SOUND SUMMARIZATIONS FOR $\mathcal{ALCHI}$ ONTOLOGIES
How to Speedup Instance Checking and Instance Retrieval

Sebastian Wandelt and Ralf Möller
Institute for Software Systems, Hamburg University of Technology, Schwarzenbergstrae 95, Hamburg, Germany

Keywords: Description logics, Ontologies, Reasoning, Scalability.

Abstract: In the last years, the vision of the Semantic Web fostered the interest in reasoning over ever larger sets of assertional statements in ontologies. In this senario, state-of-the-art description logic reasoning systems cannot deal with real-world ontologies any longer, since they rely on in-memory structures.
In these scenarios it will become more important to rely on unsound or incomplete reasoning structures, to obtain a set of candidates/obvious solutions to queries, i.e. only apply state-of-the-art reasoning systems to the computationally *hard* solutions. In this paper we propose a summarization-based approach which is always sound, but possibly incomplete. We think that this technique will support description logic systems to deal with the steadily growing amounts of assertional data.

1 INTRODUCTION

As the Semantic Web evolves, scalability of inference techniques becomes increasingly important. Even for basic description logic-based inference techniques, e.g. concept satisfiability, it is only recently understood on how to perform reasoning on large ABoxes in an efficient way. This is not yet the case for problems that are too large to fit into main memory. In this paper we present an approach to optimize instance retrieval tests on ontologies, by summarization of similar individuals in the assertional part of an ontology. The general picture is as follows:

- Keep a sound, but possibly incomplete data-structure, called Summarization, in main memory
- On incoming queries, consult the summarization first to obtain a list of candidate solutions/obvious answers
- For all left-over potential answers, compute the outcome with a description logic reasoner, e.g. in our case Racer(Haarslev and Möller, 2001)

The remaining part of the paper is structured as follows. Section 2 provides the formal background for description logics, presents some related work and introduces an example ontology, which will be used throughout the paper. In Section 3 and 4, we present our data-structure for sound, and possibly incomplete reasoning. How to use these results for description logic reasoning is shown in 5. In Section 6 we present preliminary evaluation of the proposed algorithm with respect to a benchmark ontology. We conclude with Section 7.

2 FOUNDATIONS

2.1 Description Logics

We briefly recall syntax and semantics of the description logic $\mathcal{ALCHI}$. For the details about the description logic $\mathcal{ALCHI}$, please refer to (Baader et al., 2007). We assume a collection of disjoint sets: a set of *concept names* N_{CN} , a set of *role names* N_{RN} and a set of *individual names* N_I. The *set of roles* N_R is $N_{RN} \cup \{R^- | R \in N_{RN}\}$. We say that a concept description is *atomic*, if it is a concept name. With N_C we denote all atomic concepts. For defining the semantics see (Baader et al., 2007). Furthermore we assume the notions of TBoxes ($\mathcal{T}$), RBoxes ($\mathcal{R}$) and ABoxes ($\mathcal{A}$) as in (Baader et al., 2007). With $ASSERTIONS(\mathcal{A})$ we denote the set of assertions in an ABox $\mathcal{A}$ and with $S_{\mathcal{A}}$ we denote the set of all ABoxes.

A *ontology* O consists of a 3-tuple $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox, $\mathcal{R}$ is a RBox and $\mathcal{A}$ is a ABox. We restrict the concept assertions in $\mathcal{A}$ in such a

Wandelt S. and Möller R. (2010).
SOUND SUMMARIZATIONS FOR ALCHI ONTOLOGIES - How to Speedup Instance Checking and Instance Retrieval.
In *Proceedings of the 2nd International Conference on Agents and Artificial Intelligence - Artificial Intelligence*, pages 656-661
DOI: 10.5220/0002762206560661

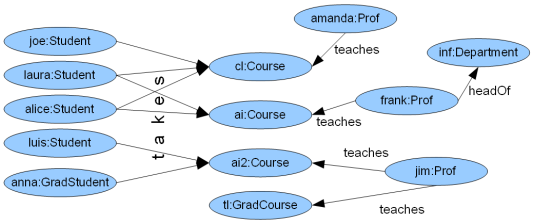

Figure 1: Guiding Example: ABox $\mathcal{A}_{EX}$ for ontology O_{EX}.

way that each concept description is an atomic concept or a negated atomic concept. This is a common assumption, e.g. in (Guo and Heflin, 2006), when dealing with large assertional datasets in ontologies. With $Ind(\mathcal{A})$ we denote the set of individuals occurring in $\mathcal{A}$. The set of equivalence classes of an equivalence relation Rel is denoted $[Rel]$. When defining an equivalence relation, we will sometimes use the following syntax for convenience: $R = \{(a,b,c,d),(e,f),(g,h,i)\}$. The meaning is, that a,b,c,d are equivalent, e and f are equivalent, and g,h,i are equivalent w.r.t. R. The expression $ran(f)$ denotes the range of a function f. Whenever we have a pair $p = \langle A,B\rangle$, we access the elements by a dot-operator, e.g. $p.A$.

2.2 Related Work

Recently, an approach for partitioning large OWL ontologies has been presented in (Guo and Heflin, 2006). The idea is to partition a large ABox into smaller ABoxes, s.t. reasoning on the smaller assertional subsets is complete, but possibly unsound. Although the authors report impressive results for the increase in performance, we see some issues as identified (Wandelt, 2008). In (Fokoue et al., 2006), the authors propose a method to reduce the number of individuals in an ABox for complete but unsound reasoning. Afterwards, a filtering algorithm is applied to obtain soundness again. This filter-step is usually quite sophisticated and time-consuming. In a similar way as the partitioning-approach given in (Guo and Heflin, 2006), a Summary ABox has to be build in a precomputation step, which depends on the actual and complete ABox. Thus, the approach is not per-se applicable to updateable ontologies. After all, our work can be seen as complementary to other work. For more information refer to Section 6.

2.3 Guiding Example

In the following we define an example ontology, which is used throughout the remaining part of the paper. The ontology is inspired by LUBM (Guo et al., 2005), a benchmark-ontology in the setting of universities. Although this is a synthetic benchmark, several (if not most) papers on scalability of ontological reasoning consider it as a base reference. We take a particular a snapshot from the LUBM-ontology (Example 1) to make our approach more visible and clear. We evaluate our ideas w.r.t. to "full" LUBM in Section 6.

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

$$\mathcal{T}_{EX} = \{Chair \equiv \exists headOf.Department \sqcap Person, Prof \sqsubseteq Person$$
$$Graduate \sqsubseteq Student, Student \equiv Person \sqcap \exists takes.Course\}$$
$$\mathcal{R}_{EX} = \{headOf \sqsubseteq worksFor\}$$
$$\mathcal{A}_{EX} = \texttt{see Figure 1}$$

3 ABOX SUMMARIZATIONS

Definition 1. *Let* N_{sum} *be a set of summarization individuals, then a* ABox-Summarization *(for an ABox* $\mathcal{A}$*) is a pair* $AS = \langle \phi_{sum}, \omega_{sum}\rangle$, *s.t.*

- ϕ_{sum} *is a total function* $Ind(\mathcal{A}) \rightarrow N_{sum}$ *and*
- ω_{sum} *is a total function* $N_{sum} \rightarrow S_{\mathcal{A}}$

The intuition of Definition 1 is that ϕ_{sum} maps each named individual in the source ABox $\mathcal{A}$ to a summarization individual, and ω_{sum} determines a set of ABox-assertions for each such summarization individual.

Example 2. *Example for ABox-Summarization* AS_{EX1} *w.r.t.* $\mathcal{A}_{EX}$*:*

$$N_{sum} = \{s,c,p\}$$
$$\phi_{sumEX} = \{(joe,s),(laura,s),(alice,s),(luis,s),(anna,s),(cl,c),(ai,c),(ai2,c),(tl,c),(amanda,p),(frank,p),(jim,p),(inf,p)\}$$

$$\omega_{sumEX} = \{(s,\{Student(s),takes(s,e),Course(e)\}),(c,\{Course(c)\}),(p,\{Prof(p)\})\}$$

Please note that the ABoxes ω_{sumEX} might introduce new individual names, e.g. the name e here.

We will define ϕ_{sum} usually by use of an equivalence relation $R_{\phi_{sum}}$, where N_{sum} (the range of ϕ_{sum}) is assumed the set of equivalence classes in $R_{\phi_{sum}}$, i.e. $[R_{\phi_{sum}}]$. The correspondence between both notions is as defined follows:

$$\phi_{sum}(a) = \phi_{sum}(b) \iff R_{\phi_{sum}}(a,b)$$

Example 3. *An example equivalence relation $R_{\phi_{sum}EX}$ for ϕ_{sumEX} could be:*

$$\begin{aligned} R_{\phi_{sum}EX} = \{&(joe, laura), (laura, joe),\\ &(laura, alice), (alice, laura),\\ &(joe, alice), (alice, joe),\\ &..\\ &(frank, jim), (jim, frank),\\ &(frank, frank)\} \end{aligned}$$

Let us relate ABox-Summarizations to description logic reasoning. Therefore, we can distinguish two properties: soundness and completeness with respect to description logic inferences.

Definition 2. *An ABox-Summarization (for an ABox $\mathcal{A}$) is* sound*, if we have that*

$$\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \rangle \models C(\phi_{sum}(a)) \implies \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a).$$

Definition 3. *A ABox-Summarization (for an ABox $\mathcal{A}$) is* complete*, if we have that*

$$\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a) \implies \langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \rangle \models C(\phi_{sum}(a)).$$

Both properties can be used to assess the quality of an ABox-Summarization, i.e. if you have neither soundness nor completeness, then the ABox-Summarization is usually not of use for description logic inferences. To further evaluate ABox-Summarizations, we introduce the notion of "'amicability"', which estimates how much the ABox-Summarization can speed up solving description logic decision problems.

Definition 4. *The* amicability *of an ABox-Summarization $AS = \langle \phi_{sum}, \omega_{sum} \rangle$ (for an ABox $\mathcal{A}$) is defined as*

$$amic(AS) = -log_{10}(\frac{ran(\phi_{sum})}{|Ind(\mathcal{A})|})$$

Example 4. *An Example ABox-Summarization AS_{EX2}, which is trivially sound and complete, is*

- $N_{sum} = Ind(\mathcal{A}_{EX})$
- $R_{\phi_{sum}EX} = \{(x,x) | x \in Ind(\mathcal{A}_{EX})\}$
- $\omega_{sumEX}(x) = \mathcal{A}_{EX}$, *for all* $x \in Ind(\mathcal{A}_{EX})$

In the example above, we do not merge any two individuals, and for reasoning over each individual we use the whole ABox. As one can see from $amic(AS_{EX1}) = 0$, this trivial "'summarization"' will not yield any improvement for reasoning over individuals.

4 SOUND SUMMARIZATION

In the following we discuss a non-trivial summarization, which always enables sound reasoning, and in some cases even complete reasoning; with a quite encouraging amicability. To define the summarizations, we first look at which individuals we want to merge. Our general intuition is, informally, that similar individuals should behave equally during reasoning. Thus we will define some notions to talk about similarity of two individuals in an ABox $\mathcal{A}$.

Definition 5. *An* Anonymous Node Successor *(ANS) of an individual a for an ABox $\mathcal{A}$ is a pair $ANS^a_{\mathcal{A}} = \langle rs, cs \rangle$, s.t. $\exists b \in Ind(\mathcal{A})$ with*

1. $\forall R \in rs.(R(a,b) \in \mathcal{A} \vee R^-(b,a) \in \mathcal{A})$
2. $\forall C \in cs.C(b) \in \mathcal{A}$
3. *cs and rs are maximal*

The third criteria (maximality) is important for cases, when two individuals can be connected by more than one role assertion.

Example 5. *An example for two anonymous node successors of frank are:*

- $ANS1^{frank}_{\mathcal{A}_{EX}} = \langle \{teaches\}, \{Course\} \rangle$
- $ANS2^{frank}_{\mathcal{A}_{EX}} = \langle \{headOf\}, \{Department\} \rangle$

Next, we combine all anonymous node successors of an individual a in an ABox $\mathcal{A}$ and add the directly asserted concepts of a, to create a summarization representative, called One Step Node.

Definition 6. *A* One Step Node *of an individual a for an ABox $\mathcal{A}$ is a pair $OSN^a_{\mathcal{A}} = \langle rootconset, ansset \rangle$, s.t.*

- $rootconset = \{C | a : C \in \mathcal{A}\}$ *and*
- *ansset is the set of all Anonymous Node Successors of individual a*

Example 6. *We have*

$$\begin{aligned} OSN^{frank}_{\mathcal{A}_{EX}} = \langle \{Prof\}, \{ &\langle \{teaches\}, \{Course\} \rangle,\\ &\langle \{headOf\}, \{Department\} \rangle \} \rangle \end{aligned}$$

We use One Step Nodes to define a similarity relation among individuals in an ABox $\mathcal{A}$.

Definition 7. *Two individuals a and b are called* One Step Node-similar *for an ABox $\mathcal{A}$, if we have $OSN^a_{\mathcal{A}} = OSN^b_{\mathcal{A}}$. We denote the One Step Node-similarity relation for an ABox $\mathcal{A}$ with R_{OSNSim}.*

And last but not least, we define the notion of an OSN-based ABox-Summarization.

Definition 8. *An* ABox-OSN-Summarization *(for an ABox $\mathcal{A}$ and an One Step Node-similarity relation R_{OSNSim}) is an ABox-Summarization $AOSNS_{\mathcal{A}} = \langle \phi_{sum}, \omega_{sum} \rangle$, s.t.*

- N_{sum}=$[R_{OSNSim}]$
- *We define ϕ_{sum} by the equivalence relation R_{OSNSim}*
- *For each $osn \in [R_{OSNSim}]$*

$$\omega_{sum}(osn) = \{C(osn^*) | C \in osn.rootconset\} \cup \bigcup_{ans \in osn.ansset} \{R(a, ans^*) | R \in ans.rs\} \cup \bigcup_{ans \in osn.ansset} \{C(ans^*) | C \in ans.cs\}$$

, where x^* denotes a fresh and unique individual name for each OSN-/ANS-object x.

Example 7. *One ABox-OSN-Summarization AS_{EX3} for $\mathcal{A}_{EX}$ and $R_{\phi_{sum}EX3} = \{(joe, lara, alice, luis), (cl, ai, ai2), (tl), (ana), (amanda), (frank), (jim), (inf)\}$ is*

$$N_{sumEX3} = \{s1, s2, c1, c2, p1, p2, p3, d1\}$$
$$\phi_{sum}(joe) = s1$$
$$\phi_{sum}(lara) = s1$$
$$..$$
$$\phi_{sum}(cl) = c1$$
$$..$$
$$\phi_{sum}(inf) = d1$$
$$\omega_{sumEX3}(s1) = \{Student(s1), takes(s1, cnew1), Course(cnew1)\}$$
$$\omega_{sumEX3}(s2) = \{GradStudent(s2), takes(s2, cnew2), Course(cnew2)\}$$
$$..$$
$$\omega_{sumEX3}(p2) = \{Prof(p2), teaches(p2, cnew5), Course(cnew5), headOf(p2, newd1), Department(newd1)\}$$
$$..$$

5 HOW TO SOLVE DECISION PROBLEMS?

The main theorem of our work is presented in Theorem 5.1.

Theorem 5.1. *Given an Ontology $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, every ABox-OSN-Summarization $AOSNS_{\mathcal{A}}$ for $\mathcal{A}$ is sound.*

This is, informally, clear, since the ABox of each summarization individual is a subset of the original ABox $\mathcal{A}$ (modulo renaming).

Proof. We have to show that $\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \rangle \models C(\phi_{sum}(a)) \implies \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a)$.

We show the proof by contraposition and then derive a contradiction: Given $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \not\models C(a)$, we have that there exists a model I_1 for $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \cup \{\neg C(a)\} \rangle$. Now, I_1 has to satisfy all the assertions in $\mathcal{A} \cup \{\neg C(a)\}$, and since $\omega_{sum}(\phi_{sum}(a)) \cup \{\neg C(\phi_{sum}(a))\}$ is structurally equivalent (by construction in Definitions 5, 6 and 8) to a subset of $\mathcal{A} \cup \{\neg C(a)\}$, we have that a rewriting of I_1 has to satisfy all the assertions in $\omega_{sum}(\phi_{sum}(a)) \cup \{\neg C(\phi_{sum}(a))\}$. Thus, there exists a model for $\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \cup \{\neg C(\phi_{sum}(a))\} \rangle$. Contradiction, since we assumed $\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \rangle \not\models C(\phi_{sum}(a))$ (by Contraposition). □

Given Theorem 5.1 above, we can speedup query answering over description logics in the following ways.

Given an individual a and an atomic concept description C, *Instance Checking* is the problem to determine, whether an Ontology $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a)$. In common description logic systems, this is done by checking, whether $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \cup \{\neg C(a)\} \rangle$ is inconsistent. Once the ABox is reasonably big, the underlying exponential behavior of tableau algorithms shows up easily, albeit all known optimizations techniques.

Given an ABox-OSN-Summarization $AOSNS_{\mathcal{A}}$ for $\mathcal{A}$, we can perform some kind of sanity check with existing tableau algorithms, but dramatically reduced ABoxes.

1. Check, whether $\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \rangle \not\models C(\phi_{sum}(a))$. If this is true, then we already know that $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a)$.
2. Check, whether $\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(\phi_{sum}(a)) \rangle \not\models \neg C(\phi_{sum}(a))$. If this is true, than we already know that $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models \neg C(a)$, and it is safe to assume that $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \not\models C(a)$ (if the underlying ontology is consistent).

Please note that we deal with a dramatically reduced assertional part in both cases. Only if both sanity checks fail, then we still have to apply the full reasoning machine, i.e. we have to check $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a)$.

The main contribution of this work is the speedup of Instance Retrieval. Given an atomic concept description C, *Grounded Instance Retrieval* is the problem to determine all individuals $a \in \mathcal{A}$, s.t. we have $\langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle \models C(a)$.

Given an ABox-OSN-Summarization $AOSNS_{\mathcal{A}}$ for $\mathcal{A}$, we can obtain a sound result set as follows:

1. $SoundResult = \emptyset$
2. For all $s \in N_{sum}$ do
   - If $\langle \mathcal{T}, \mathcal{R}, \omega_{sum}(s) \rangle \models C(s)$, then $SoundResult = SoundResult \cup \phi^{-}_{sum}(a)$

Informally, we iterate over all summarization individuals $s \in N_{sum}$ and check, whether they are instances of concept description C. If yes, then we add all individuals represented by that summarization individual to the set of sound answers.

6 IMPLEMENTATION AND EVALUATION

We implemented our proposal in Java with the help of OWLAPI(Bechhofer et al., 2003) and investigated several criteria for evaluation w.r.t. to the Lehigh University Benchmark with up to 1000 universities. Each university in the benchmark has around 20 associated departments. Please note that we have implemented all data structures in an updateable way. This is important when one deals with constantly changing information, e.g. in streams of information. To the best of our knowledge, other related work does not yet discuss updateable information yet.

1. Figure 2 shows the time needed to create ABox-OSN-Summarizations for a given number of universities. It is worth to notice that the loading time exhibits linear-time behavior, which corresponds to the size of the input data. When breaking down the loading time to triples, we are able to load around 4000 Triples/second in average.
2. Number of summarization individuals Figure 4 shows the size of N_{sum}, i.e. the number of summarization individuals in the corresponding ABox-OSN-Summarization. The number of individuals remains constant over time. Only during loading the first universities, the size grows, until the Summarization becomes saturated, i.e. no more new One Step Nodes are created, but only already known One Step Nodes used. The constant number enables constant-time sound instance retrieval in case of LUBM.
3. Time for Sound Instance Retrieval We performed Grounded Instance Retrieval for the concept description *Chair* over LUBM. As expected, the time for sound instance retrieval, shown in Figure 4, is constant as well (the number of Summarization Individuals does not change). Please note that Figure 4 only shows the time for real reasoning over the corresponding ABox-OSN-Summarization with Racer.

   Separately, we show the time necessary to look up the individual names for the summarization individuals in Figure 5. It is easy to see that, while the reasoning itself is constant, the lookup time grows linear. The reason is that the number of *Chair*-instances grows linearly as well, as shown in Figure 6. E.g. in LUBM with 1000 universities, we have already around 20000 *Chair*-instances.
4. Amicability Finally, we show the amicability of the corresponding ABox-OSN-Summarization in Figure 7.
5. Completeness It is interesting to notice that our further analysis showed that all ABox-OSN-Summarization for LUBM turned out to be not only sound, but also complete. That means, we have constant time grounded instance retrieval for LUBM. The reason is that most of the terminological axioms in LUBM are rather domain/range-constraints, which are completely covered by our One Step Nodes. We did not have time to finish these investigations, but the results look quite encouraging so far.

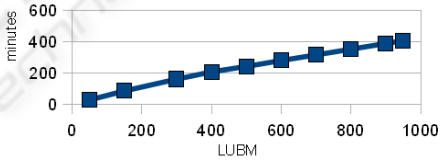

Figure 2: Loading time.

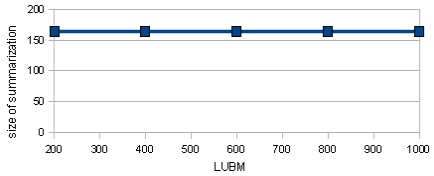

Figure 3: Size of N_{sum}.

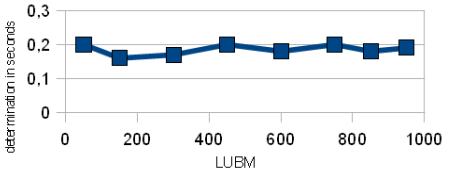

Figure 4: Reasoning over summarization individuals - solution determination.

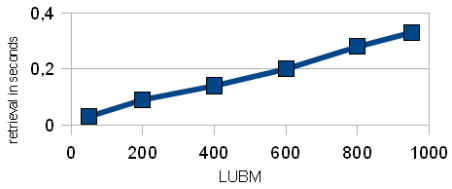

Figure 5: Reasoning over summarization individuals - look up.

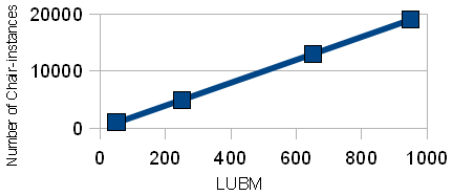

Figure 6: Number of Chair instances.

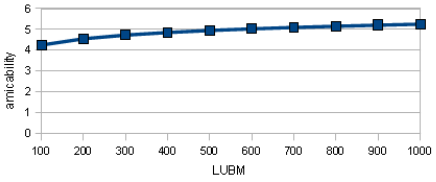

Figure 7: Amicability.

7 CONCLUSIONS AND FUTURE WORK

We have proposed a method for sound instance retrieval over ontologies. Our idea for summarization of individuals is not completely new, but to the best of our knowledge we are the first to propose sound reasoning over individuals based on similarity. The results are encouraging so far, especially in combination with the recently discovered completeness-property for LUBM. Even though other real-world ontologies might not share the completeness-property, it seems likely, that large subsets of these ontologies are still domain-/range-constraints, and thus easily covered by ABox-OSN-Summarizations. It was furthermore shown that the approach has potential to deal with updateable information.

For Future Work, we plan to further investigate the completeness of ABox-OSN-Summarizations. Furthermore it will be important to investigate more ontologies and check the performance of our proposal. In the end, LUBM is only a synthetic benchmark created to score description logic reasoning systems. But whether it will perform in real-world scenarios is still vague. Finally, we want to extend our proposal to more expressive description logics, e.g. SHIQ or even SHOIQ. While the extension is easy w.r.t. soundness, completeness testing seems to be the harder problem.

REFERENCES

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F. (2007). *The Description Logic Handbook*. Cambridge University Press, New York, NY, USA.

Bechhofer, S., Volz, R., and Lord, P. (2003). Cooking the Semantic Web with the OWL API.

Fokoue, A., Kershenbaum, A., Ma, L., Patel, C., Schonberg, E., and Srinivas, K. (2006). Using Abstract Evaluation in ABox Reasoning. In *SSWS 2006*, pages 61–74, Athens, GA, USA.

Guo, Y. and Heflin, J. (2006). A Scalable Approach for Partitioning OWL Knowledge Bases. In *SSWS 2006*, Athens, GA, USA.

Guo, Y., Pan, Z., and Heflin, J. (2005). Lubm: A benchmark for owl knowledge base systems. *J. Web Sem.*, 3(2-3):158–182.

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

Wandelt, S. (2008). Partitioning owl knowledge bases - revisited and revised. In *Description Logics*.