

Scalable Cleanup of Information Extraction Data Using Ontologies

Julian Dolby¹, James Fan¹, Achille Fokoue¹, Aditya Kalyanpur¹, Aaron
Kershenbaum¹, Li Ma², William Murdock¹, Kavitha Srinivas¹, and
Christopher Welty¹

¹ IBM Watson Research Center, P.O. Box 704, Yorktown Heights, NY 10598, USA
fanj, dolby, achille, adityakal, aaronk, murdockj, ksrinivs,
welty@us.ibm.com

² IBM China Research Lab, Beijing 100094, China
malli@cn.ibm.com

Abstract. The approach of using ontology reasoning to cleanse the output of information extraction tools was first articulated in SemantiClean. A limiting factor in applying this approach has been that ontology reasoning to find inconsistencies does not scale to the size of data produced by information extraction tools. In this paper, we describe techniques to scale inconsistency detection, and illustrate the use of our techniques to produce a consistent subset of a knowledge base with several thousand inconsistencies.

1 Introduction

The original vision of the semantic web was concerned with publishing the semantics of, and inter-connecting, the back-end databases that generate the vast majority of HTML content. A common misunderstanding about the semantic web is that the vision somehow hinges on manual markup of natural language text on web pages with semantic labels the way the original web's HTML markup was done.

While this is a misunderstanding and not really a valid criticism of the semantic web vision, the vast majority of knowledge on the web, and in organizations, is in natural language text. Exploiting this knowledge in automated ways is an important scientific and economic problem, and should not be ignored, however it is not always clear whether semantic web technology can really make any difference.

In previous work, we have reported on SemantiClean [1], a system to clean up natural language processing results using an OWL-DL reasoner that has been shown in experiments to improve the precision of relation analysis. The main shortcomings of the SemantiClean work was scalability. In this paper, we report on initial experiments to use a Scalable Highly Expressive Reasoner (SHER) [2] [3], to bring the SemantiClean approach up to the scale of current information extraction technology and to provide explanations for removed assertions.

The paper is organized as follows. After a brief background section, Section 3 presents our approach for scalable cleanup. Section 4 discusses presentation of explanations for removed assertions. The results of our experimental evaluation is presented in Section 5. In Section 6, we describe more advanced cleanup strategies. Section 7 and 8 provide the related work and conclusions.

2 Background

2.1 SemantiClean

The most problematic kind of extraction produced by natural language components we have experienced is relation extraction - the identification of relationships and their arguments in natural language text. A common type of error we see in extracted relations is the violation of simple domain and range constraints. For example, in the following sentence:

... the decision in September 1991 to withdraw tactical *nuclear bombs*,
missiles and torpedoes from US Navy ships ...

our analytics extract an ownership relation in the italicized text between nuclear (annotated as a weapon), and bombs (also a weapon), which maps to a `ownerOf` relation in the ontology. The `ownerOf` relation has a restriction limiting the domain to `Person` or `Organization` or `GPE` and a disjointness constraint between each of these and `Weapon`.

The SemantiClean approach is a simple one. We start with an ontology expressed in OWL-DL that must include negation (i.e. it must be possible to generate a contradiction in the Abox). Relations are extracted from text and stored as an RDF model that instantiates the ontology. As the relations are extracted into RDF, we construct an intermediate model. With each relation added, we run the model through a consistency check using Pellet [4], an in-memory reasoner. If it is not consistent, we drop the triple, if it is consistent, we add the triple to the final RDF model.

This triple-at-a-time technique has two problems. The main problem is scale: our information extraction technology can process, on normal desktop hardware, roughly a million of documents a day, the SemantiClean system could process hundreds of documents a day. Our analytics today produce about 70 entities (RDF nodes) and about 40 relations (RDF triples) per document, but these numbers can easily change. The point is that the size of the RDF graph could be two orders of magnitude larger than the number of documents.

A second problem is an order dependency created by the triple-by-triple approach of dropping the triple that, when added, causes the knowledge-base to change from consistent to inconsistent. When constraint violations arise from multiple triples together, it is possible that the dropped triple is not the incorrect one.

In two separate experiments, we found SemantiClean improved overall precision of relation extraction by 8% and 15%. As we continue to experiment, there

is evidence to indicate that the precision improvement may increase with the size of the graph, however we have reached the resource limits of Pellet on conventional 64-bit hardware. In the next sections, we describe techniques to scale cleanup.

2.2 Summarization

The techniques we apply in this paper assume ontologies of SHIN expressiveness. A key feature of our approach is the construction of a summary Abox \mathcal{A}' corresponding to the Abox \mathcal{A} . An individual in \mathcal{A}' represents individuals in \mathcal{A} which are members of the same concepts. Formally, an Abox \mathcal{A}' is a summary Abox of a *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 $\mathbf{f}(a) \neq \mathbf{f}(b) \in \mathcal{A}'$

The *image* of an individual u in \mathcal{A}' is the set, denoted $Image(u)$, of individuals a in \mathcal{A} such that $\mathbf{f}(a) = u$. The *accurate image* of a subset L of a summary \mathcal{A}' , denoted $AccImage(L)$, is the subset of \mathcal{A} defined as $AccImage(L) = \{a : C \in \mathcal{A} | \mathbf{f}(a) : C \in L\} \cup \{R(a, b) \in \mathcal{A} | R(\mathbf{f}(a), \mathbf{f}(b)) \in L\} \cup \{a \neq b \in \mathcal{A} | \mathbf{f}(a) \neq \mathbf{f}(b) \in L\}$

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} [2]. However, the converse does not hold.

In general, the summary Abox \mathcal{A}' is dramatically smaller than the original Abox \mathcal{A} .

2.3 Refinement

If the summary Abox is inconsistent, then, in general, we cannot directly conclude anything about the original Abox consistency status.

Our approach [3] for resolving summary Abox inconsistencies is to iteratively *refine* the summary. A refinement step consists of splitting a given summary individual by the sets of role assertions that are present in the original Abox for the Abox individuals mapped to the given summary individual. 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 1) we typically know that we have converged on a real inconsistency (see [3] for more details).

Definition 1 *Let \mathcal{A}' be a summary Abox of an Abox \mathcal{A} obtained through the summary mapping \mathbf{f} . Let H be a subset of \mathcal{A}' . 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. $s : C \in H$ 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 .

3 Summarization and Refinement for Abox Cleanup

In previous work [2] [3], we established that summarization and refinement techniques enable scalable consistency checking and membership query answering over very large and expressive knowledge bases. In this section, we show how these techniques can be adapted to address the issue of detecting and resolving sources of inconsistencies in large knowledge bases such as those generated by text analytic tools.

3.1 The cleanup problem

The Abox cleanup problem consists in identifying consistent subsets of an inconsistent Abox. Ideally, it is desirable to identify maximal consistent subsets, i.e. subsets that are consistent but the addition of a single assertion from the inconsistent Abox yields an inconsistency. Unfortunately, computing a single maximal consistency subset is known to be intractable even for realistic small and medium size expressive Aboxes [5].

In our approach, we do not require a cleansed Abox to be a maximal consistent subset. However, each removed assertion must be associated with a unique justification containing it in the original Abox. Since an assertion is removed to avoid the inconsistency created by its associated justification, two distinct assertions must not be associated with the same justification (there is no need to remove more than one assertion from a justification to avoid an inconsistency). Finally, for two justifications \mathcal{J}_1 and \mathcal{J}_2 having assertions x and y in common, if the justification associated with x is \mathcal{J}_1 , then \mathcal{J}_2 cannot be associated with y (it is clearly not optimal to remove both x and y to avoid the inconsistencies due to \mathcal{J}_1 and \mathcal{J}_2). When these three conditions are satisfied, we say that the cleansed Abox is a *justification-based consistent subset* as formally defined below:

Definition 2 *A justification-based consistent subset CA of an inconsistent Abox \mathcal{A} w.r.t. its Tbox \mathcal{T} and Rbox \mathcal{R} is a consistent subset of \mathcal{A} w.r.t. \mathcal{T} and \mathcal{R} such that there is a set E of justifications of \mathcal{A} and a bijection e from $\mathcal{A} - CA$ to E such that, for assertions x and y in $\mathcal{A} - CA$, the following hold: (a) $x \in e(x)$, and (b) for \mathcal{J}_1 and \mathcal{J}_2 in E such that $\{x, y\} \subseteq \mathcal{J}_1 \cap \mathcal{J}_2$, $e(x) = \mathcal{J}_1$ implies $e(y) \neq \mathcal{J}_2$.*

A maximal consistent subset is always a justification-based consistent subset, but the converse does not hold if non-disjoint justifications exist. The following example illustrates the differences between maximal consistent subsets and justification-based consistent subsets.

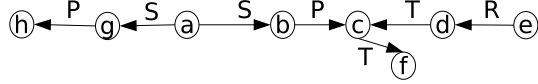


Fig. 1. Simple Abox

$$\mathcal{R} = \{Range(S) = \forall P.A, Range(P) = \neg A, Range(T) = A, Range(R) = \forall T.\neg A\}$$

Computing a justification-based consistent subset does not require an exhaustive search of justifications. For example, Algorithm 1 provides a straightforward computation of a justification-based consistent subset of an inconsistent Abox. For each inconsistency justification found, one of its assertions is removed from the Abox, and the algorithm continues looking for further justifications and removing axioms from the Abox until it becomes consistent. Likewise, the triple-at-time approach described in our previous work [1] yields a justification-based consistent subset. However, depending on the order in which justifications are found and the choices of removed assertions, a particular execution of these two algorithms might not result in a maximal consistent subset. For instance in the previous example (Figure 1), if an execution of Algorithm 1 first discovers J_1 and J_2 (see below) and chooses to remove $S(a, b)$ and $R(e, d)$, it will at some point find J_3 and be forced to remove either $P(b, c)$ or $T(d, c)$; i.e. it will not find a maximal consistent subset.

$$J_1 = \{S(a, b), P(b, c)\} \quad J_2 = \{R(e, d), T(d, c)\} \quad J_3 = \{P(b, c), T(d, c)\} \quad J_4 = \{S(a, g), P(g, h)\} \quad AllJustifications = \{J_1, J_2, J_3, J_4\}$$

$$MA = \mathcal{A} - \{P(b, c), R(e, d), S(a, g)\} \quad JCA = MA - \{S(a, b)\}$$

$$CA_1 = JCA - \{T(c, f)\} \quad CA_2 = JCA - \{P(g, h)\}$$

MA is a maximal consistent subset of \mathcal{A} . JCA is clearly not maximal (it is a proper subset of MA , a maximal consistent subset). However, JCA is a justification-based consistent subset of \mathcal{A} : the set E of explanations is the set of all justifications, and e , defined as $e(S(a, b)) = J_1$, $e(R(e, d)) = J_2$, $e(P(b, c)) = J_3$, and $e(S(a, g)) = J_4$, satisfies conditions (a) and (b) of Definition 2. CA_1 is not a justification-based consistent subset because $T(c, f)$ is not present in any justification. CA_2 is not a justification-based consistent subset because $S(a, g)$ and $P(g, h)$ have both been removed, but they only appear in a single justification, J_4 .

The most important limitation of the previously described algorithms to compute a justification-based consistent subset is their obvious inability to scale

to large and expressive Aboxes containing many inconsistencies, such as those generated from text analytic tools. This is an issue we address in this paper.

```

NaiveJustificationConsistentSubset(Abox  $\mathcal{A}$ , Tbox  $\mathcal{T}$ , Rbox  $\mathcal{R}$ )
begin
   $A \leftarrow \mathcal{A}$ ;
  while inconsistent( $A$ ,  $\mathcal{T}$ ,  $\mathcal{R}$ ) do
    Find a justification  $J$  in  $A$ ;
    select an Abox assertion  $x$  in  $J$ ;
     $A \leftarrow A - \{x\}$ ;
  end
  return  $A$ ;
end

```

Algorithm 1: Naive Justification-based Consistent Subset Computation

3.2 Scalable Approach to Abox Cleanup

In order to scale the computation of a justification-based consistent subset of an Abox \mathcal{A} , our approach identifies justifications in the dramatically reduced summary \mathcal{A}' of \mathcal{A} . Justifications are then refined until they become *precise* or they disappear from the refined summary. Our Algorithm 2 simulates on \mathcal{A}' an execution of the naive algorithm 1 applied to \mathcal{A} .

For an acyclic (considering the undirected graph induced by role and differenceFrom assertions) precise justification in the summary \mathcal{A}' , if it has at least one role assertion, one of its role assertions is removed from the summary; otherwise, one of its concept assertions is removed³. The accurate image of the summary \mathcal{A}'_1 obtained from removing a role assertion $R(u, v)$ of a precise acyclic justification is the Abox \mathcal{A}_1 resulting from the removal of all R role assertions relating images of u to images of v from the original Abox \mathcal{A} . Formally $\mathcal{A}_1 = \mathcal{A} - \{R(a, b) \in \mathcal{A} \mid \mathbf{f}(a) = u \text{ and } \mathbf{f}(b) = v\}$. To show that these assertions removed from \mathcal{A} can also be removed by an execution of the naive algorithm 1, we need to establish that for each removed assertion $R(a, b)$ ⁴: there is a justification J in the Abox such that

- $R(a, b) \in J$, and
- J does not contain any removed assertion besides $R(a, b)$.

Theorem 1 *Let \mathbf{f} be a summary function mapping an Abox \mathcal{A} , inconsistent w.r.t. to its Tbox \mathcal{T} and its Rbox \mathcal{R} , to its summary \mathcal{A}' . Let J' be a precise and acyclic justification of \mathcal{A}' . Let $R(u, v)$ be a role assertion in J' . For individuals a and b in \mathcal{A} such that $\mathbf{f}(a) = u$ and $\mathbf{f}(b) = v$ and $R(a, b) \in \mathcal{A}$, the following conditions hold: there is an inconsistent justification J of \mathcal{A} such that:*

³ We remove role assertions instead of concept assertions because concept assertions produced by our text analytic tools have a higher accuracy than role assertions.

⁴ We focus on role assertions, but the proof for concept assertion removal is similar.

1. $R(a, b) \in J$, and
2. $\{R(x, y) \in \mathcal{A} \mid \mathbf{f}(x) = u \text{ and } \mathbf{f}(y) = v\} \cap J = \{R(a, b)\}$

Proof. The main idea behind the proof is that, viewing J' as a pattern, for each pair of individuals (a, b) in \mathcal{A} such that $\mathbf{f}(a) = u$, $\mathbf{f}(b) = v$, and $R(a, b) \in \mathcal{A}$, there is an instance I_{ab} of the pattern J' in \mathcal{A} such that a is mapped to u and b is mapped to v . In other words, I_{ab} is isomorphic⁵ to J' . Since J' is justification, it follows that I_{ab} is also a justification. Since J' is acyclic, it can always be written as a disjoint union of three sets $J' = J'_u \cup \{R(u, v)\} \cup J'_v$ where J'_u is the subset of J' containing the individual u but not v whereas J'_v is the subset of J' containing v but not u . If J'_u and J'_v are empty (i.e. J' consists of the single edge $R(u, v)$), the existence of I_{ab} is a direct consequence of the summarization process.

We now consider the general case where J'_u and J'_v are not empty (if one of them is empty the proof is similar to this general case). J'_u and J'_v must be acyclic since there are subsets of an acyclic justification. Lemma 1 below establishes the existence of a subset I_a (resp. I_b) of \mathcal{A} isomorphic to J'_u (resp. J'_v) such that a (resp. b) is mapped to u (resp. v) and each individual x of I_a (resp. I_b) is mapped to the individual $\mathbf{f}(x)$ of J'_u (resp. J'_v). It follows that $I_{ab} \stackrel{def}{=} I_a \cup \{R(a, b)\} \cup I_b$ is a subset of \mathcal{A} isomorphic to J' such that, for each individual x in I_{ab} , x is mapped to the individual $\mathbf{f}(x)$ of J' . The isomorphism establishes property (1) of the theorem, namely, I_{ab} is a justification containing $R(a, b)$. Property (2) is a direct consequence of the fact that, for each x of I_{ab} , x is mapped to the individual $\mathbf{f}(x)$ of J' .

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

If L is a non-empty acyclic precise subset of \mathcal{A}' , 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 $s : D \in L$ then $\rho(s) : D \in I$
- (e) $\rho(s) = x$ iff. $\mathbf{f}(x) = s$

Proof. See proof of Lemma 1 in the technical report [6]. Note that property (e) is satisfied by the ρ function presented in the proof.

For a precise cyclic justification J' in the summary, although, in most cases, it can be directly concluded that it corresponds to a real inconsistency in the Abox

⁵ Two Aboxes are isomorphic iff. by renaming individuals in one Abox it becomes identical to the other. The renaming must be such that two individuals with different names in the original Abox are not assigned the same name.

[3], removing an assertion of J' from the summary might be too conservative because the accurate image of the removed assertion in \mathcal{A} might not correspond to a set of Abox assertions that can be removed by an execution of the naive justification-based consistent subset Algorithm 1.

Consider the following example:

$$\mathcal{A} = \{R(a1, a2), R(a2, a3), R(a3, a1), T(a1, b), a1, a2, a3 : A \sqcap \forall R. \forall R. \forall R. \neg A\}$$

$$\mathbf{f}(a1) = \mathbf{f}(a2) = \mathbf{f}(a3) = u, \mathbf{f}(b) = v$$

$$\mathcal{A}' = \{R(u, u), u : A \sqcap \forall R. \forall R. \forall R. \neg A, T(u, v)\} \quad J' = \mathcal{A}' - \{T(u, v)\}$$

J' is a precise cyclic justification, and it can directly be shown, based on the application of deterministic tableau rules to the summary (see [3] for more details), that it represents to a real inconsistency in \mathcal{A} . Removing $R(u, u)$ creates a consistent summary of a consistent subset of \mathcal{A} . Unfortunately, this consistent subset is not a justification-based consistent subset. Indeed, Definition 2 is not satisfied since $R(a1, a2)$ and $R(a2, a3)$, have been removed, but they only appear in a single justification, J' .

To avoid removing more assertions than needed in the Abox, a conclusive precise cyclic justification J' is further refined until at least one role assertion becomes *super precise*, at which point it is removed.

Definition 3 *A role assertion $R(u, v)$ in a summary \mathcal{A}' of an Abox \mathcal{A} is super precise iff. $\{R(u, v)\}$ is precise and $|Image(u)| = |Image(v)| = 1$ (i.e. only one individual in the Abox is mapped to u or v).*

Algorithm 2 represents our summarization and refinement based approach to compute a justification-based consistent subset of a large and expressive Abox \mathcal{A} . It takes as input an inconsistent Abox \mathcal{A} and its Tbox \mathcal{T} and its Rbox \mathcal{R} . It returns a triple consisting of (1) a justification-based consistent subset of \mathcal{A} , (2) a summary of (1), and (3) a map associating a justification J' found in a summary to its assertion that was selected for removal.

Theorem 2 *Algorithm 2 computes a justification-based consistent subset of the input Abox \mathcal{A}*

Proof. After each assertion removal in the summary \mathcal{A}' , \mathcal{A}' is transformed to a summary \mathcal{A}'' . As a direct consequence of Theorem 1 and the definition of super precise role assertion, \mathcal{A}'' is a summary of an Abox which can be created from $AccImage(\mathcal{A}')$ after several iterations of the naive algorithm 1 applied to $AccImage(\mathcal{A}')$. It follows that the final consistent summary \mathcal{A}'_f returned by Algorithm 2 is such that $AccImage(\mathcal{A}'_f)$ is a justification-based consistent subset of \mathcal{A} .

3.3 Approximate Cleanup

Computing a justification-based consistent subset of an Abox \mathcal{A} using Algorithm 2 can be expensive if the summary \mathcal{A}' has precise cyclic justifications whose accurate images in \mathcal{A} are very large. The following example illustrates a worst case situation: $\mathcal{A} = \{R(a_1, a_2), R(a_2, a_3), \dots, R(a_n, a_1), a_1 : A \sqcap \forall R. \neg A\}$

```

JustificationBasedConsistentSubset(Abox  $\mathcal{A}$ , Tbox  $\mathcal{T}$ , Rbox  $\mathcal{R}$ )
begin
   $\mathcal{A}' \leftarrow$  compute the summary Abox of  $\mathcal{A}$ ;
   $Results \leftarrow \emptyset$ ;
  while inconsistent( $\mathcal{A}'$ ,  $\mathcal{T}$ ,  $\mathcal{R}$ ) do
    Find Justifications in  $\mathcal{A}'$ ;
     $ACJ \leftarrow$  select precise acyclic justifications from Justifications;
     $CJ \leftarrow$  select precise conclusive cyclic justifications from Justifications
    that have at least one super precise role assertion;
     $Results \leftarrow Results \cup \text{removeAssertion}(\mathcal{A}', ACJ)$  ;
     $Results \leftarrow Results \cup \text{removeAssertionInCyclicJ}(\mathcal{A}', CJ)$  ;
     $Justifications \leftarrow Justifications - (ACJ \cup CJ)$  ;
    Execute refinement on  $\mathcal{A}'$  using Justifications ;
  end
  return (AccImage( $\mathcal{A}'$ ),  $\mathcal{A}'$ ,  $Results$ ) ;
end
removeAssertion(SummaryAbox  $\mathcal{A}'$ , SetOfJustifications  $JS$ )
begin
   $Results \leftarrow \emptyset$ ;
  for  $J$  in  $JS$  do
    if hasRoleAssertions( $J$ ) then
       $Assertion \leftarrow$  select a role assertion from  $J$ ;
    else
       $Assertion \leftarrow$  select a concept assertion from  $J$ ;
    end
    Remove  $Assertion$  from  $\mathcal{A}'$ ;
     $Results \leftarrow Results \cup (J, Assertion)$ ;
  end
  return  $Results$ ;
end
removeAssertionInCyclicJ(SummaryAbox  $\mathcal{A}'$ , SetOfJustifications  $JS$ )
begin
   $Results \leftarrow \emptyset$ ;
  for  $J$  in  $JS$  do
     $Assertion \leftarrow$  select a super precise role assertion from  $J$ ;
    Remove  $Assertion$  from  $\mathcal{A}'$ ;
     $Results \leftarrow Results \cup (J, Assertion)$ ;
  end
  return  $Results$ ;
end

```

Algorithm 2: Summarization and Refinement based Justification-based Consistent Subset Computation

$\mathcal{R} = \{Trans(\mathcal{R})\}$ (i.e. R is transitive) $\mathcal{A}' = J' = \{R(u, u), u : A \sqcap \forall R. \neg A\}$

J' is a conclusive precise cyclic justification (it is conclusive based on the application of deterministic tableau expansion rules on \mathcal{A}' -see [3] for more details). However, the only precise justification with a super precise role assertion derived from J' through iterative refinement is the whole Abox \mathcal{A} (the length n of the cycle in \mathcal{A} could be very significant)! However, in our experimental evaluation,

we have not witnessed such an extreme situation. In general, conclusive precise cyclic justifications have fairly small accurate images consisting at most of a few dozen individuals.

An alternative strategy is to not require conclusive precise cyclic justifications in the summary to have a super precise role assertion which can then be removed. We simply remove a selected role assertion from the justification. In doing so, Algorithm 3 always produces a consistent subset of the input Abox \mathcal{A} , but this subset is no longer guaranteed to be a justification-based consistent subset. However, when an role assertion $R(u, v)$ is removed from a conclusive precise cyclic justification in the summary, an upper bound of the number of corresponding extraneous Abox assertions removed is given by $|AccImage(\{R(u, v)\})| - 1$. As shown in the experimental evaluation section, in practice, the upper bound of the total number of extraneous removed assertions is a small fraction of the total number of removed assertions. Thus, this approximation is quite precise in practice.

ApproximateJustificationConsistentSubset(Abox \mathcal{A} , Tbox \mathcal{T} , Rbox \mathcal{R})

begin

$\mathcal{A}' \leftarrow$ compute the summary Abox of \mathcal{A} ;

$Results \leftarrow \emptyset$;

while *inconsistent*($\mathcal{A}', \mathcal{T}, \mathcal{R}$) **do**

 Find *Justifications* in \mathcal{A}' ;

$ACJ \leftarrow$ select precise acyclic justification from *Justifications*;

$CJ \leftarrow$ select precise conclusive cyclic justifications from *Justifications*;

$Results \leftarrow Results \cup \text{removeAssertion}(\mathcal{A}', ACJ \cup CJ)$;

$Justifications \leftarrow Justifications - (ACJ \cup CJ)$;

 Execute refinement on \mathcal{A}' using *Justifications* ;

end

return (*AccImage*(\mathcal{A}'), \mathcal{A}' , $Results$) ;

end

Algorithm 3: Summarization and Refinement based Approximation of Justification-based Consistent Subset Computation

4 Explanation

In this section, we discuss the appropriate presentation of information about justifications computed by algorithms presented in the previous section.

As shown in the experimental evaluation section, large Aboxes produced by text analytic tools may have thousands of inconsistencies involving more than one assertion. A justification in the summary can be viewed as a pattern having one or more instances (or isomorphic justifications) in the actual Abox. A summary justification provides a level of abstraction that represents all Abox justifications isomorphic to it.

Furthermore, in our experiments, we have observed that many acyclic justifications in the summary are isomorphic. Therefore, we can represent a group of isomorphic summary justifications with a single justification, called an abstract summary justification. An abstract summary justification abstracts out the names of nodes in the summary, and represents all Abox justifications that have the same pattern of role and concept assertions. Thus, it can be represented as SPARQL query describing its patterns of role and concept assertions. This SPARQL query can then be used to retrieve the set of triples that are involved in the justifications.

For cyclic justifications in the summary such that all role assertions are super precise, we can also group isomorphic justifications and represent them by a single abstract justification, which can be associated with a SPARQL query. However, if a cyclic justification has some edges that are not super precise, the exact pattern represented by the justification is not known (e.g. if the justification consists of just a single role assertion relating an individual to itself, it definitely represents at least one simple cycle in the Abox, but the length of the cycle is unknown). If the user wants more details on a cyclic justification with role assertions that are not super precise, its accurate image in the Abox will have to be retrieved.

5 Computational Experience

Our tests were conducted on a 64 bit AMD dual 1GMHz processor 8G RAM Linux machine, and a maximum heap size of 1G. The datasets were stored in a DB2 database. We tested the performance and scalability properties of our approach (Algorithm 3) with 4 datasets generated from text analysis of 100, 500, 1500 and 3683 documents. Table 1 shows the characteristics of these datasets from the perspective of the number of individuals (I), the number of role assertions (R.A.), the number of summary justifications found (J), the number of abstract summary justifications (A.J.), the number of deleted assertions (D.A.), the computed upper bound of the number of extra triples that would have been removed for these datasets if all of these triples were eliminated (Max. E.D.A.), and the time to find these justifications in minutes. The last column represents the number of overlapping abstract justifications of each dataset with the 500 document dataset.

Dataset	I	R.A.	J	A.J.	D.A.	Max. E.D.A.	Time	AJ Overlap with 500
100	8,628	15,521	191	97	299	19	10	84
500	32,787	62,414	625	203	1,150	89	19	203
1500	104,507	195,206	1,570	360	3,910	359	37	169
3683	286,605	513,522	2,744	561	9,574	967	67	168

As shown in Table 1, the time to find justifications is linear. We compare our results for detecting inconsistencies with a technique described in [1]. For

the purposes of this comparison, we focus on the 500 document dataset, because this dataset was used by the triple-at-a-time technique [1] to validate the use of inconsistency checking for cleansing data extracted from text. When a knowledge base has justifications with multiple role assertions, the two techniques are not likely to agree on the set of inconsistent triples because of different choices of role assertions to remove. However, because a vast majority of justifications in the 500 document dataset have a single role assertion, each of the triples found to be inconsistent by the triple-at-a-time technique are also found to be inconsistent using our technique.

The column labeled Max. E.D.A., which represents an upper bound of the number of extraneous assertions removed by our Algorithm 3, shows that the approximate Algorithm 3 is quite precise: the number of extraneous assertions is always less than 10% of the removed assertions and less than 0.1% of the total number of assertions in the original dataset. Moreover, this upperbound is conservatively computed as indicated in section 3.3. In practice, the real number of extraneous assertions might be much smaller. For example, while the computed conservative upperbound of the number of extraneous assertions for the 500 dataset is 89, the actual number was just 7. We removed all assertions removed by the triple-at-a-time technique plus 7 additional assertions.

Finally, the last column of Table 1 indicates substantial overlap between the justifications found in the different datasets. This suggests that patterns of justifications found in small datasets can directly be searched and removed from much larger datasets before starting Algorithm 2 or 3.

6 Sophisticated Cleanup techniques

The fully-automated cleanup strategy based on Algorithm 2 or 3 is inflexible in that it randomly selects and removes assertions present in justifications. An alternative is to make the process more flexible and interactive by allowing the user to specify only a fragment of the error-causing assertions to be removed.

For this purpose, it would be desirable if the system could assist the user in determining likely assertion candidates for removal, by using some sensible metrics for *ranking* assertions. [5] presents a set of strategies for ranking assertions in justifications, including:

- the number of distinct justifications that an assertion appears in – higher the frequency, lower the assertion rank, since it signifies that the assertion is responsible for producing more errors,
- provenance information about the assertion such as its source – in this case, the accuracy of the text analytic tools generating the assertion would be relevant,
- using a history of previous error patterns to identify suspicious assertions – which in our case amounts to storing abstract justifications from previous cleanup sessions on similar Aboxes.

Given a set of ranked assertions, the user could then choose to remove only the low priority assertions from the Abox. However, in doing so, the resultant Abox is not guaranteed to be consistent (since the earlier approach only guarantees a consistent Abox if the specific randomly selected assertions in the justifications were removed). Thus, we would need to run Algorithm 2 or 3 over the modified Abox again to obtain a new set of justifications, and repeat this process of ranking and removing erroneous assertions iteratively till the entire Abox was clean.

7 Related Work

Recently, there has been a lot of work on repairing inconsistencies in OWL-DL Ontologies. Broadly, this work falls into two categories. The first approach, described in [7], [5], involves identifying a single source of the inconsistency (justification) in the ontology by modifying the internals of the DL tableaux reasoner (a technique known as *tableau tracing*), and then using Reiter’s Hitting Set Tree Algorithm [8] to discover all justifications, in order to arrive at a maximal consistent subset of the ontology. We employ the tableau tracing solution in [5] to derive a single justification. However, given the exponential nature of Reiter’s search, using it to fully repair an ontology containing hundreds or thousands of inconsistency-causing justifications, as is in our case, is clearly not feasible.

The second approach is based on phrasing the problem as a belief revision as done in [9], and then revising the knowledge base to get rid of the inconsistency by *rewriting* the axioms to preserve semantics, e.g., introducing disjunctions. On a similar note, [10] proposes tolerating inconsistent theories and using a non-classical form of inference to derive meaningful results from a consistent sub-theory. These solutions do not fit in within the nature of our application given the abundance of inconsistencies, and the fact that most of the errors are actual noise (not partially correct, rewritable axioms) created by the text analytics tools. Instead, we plan to use metrics for ranking erroneous axioms as suggested in [5] to perform sophisticated cleanup (Section 6), based on factors related to the text analytic process.

The other key difference, though, in terms of related work, is that none of the approaches presented above can scale to very large Aboxes containing millions of assertions. Our Abox summarization and refinement techniques have been shown to scale Abox reasoning in a massive way [3].

To conclude, the solution in this paper is the first of its kind that provides a scalable and efficient way to clean very large Aboxes containing numerous inconsistencies.

8 Conclusions

Modern Natural Language Processing techniques are extremely scalable but generate noisy data. In previous work we have introduced the problem of cleaning

noisy data using semantic web technologies. While showing an overall improvement in precision, the approach had scalability problems, as well as an order dependency. We have been also investigating techniques for summarization and refinement of Aboxes for scalability. In this paper we put the two together, and introduce a new technique for justification-based consistent subset finding that identifies patterns of inconsistent data in a scalable way.

We are only in the initial phases of putting together scalable inference with scalable natural language processing, but the results presented here are extremely promising; early experiments show a *linear increase* in processing time as the data increases. Due to the challenges of scaling formal evaluations, we have only presented results that scale to an RDF graph of a few hundred thousand nodes, as (random) subsets of this graph have been verified. In other experiments we have managed to process millions of RDF nodes, and we have every reason to believe this technique will scale at the same rate as our information extraction, while improving precision of relation extraction by 8-15% or more.

References

1. Welty, C.A., Murdock, J.W.: Towards knowledge acquisition from information extraction. In: Proc. of the fifth International Semantic Web Conference. (2006) 709–722
2. A.Fokoue, A.Kershenbaum, L.Ma, E.Schonberg, K.Srinivas: The summary abox: Cutting ontologies down to size. Proc. of the Int. Semantic Web Conf. (ISWC 2006) (2006) 136–145
3. Dolby, J., A.Fokoue, Kalyanpur, A., A.Kershenbaum, L.Ma, E.Schonberg, K.Srinivas: Scalable semantic retrieval through summarization and refinement. (In: Proc. of the AAAI Conf. 2007 (To appear) [http://domino.research.ibm.com/comm/research_projects.nsf/pages/iaa.index.html/\\$FILE/SHER-AAAI2007.pdf](http://domino.research.ibm.com/comm/research_projects.nsf/pages/iaa.index.html/$FILE/SHER-AAAI2007.pdf))
4. Sirin, E., Parsia, B.: Pellet: An owl dl reasoner. In: Description Logics. (2004)
5. Kalyanpur, A.: Debugging and Repair of OWL-DL Ontologies. PhD thesis, University of Maryland, <https://drum.umd.edu/dspace/bitstream/1903/3820/1/umi-umd-3665.pdf> (2006)
6. Julian Dolby, e.a.: Technical report: Scalable semantic retrieval through summarization and refinement. In: [http://domino.research.ibm.com/comm/research_projects.nsf/pages/iaa.index.html/\\$FILE/techReport2007.pdf](http://domino.research.ibm.com/comm/research_projects.nsf/pages/iaa.index.html/$FILE/techReport2007.pdf). (2007)
7. Schlobach, S.: Diagnosing terminologies. In: In Proceedings of AAAI-05. (2005) 670–675
8. Reiter, R.: A theory of diagnosis from first principles. *Artificial Intelligence* **32** (1987) 57–95
9. Meyer, T., Lee, K., Booth, R.: Knowledge integration for description logics. In: AAAI. (2005) 645–650
10. Huang, Z., van Harmelen, F., ten Teije, A.: Reasoning with inconsistent ontologies. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI'05), Edinburgh, Scotland (2005) xxx