Aaron Kershenbaum[1], Li Ma[2],
Edith Schonberg[1], and Kavitha Srinivas[1]

[1] IBM Watson Research Center,P.O.Box 704, Yorktown Heights, NY 10598, USA
{achille, aaronk, ediths, ksrinivs}@us.ibm.com
[2] IBM China Research Lab, Beijing 100094, China
malli@cn.ibm.com

Abstract. Reasoning on OWL ontologies is known to be intractable in the worst-case, which is a serious problem because in practice, most OWL ontologies have large Aboxes, i.e., numerous assertions about individuals and their relations. We propose a technique that uses a summary of the ontology (*summary Abox*) to reduce reasoning to a small subset of the original Abox, and prove that our techniques are sound and complete. We demonstrate the scalability of this technique for consistency detection in 4 ontologies, the largest of which has 6.5 million role assertions.

1 Introduction

Description Logic (DL) provides the theoretical foundation for semantic web ontologies (OWL). A DL ontology can be divided conceptually into three components: the Tbox, the Rbox and the Abox. The Tbox contains assertions about concepts such as subsumption ($Man \sqsubseteq Person$) and equivalence ($Man \equiv MaleHuman$). The Rbox contains assertions about roles and role hierarchies ($hasSon \sqsubseteq hasChild$). The Abox contains role assertions between individuals ($hasChild(John, Mary)$) and membership assertions ($John : Man$).

All common reasoning tasks in expressive DL ontologies, such as query answering [1], reduce to consistency detection. As an example, a standard approach to testing if *John* is a member of the concept *Man* requires testing if the addition of the assertion ($John : \neg Man$) makes the Abox inconsistent. A challenge is that consistency detection in expressive DL is well known to be intractable in the worst-case [2]. Given that the size of an Abox may be in the order of millions of assertions, this complexity poses a serious problem for the practical use of DL ontologies, which often reside in frequently updated transactional databases. Although highly optimized DL tableau algorithms exist, they cannot be easily adapted to Aboxes in secondary storage, especially for frequently changing Aboxes. One approach that has been applied to reasoning on Aboxes in secondary storage is to convert DL to disjunctive datalog, and use deductive databases to reason over the Abox [3].

We propose an alternative technique that operates on Aboxes stored in traditional relational databases. Our technique exploits a key observation about

I. Cruz et al. (Eds.): ISWC 2006, LNCS 4273, pp. 343–356, 2006.

real world Aboxes, namely, similar individuals are related to other individuals in similar ways (e.g. fathers and mothers are related to their children by the *hasChild* role). Specifically, our technique builds a summary Abox $\mathcal{A}'$ of the original Abox $\mathcal{A}$, by aggregating similar individuals and assertions. The advantages of our summary Abox $\mathcal{A}'$ are: (a) $\mathcal{A}'$ is dramatically smaller than $\mathcal{A}$; (b) Reasoning on $\mathcal{A}'$ isolates a small relevant portion of $\mathcal{A}$ needed to obtain the correct answer; (c) $\mathcal{A}'$ can be computed efficiently using straightforward relational database queries; (d) $\mathcal{A}'$ can be maintained as changes occur to $\mathcal{A}$, and is thus resilient to change; (e) $\mathcal{A}'$ only needs to be computed once, and can be reused for answering subsequent queries.

To isolate relevant portions of $\mathcal{A}$ for a specific reasoning task, we introduce efficient filtering techniques that operate on $\mathcal{A}'$. In this paper, we demonstrate the utility of such filtering techniques for the task of Abox consistency detection, although the approach can be generalized to query answering. Our filtering techniques are based on the conservative assumption that any individual in the Abox may be inferred to be a member of any concept in the closure of the Abox. Informally, the closure is the set of concepts used in the Abox, and their subexpressions. (To generalize this to query answering, the closure would include the negated concept in the query.) The effect of filtering is to produce multiple partitions in the summary Abox. In practice, most partitions consist of a single individual, which can be checked with a concept satisfiability test. For partitions of $\mathcal{A}'$ with multiple individuals, if the partition is consistent, then the image in $\mathcal{A}$ that corresponds to the partition is also consistent. If a partition is inconsistent, the inconsistency could arise from either the summarization technique, or a real inconsistency in the original Abox. In this case, we perform a consistency check on the image in $\mathcal{A}$ of the inconsistent partition in $\mathcal{A}'$.

Our techniques proved very effective on the 4 large Aboxes that we studied: a vast majority of partitions (95%) had just one individual. Only one of the 4 ontologies we studied required us to check the image of the partition in the original Abox. Even in this case, our consistency check was performed in 6.3 s on 4045 individuals and 2942 role assertions instead of the 1,106,858 individuals and 6,494,950 role assertions in the entire Abox.

Our key contributions in this paper are as follows: (a) We present a technique to summarize an Abox in secondary storage into a dramatically smaller $\mathcal{A}'$. (b) We describe the use of filtering techniques to construct a reduced version of $\mathcal{A}'$. This filtering produces many partitions, which are then exploited in scaling the consistency check. The filtering techniques we describe works for SHIN Aboxes (SHIN is a DL language that is described in the Background). (c) We show the application of these techniques to 4 ontologies, where we show dramatic reductions in space and time requirements for consistency checking.

1.1 Background

The techniques we apply in this paper assume ontologies of SHIN expressiveness. In this section, we briefly introduce the semantics of SHIN, which is equivalent to OWL-DL (http://www.w3.org/2001/sw/WebOnt) minus nominals and datatype

reasoning, as shown in Table 1 (We assume the reader is familiar with Description Logics). In the definition of the semantics of SHIN, $\mathcal{I}= (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ refers to an interpretation where $\Delta^{\mathcal{I}}$ is a non-empty set (the domain of the interpretation), and $.^{\mathcal{I}}$, the interpretation function, maps every atomic concept C to a set $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, every atomic role R to a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} X \Delta^{\mathcal{I}}$, and every individual a to $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Trans($R$) in the table refers to a transitive role R.

Table 1. SHIN Description Logic

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

(a) Constructors

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

(b) Axioms

An RBox $\mathcal{R}$ is a finite set of transitivity axioms of the form 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 $a \dot{\neq} b$.

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 1(b)). 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 standard technique for checking the consistency of a SHIN Abox is to use a tableau algorithm [4], which executes a set of non-deterministic expansion rules to satisfy constraints in $\mathcal{A}$ until either no rule is applicable or an obvious inconsistency (clash) is detected.

2 Summary Abox

Intuitively, the Abox contains many redundant assertions from the point of view of consistency checking that can be collapsed to create a reduced *summary Abox*. The summary Abox captures this redundancy by collapsing across individuals that are members of the same concept sets as shown in Figures 1 and 2 below. As shown in Figure 2, a single node a represents $a1$ and $a2$ because they are both members of A, and they are not explicitly asserted to be different from each other (similarly for b and d). Any explicit assertions that two individuals are different from each other ($c1$ and $c2$) are maintained in the summary Abox. Reasoning over such a summary corresponds to reasoning over the original Abox, as shown formally below.

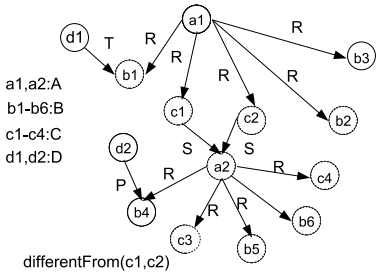

Fig. 1. Original Abox

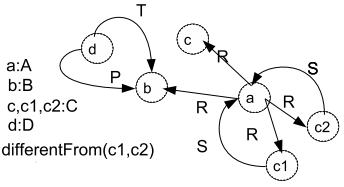

Fig. 2. Canonical Summary Abox

Definition 1. A summary Abox is an Abox $\mathcal{A}'$ that is generated from any SHIN Abox $\mathcal{A}$ using 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 $\mathbf{f}(a) \neq \mathbf{f}(b) \in \mathcal{A}'$

Theorem 2. 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 of Theorem 2 does not hold.

Proof. Let us assume that $\mathcal{A}'$ is consistent w.r.t. $\mathcal{T}$ and $\mathcal{R}$. Therefore there is a model $\mathcal{I}' = (\Delta^{\mathcal{I}'}, .^{\mathcal{I}'})$ of $\mathcal{A}'$ w.r.t. $\mathcal{T}$ and $\mathcal{R}$. A model of $\mathcal{A}$ can easily be built from $\mathcal{I}'$ by interpreting an individual a in $\mathcal{A}$ in the same way as $\mathbf{f}(a)$ is interpreted by $\mathcal{I}'$. Formally, let $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ be the interpretation of the $\mathcal{A}$ w.r.t. $\mathcal{T}$ and $\mathcal{R}$ defined as follows: $\Delta^{\mathcal{I}} = \Delta^{\mathcal{I}'}$; for a concept $C \in \mathcal{T}$, $C^{\mathcal{I}} = C^{\mathcal{I}'}$; for a role R in $\mathcal{R}$, $R^{\mathcal{I}} = R^{\mathcal{I}'}$; for an individual a in $\mathcal{A}$, $a^{\mathcal{I}} = \mathbf{f}(a)^{\mathcal{I}'}$. $\mathcal{I}$ is a model of $\mathcal{A}$ w.r.t. $\mathcal{T}$ and $\mathcal{R}$ as a direct consequence of the fact that $\mathcal{I}'$ is a model of $\mathcal{A}'$ and $\mathcal{A}'$ satisfies the 3 conditions stated in definition 1 (See [5] for more details). □

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 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 $R(a, b) \in \mathcal{A}$.
(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).

If a summary Abox $\mathcal{A}'$ is not consistent, either there is a real inconsistency in $\mathcal{A}$ or the process of summarization caused an artificial inconsistency. In section 3.4, we explain how we apply filtering to partition the summary, to provide a scalable consistency check of the original Abox even when the summary is

inconsistent. Note that the summary Abox can be computed efficiently from a relational database. Furthermore, it only needs to be computed once, and can be maintained incrementally with changes to the Abox.

3 Abox Filtering

We perform filtering on the canonical summary Abox described in Section 2, but for the purpose of exposition, we describe filtering techniques on the original Abox first. Our filtering technique assumes that the Tbox $\mathcal{T}$ is transformed, through splitting and absorption [6], into two disjoint sets $\mathcal{T}_u$ and $\mathcal{T}_g$ such that $\mathcal{T}_u$, the unfoldable part of $\mathcal{T}$, only contains axioms of the form $A \sqsubseteq D$ and $\neg A \sqsubseteq D$, where A is an atomic concept. $\mathcal{T}_g$ contains general concept inclusions of the form $C \sqsubseteq D$ that could not be absorbed in $\mathcal{T}_u$ (where C is a complex concept).

We assume that for an Abox $\mathcal{A}$, an Rbox $\mathcal{R}$, and a Tbox $\mathcal{T} = \mathcal{T}_u \cup \mathcal{T}_g$, all concepts appearing in $\mathcal{T}$ and $\mathcal{A}$ are in the negation normal form (NNF). For a concept expression C in NNF, $clos(C, \mathcal{T}, \mathcal{R})$ is the smallest set X containing C, closed under concept sub-expression, such that, for an atomic 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 Trans(R), then $\forall R.C \in X$. Formally, we define the closure of $\mathcal{A}$ w.r.t. $\mathcal{T}$ and $\mathcal{R}$, denoted $clos(\mathcal{A}, \mathcal{T}, \mathcal{R})$, as $\bigcup_{a:C \in \mathcal{A}} clos(C, \mathcal{T}, \mathcal{R}) \cup \bigcup_{C \sqsubseteq D \in \mathcal{T}_g} clos(NNF(\neg C) \sqcup D, \mathcal{T}, \mathcal{R})$. When there is no ambiguity, we use $clos(\mathcal{A})$ instead of $clos(\mathcal{A}, \mathcal{T}, \mathcal{R})$.

3.1 Motivation

To provide an intuition for our criteria for filtering role assertions, we first informally describe a subset of tableau expansion rules defined in [4]. We assume that a and b are named individuals in $\mathcal{A}$, x is an unnamed individual, C is a concept in $clos(\mathcal{A})$, and R is a role. Named individuals are in $\mathcal{A}$ before applying any expansion rules, while unnamed individuals are introduced as a result of expansion rules. An individual b is defined to be an R-neighbor of a iff there is an assertion $Q(a, b)$ or $Q^-(b, a)$ in $\mathcal{A}$ where $Q \sqsubseteq^* R$.

$\forall$-*rule:* $a : (\forall R.C) \in \mathcal{A}$, b is an R-neighbor of a, and $b : C \notin \mathcal{A} \Rightarrow$ add $b : C$ to $\mathcal{A}$.

$\leq$-*rule:* $a : (\leq nR) \in \mathcal{A}$ and, for $1 \leq i \leq m, m > n$, b_i is an R-neighbor of a, and for two of these b_k and b_j the assertion $b_k \neq b_j$ is not in $\mathcal{A} \Rightarrow$

(i) Merge (identify) b_j with b_k (the precise rules to determine which of b_j or b_k is selected is detailed in [4].)

(ii) add the assertions in $\mathcal{L}(b_j)$ to $\mathcal{L}(b_k)$

(iii) for every role assertion with b_j, replace b_j with b_k. In effect, this step adds new role assertions to the $\mathcal{A}$.

$\exists$-*rule:* $a : (\exists R.C) \in \mathcal{A}$ and for R-neighbors b of a, $b : C \notin \mathcal{A} \Rightarrow$ add $R(a, x)$ and $x : C$ to $\mathcal{A}$.

$\geq$-*rule:* $a : (\geq nR) \in \mathcal{A}$ and a does not have n distinct R-neighbors $\Rightarrow$ add $R(a, x_i)$ to $\mathcal{A}, 1 \leq i \leq n$ where the x_i are new distinct unnamed individuals.

$\forall_+$-*rule:* $a : \forall R.C \in \mathcal{A}$, $P \sqsubseteq^* R$ is transitive, b is a P-neighbor of a, and $b : (\forall P.C) \notin \mathcal{A} \Rightarrow$ add $b : (\forall P.C)$ to $\mathcal{A}$.

We make the key observation that role assertions of the form $R(a, b)$ may affect the outcome of an execution of the tableau algorithm only if one of the following two conditions holds:

1. They can be used to trigger the application of tableau rules that alter the original Abox. As an example of such alteration, a role assertion can be used to add new membership assertions about named individuals (e.g., a new concept C can be propagated to b's concept set through a role assertion $R(a, b)$ by the application of the $\forall$ rule on a if $a : (\forall R.C) \in \mathcal{A}$, and b is an R-neighbor of a).
2. They can be involved in clash detection due to a violation of a maximum cardinality restriction. As an example, if $a : (\leq nR)$ is in the Abox and b is one of $n + 1$ mutually distinct R-neighbors of a, then $R(a, b)$ is important for clash detection.

These conditions can *only* be brought about by either the application of the $\forall$, $\leq$ or $\forall_+$-rules or the presence of a maximum cardinality constraint. In contrast, the $\exists$-rule and $\geq$-rule do not use existing role assertions $R(a, b)$; instead, they result in the creation of new role assertions and new unnamed individuals for satisfying the $\exists R.C$ and $\geq nR$ constraints.

3.2 Criteria for Filtering Role Assertions

Our filtering criteria guarantee that the absence of a role assertion will not affect the outcome of any execution of the non-deterministic tableau algorithm. Our goal is to define criteria which are efficient to evaluate using simple queries against relational databases, while balancing the tradeoff between filtering precision and cost. For instance, by assuming any concept in the $clos(\mathcal{A})$ can reach the concept set of any individual in the Abox during any execution of the tableau algorithm, we avoid tableau operations which are expensive in relational databases. We will say that a role R is *part of a universal restriction* $\forall P.C$ iff $R \sqsubseteq^* P$. (Similarly for maximum cardinality restriction).

To filter a role assertion $R(a, b)$, it must satisfy either 1) or both 2) and 3):

1) Absence of universal and maximum cardinality restrictions: We make the simple observation that if a role R and its inverse R^- are not part of any universal or maximum cardinality restrictions, then $R(a, b)$ can never be used to alter the original Abox or detect a clash, so it can be ignored.

2) Absence of universal rules triggering: Even if R is part of a universal restriction, it may never trigger the application of the universal rules ($\forall_+, \forall$). We define the conditions under which we can guarantee that the universal rules will never be triggered as follows. If R (resp. R^-) is part of a universal restriction $\forall P.C$ in $clos(\mathcal{A})$, then $R(a, b)$ is *irrelevant with respect to* $\forall P.C$ if $b : C \in \mathcal{A}$ (resp. $a : C \in \mathcal{A}$) and R (resp. R^-) has no transitive superroles.

To satisfy the filtering condition, $R(a, b)$ must be irrelevant with respect to all universal restrictions $\forall P.C$ in $clos(\mathcal{A})$, where R or R^- is part of $\forall P.C$.

3) Absence of maximum cardinality restrictions triggering: For the $\leq$-rule to be triggered, there needs to be a violation of a maximum cardinality constraint $a :\leq nP$, where the individual a has more than n P-neighbors, which then causes a merger between individuals. We introduce a technique to conservatively estimate an upper bound on P-neighbors, such that at any step of any

possible execution of tableau algorithm, the number of P-neighbors of a is less than or equal to this upper bound. If R (resp. R^-) is part of a maximum cardinality restriction $\leq nP$ in $clos(\mathcal{A})$, then $R(a, b)$ is *irrelevant with respect to* $\leq nP$ if the upper bound on the number of P-neighbors of a (resp. b) is less than or equal to n. Note that this also guarantees that no clash can occur from the presence of the role assertion.

To satisfy the filtering condition, $R(a, b)$ must be irrelevant with respect to all maximum cardinality restrictions $\leq nP$ in $clos(\mathcal{A})$, where R or R^- is part of $\leq nP$.

Upper bound on the number of P-neighbors. Unfortunately, in expressive logics such as SHIN, computing an upper bound does not simply involve counting the number of explicit P-neighbors of a that are present in $\mathcal{A}$. Figure 3 shows examples of the three ways that an individual a can acquire a new P-neighbor during the execution of the tableau algorithm:

3(A) Individual a acquires a new P-neighbor x, where x is an unnamed individual, to satisfy $\exists P.C$ in a's concept set.

3(B) Individual a is merged with a named individual d and acquires a new P-neighbor in order to satisfy a maximum cardinality restriction $c :\leq nQ$.

3(C) Individual a is merged with an unnamed individual x and acquires a new P-neighbor c, where c is either a named or unnamed individual. This occurs in this example because of two conditions: (i) there is a role S^- that is *attracted* to P^- because a common super role Q is part of a maximum cardinality restriction and (ii) a role generator of the form $\exists T.B$ or $\geq mT$ is in the concept set of c, where $T \sqsubseteq^* P^-$.

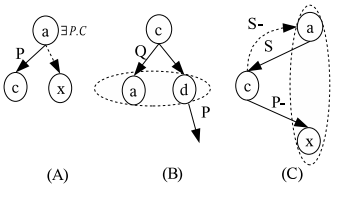

Fig. 3. Acquisition of P-neighbors

Accounting for P-neighbors acquired through situations like 3(B) and 3(C) is not obvious. Therefore we define sufficient conditions under which these situations cannot occur, so that an upper bound of $a's$ P-neighbors can be computed safely and efficiently. If any of these conditions are violated, then a merger of a may result in an increase of its number of P-neighbors, and hence we do not filter $R(a, b)$:

(C1) P is *safe* in $\mathcal{A}$.
Intuitively, the notion of safety ensures that a merger of a named individual a with an unnamed individual x that would increase the number of P-neighbors of a, as illustrated in 3(C), cannot occur. If P is safe, then either

condition (i) or (ii) in 3(C) must be false. More generally, for a given role P, we say that T belongs to the set $attractant(P)$ iff there is a role Q such that $P \sqsubseteq^* Q$, $T \sqsubseteq^* Q$, and $\leq nQ \in clos(\mathcal{A})$. A role P is *safe* if one of the two conditions hold: (a) $attractant(P) \subseteq \{P\}$ and $attractant(P^-) \subseteq \{P^-\}$ (b) For all subroles Q of P or P^- there are no Q-generators (i.e. $\geq mQ$ or $\exists Q.C$) in $clos(\mathcal{A})$.

(C2) For any role Q, if a is a Q-neighbor of some named individual c then there is no concept of the form $(\leq nQ)$ in $clos(\mathcal{A})$.

(C3) For any role S, if some named individual c is a S-neighbor of a and $\leq nS$ is in $clos(\mathcal{A})$, then S is safe in $\mathcal{A}$.

Conditions (C2) and (C3) ensure that a merger of a and a named individual as illustrated in 3(B) is impossible. (C2) by itself is not sufficient because, even if Q is not part of a maximum cardinality restriction, Q^- may have an attractant T^-, where $\exists T^- B$ is in the concept set of a. As described in 3(C), these conditions can cause a merger between c and an unnamed individual x, so that a becomes a T-neighbor of c. If T is part of a maximum cardinality restriction, a itself may become mergable. Condition (C3) prevents mergers between c and unnamed individuals that would make a a T-neighbor of c, thus preventing a from becoming mergable.

If a and P satisfy (C1), (C2) and (C3), an upper bound on the number of P-neighbors can be computed using the following formula:

$$|P(a)| + |Some(P, a)| + \sum_{\geq mP \in Min(P,a)} m$$

where before the application of any tableau rules, $|P(a)|$ denotes the number of P-neighbors of a, $Some(P, a) = \{\exists P.C \in clos(\mathcal{A}) \mid$ there is no P-neighbor d of a such that $d : C \in \mathcal{A}\}$, and $Min(P, a) = \{\geq mP \in clos(\mathcal{A}) \mid$ there are no individuals d_i such that, for $1 \leq i \leq m$, d_i is a P-neighbors of a, and if $j \neq k$, then $d_k \dot{\neq} d_j \in \mathcal{A}\}$

Intuitively, the upper bound is the sum of the explicit P-neighbors of a before the application of tableau rules, plus the maximum number of unnamed individuals that can be generated by the application of the $\exists$- and $\geq$- rules, excluding any existential or minimum cardinality restrictions that are already satisfied prior to the application of tableau rules.

3.3 Correctness of Filtering Criteria

Since some of the notions introduced in the previous section are defined in the context of the tableau algorithm, we first briefly present some important concepts related to this algorithm. As described in [4], the tableau algorithm operates on completion forest $F = (G, \mathcal{L}, \dot{\neq}, \dot{=})$ where G is graph; $\mathcal{L}$ is a mapping from a node x in F to a set of concepts, $\mathcal{L}(x)$, in $clos(\mathcal{A})$, and from an edge $< x, y >$ in F to a set of roles, $\mathcal{L}(< x, y >)$, in $\mathcal{R}$; $\dot{=}$ is an equivalence relation on nodes of G; and $\dot{\neq}$ is the binary relation *distinct from* on nodes of G. To check the consistency of $\mathcal{A}$, F is initialized as follows. There is a node a in G iff there is an individual a in $\mathcal{A}$. $< x, y >$ is an edge in G with $R \in \mathcal{L}(< x, y >)$ iff $R(x, y) \in \mathcal{A}$.

For x and y in G, $x \dot{\neq} y$ iff $x \dot{\neq} y \in \mathcal{A}$. A *root* node a is a node present in the initial forest, and *unnamed* nodes are created by $\exists$ and $\geq$ rules.

Next, we show that conditions (C1), (C2) and (C3) in Section 3.2 are sufficient to rule out mergers which can increase the number of P-neighbors as shown in Figure 3(B) and (C). Lemma 3 below is an important step towards this goal.

Lemma 3. Let P be a role that is safe in $\mathcal{A}$. At any step of any execution of the tableau algorithm on $\mathcal{A}$, the following holds: if there is an unnamed node x such that P or P^- is in the $\mathcal{L}(< parent(x), x >)$, then $|\mathcal{L}(< parent(x), x >)| = 1$, where $parent(x)$ denotes the parent node of x in the completion forest (Note that in a SHIN completion forest, unnamed nodes are always in a tree rooted at a root node).

Proof. Easily proven by induction on the iterations of the tableau algorithm. See [5] for more details. A direct consequence of this lemma is that a merger between a and an unnamed node cannot increase the number of P-neighbors of a if P is safe (See [5] for more details).

Now, we need to prove that if (C2) and (C3) are satisfied for a named individual a, a cannot be merged with another named individual. First, we formally define the notion of *mergeability with a named individual*.

Definition 4. A named individual a in $\mathcal{A}$ is *mergeable with named individuals* in $\mathcal{A}$ iff there is at least one execution of the tableau algorithm on $\mathcal{A}$ such that, at some step, the root node a is merged with another root node b. When there is no ambiguity, we simply say that a is mergeable.

Theorem 5: If a named individual a in $\mathcal{A}$ satisfies conditions (C2) and (C3), then a is *not mergeable with named individuals* in $\mathcal{A}$.

Proof Sketch. By induction on the iterations of the tableau algorithm using Lemma 3 [5]. □

Finally, the correctness of our filtering criteria relies on the following theorem:

Theorem 6. A role assertion $R(a, b)$ can safely be ignored in an Abox $\mathcal{A}$ if it is irrelevant with respect to universal restrictions and irrelevant with respect to maximum cardinality restrictions, as defined in section 3.2.

Proof. Let $R(a, b)$ be a role assertion irrelevant w.r.t. maximum cardinality and universal restrictions in an Abox $\mathcal{A}$. Let $\mathcal{A}'$ be the Abox defined as $\mathcal{A}' = \mathcal{A} - \{R(a, b), R^-(b, a)\}$. If $\mathcal{A}$ is consistent, $\mathcal{A}'$ is obviously consistent. We show that if $\mathcal{A}'$ is consistent, a model of $\mathcal{A}$ can be constructed by applying the tableau algorithm rules in a particular way. [1]

First, for a root node c in the completion forest F, the root node $\alpha(c)$ is defined as follows (Informally, $\alpha(c)$ corresponds to the node in which c has been directly or indirectly merged): if $\mathcal{L}(c) \neq \emptyset$ then $\alpha(c) = c$; otherwise, $\alpha(c) = d$, where d is the unique root node in F with $\mathcal{L}(d) \neq \emptyset$ and $d \doteq c$.

[1] A direct model-theoretic proof cannot easily be provided here, see [5] for details.

Since $\mathcal{A}'$ is consistent, we can apply the tableau expansion rules on $\mathcal{A}'$ without creating a clash in such a way that: (1) $\exists$-rule is never triggered to satisfy a constraint $\exists P.C \in \mathcal{L}(\alpha(a))$(resp. $\mathcal{L}(\alpha(b))$) where $\leq nP \in clos(\mathcal{A})$, R (resp. R^-) is part of $\leq nP$, and $b : C \in \mathcal{A}$ (resp. $a : C \in \mathcal{A}$); and (2) $\geq$-rule is never triggered to satisfy a constraint $\geq nP \in \mathcal{L}(\alpha(a))$ (resp. $\mathcal{L}(\alpha(b))$) where $\leq nP \in clos(\mathcal{A})$, R (resp. R^-) is part of $\leq nP$, and, in the Abox $\mathcal{A}$, b (resp. a) is one of n P-neighbors of a (resp. P-neighbors of b) explicitly asserted to be distinct. Such a rule application yields a clash-free completion forest F, and the only nodes on which expansion rules may be applicable are $\alpha(a)$ and $\alpha(b)$ (The only applicable rules are $\exists$-rule and $\geq$-rule).

Next, we modify F to create a completion forest F' by adding to F the edge $< \alpha(a), \alpha(b) >$ if it was not already in F, and by adding R to $\mathcal{L}(< \alpha(a), \alpha(b) >$), if it was not already there. We show that F' is complete (i.e. no rules are applicable) and clash-free. The fact that, in F', $R \in \mathcal{L}(< \alpha(a), \alpha(b) >)$ ensures that the $\exists$ and $\geq$ rules, which may have been applicable on $\alpha(a)$ or $\alpha(b)$ in F, are not applicable on $\alpha(a)$ and $\alpha(b)$ in F'. However, the same fact may now make the $\forall$, $\forall_+$, $\leq$, and $\leq_r$ rules applicable on $\alpha(a)$ or $\alpha(b)$ in F'. We show that this cannot be the case.

The definition of irrelevance w.r.t. universal restrictions given in section 3.2 obviously ensures that $\forall$ and $\forall_+$ rules are not applicable on $\alpha(a)$ or $\alpha(b)$ in F'. $\leq$, and $\leq_r$ rules are not applicable on $\alpha(a)$ or $\alpha(b)$ in F' as a direct consequence of the following claim: **Claim:** if $R(a, b)$ is irrelevant w.r.t $\leq nP \in clos(\mathcal{A})$ and R (resp. R^-) is part $\leq nP$, then the number of P-neighbors of a (resp. P-neighbors of b) in F is less than or equal to n. Furthermore, if it is equal to n, then, in F, $\alpha(b)$ is a P-neighbor of $\alpha(a)$ (resp. $\alpha(a)$ is a P-neighbor of $\alpha(b)$).

The proof of this claim is a direct consequence of Lemma 3, Theorem 5 and the fact that the upper-bound (defined in section 3.2) of P-neighbors of a is less than or equal to n (See [5] for more details).

The addition of $R \in \mathcal{L}(< \alpha(a), \alpha(b) >)$ to F cannot create a clash of the form $\{C, \neg C\}$ in F', and the previous claim implies that a clash in F' due to a violation of a maximum cardinality constraint on $\alpha(a)$ or $\alpha(b)$ is not possible. Thus, F' is a complete clash-free completion forest such that $R \in \mathcal{L}(< \alpha(a), \alpha(b) >)$. Therefore, a tableau for $\mathcal{A}$ can be built from F' as in [4], so $\mathcal{A}$ has a model. □

3.4 Summary Abox Filtering

We apply the filtering criteria described in Section 3.2 to a canonical summary Abox $\mathcal{A}'$. For correctness with respect to cardinality restrictions, we need to augment the canonical summary Abox with role assertion statistics, since role assertions are merged by the summary Abox transformation. For each role R that is part of a cardinality restriction, we associated with R the maximum number of R-neighbors that any individual a has in $\mathcal{A}$. With this augmentation, it is clear that the proofs in Section 3.3 apply to the canonical summary Abox.

Typically, filtering $\mathcal{A}'$ creates distinct partitions, and we apply the tableau algorithm to each partition separately. If all of the partitions are consistent, then we are done. Otherwise, we need to check $\mathcal{A}$. However, even when $\mathcal{A}'$ itself

is inconsistent, some of its partitions may be consistent, and we only have to check portions of $\mathcal{A}$ which correspond to the filtered inconsistent partitions of $\mathcal{A}'$. Thus, partitioning a summary Abox is an effective way of isolating a potential inconsistency in $\mathcal{A}$. Furthermore, filtering $\mathcal{A}'$ is very efficient since $\mathcal{A}'$ is relatively small. For partitions consisting of a single individual, checking consistency is just checking concept satisfiability.

More precisely, let ${\mathcal{A}'}_p$ be a partition of individuals and assertions in $\mathcal{A}'$. The *image* of ${\mathcal{A}'}_p$ in $\mathcal{A}$ is defined to be the maximum subset of the individuals and assertions in $\mathcal{A}$ which map to ${\mathcal{A}'}_p$ via the summary Abox function **f**. If a role assertion $R(a, b)$ is irrelevant in $\mathcal{A}'$, then all role assertions in its image in $\mathcal{A}$ are also irrelevant. By theorem 2, if a partition ${\mathcal{A}'}_p$ is consistent, then its entire image in $\mathcal{A}$ can be ignored. Finally, retrieving the image in $\mathcal{A}$ of an inconsistent partition ${\mathcal{A}'}_p$ is a simple database operation.

For example, suppose we filter all R-role assertions from the summary Abox $\mathcal{A}'$ in Figure 2. The resulting summary Abox shown in Figure 4 consists of three partitions: X, Y, and Z. We run the consistency check on each partition. For singleton partition Z, we just need to check concept satisfiability. Assuming only partition X is inconsistent, we need to check only the consistency of its image in $\mathcal{A}$, shown in Figure 5, which is $d1, d2 : D; b1 - b6 : B; T(d1, b1)$ and $P(d2, b4)$. Checking isolated individuals $b2, b3, b5$ and $b6$, involves just concept satisfiability.

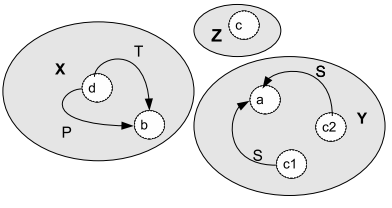

Fig. 4. Filtered summary Abox

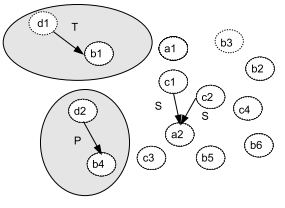

Fig. 5. Filtered Abox

4 Computational Experience

We tested our approach on the four ontologies shown in Table 2(a). Their expressiveness is given in the first column (Exp) of Table 2(a) . The number of concepts (C) and roles (R) reported in the table reflect concepts and roles actually used in the Abox. In the tables, R.A. stands for role assertions, and I for individuals. In all the experiments reported here, the Aboxes were stored in a relational database on a 64 bit AMD 997 Mhz dual processor 8G RAM machine. We tested our program as a client to the database server both on a 32 bit single 1.8 Ghz processor 1.5 G RAM machine, and on the 64 bit machine described above. Running times reported in the tables are in seconds on the 64 bit machine. On the 32 bit machine the times were 2 times slower, but the program ran on both machines with minimal space requirements (512 M heap).

The Biopax ontology contains biological pathway data for 11 organisms publicly available from Biocyc (http://biocyc.org). LUBM [7] is a benchmark ontology

that was scaled to different numbers of universities (5-30) in our experiments. We used an OWL-DL version of LUBM [8], but with nominals removed. The NIMD ontology expresses relationships between persons, places and events (http://ksl.stanford.edu/projects/NIMD/Kani-dl-v1.owl). Its Abox was generated from text analysis of unstructured documents [9]. The semantic traceability (ST) ontology specifies the relationships among software artifacts. Its Abox was generated from a program that extracted relationships between software artifacts of a middleware application. The sizes of the last 4 Aboxes shown in Table 2(a) are beyond the capabilities of in-memory reasoners such as Pellet and KAON2, when tested on the 64-bit machine with a 4G heap size.

Table 2. ABoxes and Summary Aboxes prior to filtering

Ontology	Exp	C	R	I	R.A.
Biopax	ALCHF	31	40	261,149	582,655
LUBM-1	SHIN	91	27	42,585	214,177
LUBM-5	SHIN	91	27	179,871	927,854
LUBM-10	SHIN	91	27	351,422	1,816,153
LUBM-30	SHIN	91	27	1,106,858	6,494,950
NIMD	SHIF	19	27	1,278,540	1,999,787
ST	SHI	16	11	874,319	3,595,132

(a) Experimental Aboxes

Ontology	C	R	I	R.A	Time
Biopax	31	40	81	583	46
LUBM-1	91	27	410	16,233	12
LUBM-5	91	27	598	35,375	60
LUBM-10	91	27	673	49,176	128
LUBM-30	91	27	765	79,845	485
NIMD	19	27	19	55	77
ST	16	11	21	183	197

(b) Summaries

Table 2(b) shows the size of the corresponding summary Aboxes prior to any filtering, and the time to compute the summaries. As noted in earlier sections, the summary Abox can be computed once, and maintained with changes to the Abox.

Table 3(a) shows the effectiveness of filtering the summary ABox for the consistency detection test, and the time to perform filtering. Note that the filtering step is dynamic, i.e., it must be computed on the summary box for each incoming query. The filtering step can create partitions. In Table 3(a), the first number in the first column (Sin.+Mult) indicates the number of partitions with single individuals, and the second number indicates the number of partitions with multiple individuals. The rest of the columns show the size of the Abox that is left after removing all partitions with single individuals.

Table 3(b) shows the size of the Abox on which we had to perform the consistency check. All times for the consistency check were measured using the Pellet OWL reasoner. For those Aboxes where the filtered summary Abox in Table 3(a) was consistent, the size of the ABox was simply that in Table 3(a) . For some ontologies, however (e.g., all LUBM ontologies marked with an asterisk), the filtered Abox was inconsistent because of our summarization techniques. For these ontologies, we had to retrieve the image of the inconsistent partition from the original Abox. In these cases, the size shown in Table 3(b) is the image of the partition in the original Abox. Time for consistency check is provided in seconds. This includes the time for the concept satisfiability check for partitions with single individuals, the time for the consistency check on the filtered summary, and

Table 3. Filtering and consistency check

Ontology	Sin.+Mult.	C	R	I	R.A.	Time
Biopax	42+1	13	1	38	98	1.6
LUBM-1	130+2	28	5	280	284	1.4
LUBM-5	172+2	28	5	426	444	2.1
LUBM-10	199+2	28	5	474	492	2.5
LUBM-30	220+2	28	5	545	574	2.8
NIMD	17+1	2	1	2	1	0.6
ST	3+1	15	2	18	50	0.3

(a) Summary Aboxes after filtering

Ontology	I	R.A.	Time	Consistent
Biopax	38	98	0.7	Yes
LUBM-1*	140	102	1	Yes
LUBM-5*	644	466	1.5	Yes
LUBM-10*	1283	938	2	Yes
LUBM-30*	4045	2942	3.5	Yes
NIMD	2	1	0.2	Yes
ST	-	-	0.1	No

(b) Sizes for consistency check

the time for retrieving and checking the image of the inconsistent partition on the original Abox. As shown in Table 3(b), ST was an inconsistent ontology, but we determined this purely based on a concept satisfiability check for partitions with single individuals. We also deliberately injected an inconsistency for one of the Biopax databases (agrocyc), to check if we could detect an inconsistency that could not simply be detected by a concept satisfiability check. We were able to detect that the Abox was inconsistent using our algorithm.

5 Related Work

There are many highly optimized reasoners such as Pellet [10], Racer [11], InstanceStore [12], and Kaon2 [3] designed for consistency checking, but only InstanceStore and Kaon2 can be extended to Aboxes in secondary storage [2]. Kaon2 applies to deductive databases, whereas our techniques work with relational databases. InstanceStore is limited to role-free Aboxes. In theory, Instance Store can handle Aboxes with role assertions through a technique called precompletion [13], but this may not be practical for large Aboxes stored in databases because it could result in an exponential number of Aboxes. Our approach can be compared with optimization techniques such as model caching and Abox contraction [14], and partitioning techniques [15], but again, it is unclear how such techniques can be applied to large Aboxes in databases.

6 Conclusions

We have demonstrated a technique to scale consistency detection to large Aboxes in secondary storage by extracting a small representative Abox. Further, we have shown that, in practice, this technique works efficiently on four large ontologies. Our plan is to extend this approach to apply more accurate analysis techniques, extend its applicability to more expressive languages, and to optimize these techniques for efficient query processing.

[2] RacerPro version 1.9.0 does not provide that capability, but its user guide indicates that it will be available in a future version.

References

1. Horrocks, I., Tessaris, S.: Querying the semantic web: a formal approach. In Horrocks, I., Hendler, J., eds.: Proc. of the 1st Int. Semantic Web Conf. (ISWC 2002). Number 2342 in Lecture Notes in Computer Science, Springer-Verlag (2002) 177–191
2. Donini, F.: Complexity of reasoning. In Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: Description Logic Handbook. Cambridge University Press (2002) 101–141
3. U.Hustadt, Motik, B., Sattler, U.: Reducing shiq description logic to disjunctive datalog programs. (Proc. of 9th Intl. Conf. on Knowledge Representation and Reasoning (KR2004)) 152–162
4. Horrocks, I., Sattler, U., Tobies, S.: Reasoning with individuals for the description logic SHIQ*. Proc. of 17th Int.Conf. on Automated Deduction (2000) 482–496
5. Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K.: Scalable reasoning: Cutting ontologies down to size. In: http://www.research.ibm.com/iaa/techReport.pdf. (2006)
6. Horrocks, I., Tobies, S.: Reasoning with axioms: Theory and practice. In: KR. (2000) 285–296
7. Guo, Y., Pan, Z., Heflin, J.: An evaluation of knowledge base systems for large owl datasets. Third International Semantic Web Conference (2004) 274–288
8. Ma, L., Yang, Y., Qiu, Z., Xie, G., Pan, Y.: Towards a complete owl ontology benchmark. In: Proc. of the third European Semantic Web Conf.(ESWC 2006). (2006) 124–139
9. Welty, C., Murdock, J.W.: Towards knowledge acquisition from information extraction. In: Proc. of the fifth International Semantic Web Conf.(ISWC 2006). (2006)
10. Sirin, E., Parsia, B.: Pellet: An owl dl reasoner. In: Description Logics. (2004)
11. Haarslev, V., Moller, R.: Racer system description. Conf. on Automated Reasoning (IJCAR 2001) (2001) 701–705
12. Bechhofer, S., Horrocks, I., Turi, D.: The owl instance store: System description. Proc. of 20th Int.Conf. on Automated Deduction (2005) 177–181
13. Tessaris, S., Horrocks, I.: Abox satisfiability reduced to terminological reasoning in expressive description logics. In: LPAR. (2002) 435–449
14. Haarslev, V., Moller, R.: An empirical evaluation of optimization strategies for abox reasoning in expressive description logics. Proc. of the International Workshop on Description Logics (1999) 115–199
15. Grau, B.C., Parsia, B., Sirin, E., Kalyanpur, A.: Automatic partitioning of owl ontologies using e-connections. In: Description Logics. (2005)