Proc. 23rd Int. Workshop on Description Logics (DL2010), CEUR-WS 573, Waterloo, Canada, 2010.

Towards Soundness Preserving Approximation for ABox Reasoning of OWL2

Yuan Ren, Jeff Z. Pan and Yuting Zhao

Department of Computing Science
University of Aberdeen
Aberdeen, UK

Abstract. ABox Reasoning in large scale description logic (DL) knowledge bases, e.g. ontologies, is important for the success of many semantic-enriched systems. Performance of existing approaches, such as the tableau-based approach, and the disjunctive datalog approach, is restricted by their theoretical worst case complexity bound. In this paper, we propose a soundness-preserving approximate reasoning approach to address this issue. We first approximate an ontology in DL $\mathcal{RO}$, a major fragment of OWL2-DL, to DL $\mathcal{EL}^{++}$, the underpin of OWL2-EL, plus an additional table maintaining the complementary relations between concept names. Then we can perform ABox reasoning either internally, or externally of the TBox with additional completion rules. The approximation and reasoning can be performed in PTIME. Our preliminary evaluation shows that our approach can outperform existing DL reasoners on real world and benchmark ontologies.

1 Introduction

With the fast development of the semantic web and knowledge intensive systems, the representation and reasoning over large-scale ontologies have become important topics for research community. Web Ontology Language (OWL), the de facto standard ontology language, is based on the family of Description Logics (DLs). In the last decades, many research attentions have been paid to complexity and reasoning algorithms of various dialects of the DLs such as $\mathcal{SROIQ}$, the underpinning of the OWL2-DL, and $\mathcal{EL}^{++}$, the underpinning of the OWL2-EL [9].

Most of these works focus on TBox reasoning such as deciding subsumption between concept expressions, or checking whether a particular concept is satisfiable. ABox reasoning such as deciding to which concepts a particular individual belongs is usually realised by extensions of TBox algorithms [6], or by being reduced to TBox reasoning [12]. Other works [7] reduces ontologies into disjunctive datalog to provide dedicated ABox reasoning. In either case, the reasoning complexity is high for expressive DL fragments. However, ABox can be encoded in very expressive DLs, rather large and changing frequently for which traditional solutions can not provide efficient answers.

To solve this problem, the approximation approaches have been studied and evaluated [10, 14, 3, 13, 14]. However, most of these works still replying on reasoners of expressive DLs. For example, [10] achieves efficient query answering by pre-computing the materialization of the original ontology with an OWL DL reasoner. [3, 13] are based

on a specific reasoner KAON2 [8], which supports DL $\mathcal{SHIQ}$. The applicability of these approaches are restricted by the capability of the heavy-weight reasoner and hence the reasoning complexity can not be substantially reduced.

In our early work [11] we presented an approximate reasoning approach to reduce TBox reasoning in DL $\mathcal{R}$ to that in DL $\mathcal{EL}^+$. The reasoning complexity is reduced from 2EXPTIME-hard to PTIME and the soundness of results is preserved. In this paper, we extend this approach to support ABox reasoning in DL $\mathcal{RO}$, i.e. DL $\mathcal{SHO}$ plus role chains. Given an $\mathcal{RO}$ ontology, we first approximate it into an $\mathcal{EL}^{++}$ ontology with a complement table (CT) maintaining the complementary relations between named concepts, then extend the $\mathcal{EL}^{++}$ reasoning with additional completion rules to entail logical consequence, for both TBox and ABox. This approach is tractable and soundness-preserving.

The rest of this paper is organised as follows: in Sec. 2 we briefly introduce the DL $\mathcal{RO}$ and $\mathcal{EL}^{++}$, and discuss the technical challenge of existing approximate reasoning approaches. In Sec. 3 we present our approach for approximate ABox reasoning, particularly, we show how the ABox approximate reasoning should be combined with the TBox approximate reasoning. In Sec. 4 we present some preliminary evaluation of our approach and Sec. 5 concludes the paper.

2 Technical Motivations

In [11] we presented an approach to approximating $\mathcal{R}$ TBox to $\mathcal{EL}^+$ TBox with an additional complement table. We note that $\mathcal{EL}^{++}$, an extension of $\mathcal{EL}^+$ that supports singletons, is also tractable. Thus it is natural to allow the using of nominals in the original ontology. This leads to the DL $\mathcal{RO}$.

In order to motivate our investigation on syntactic approximation of $\mathcal{RO}$ ontologies to $\mathcal{EL}^{++}$ ontologies, this section first briefly introduces $\mathcal{RO}$ and $\mathcal{EL}^{++}$ and then illustrates the technical challenges in their ABox reasoning and approximation.

In $\mathcal{RO}$, concepts C, D can be inductively composed with the following constructs:

$$\top \mid \bot \mid A \mid C \sqcap D \mid \exists r.C \mid \{a\} \mid \neg C$$

where $\top$ is the top concept, $\bot$ the bottom concept, A atomic concept, n an integer number, a an individual and r an atomic role. Conventionally, $C \sqcup D$ and $\forall R.C$ are used to abbreviate $\neg(\neg C \sqcap \neg D)$ and $\neg\exists R.\neg C$, respectively. Note that $\{a_1, a_2, \ldots, a_n\}$ can be regarded as abbreviation of $\{a_1\} \sqcup \{a_2\} \sqcup \ldots \sqcup \{a_n\}$. Without loss of generality, in what follows, we assume all the concepts to be in their negation normal forms (NNF)[1] and use ~C to denote the NNF of $\neg C$. We also call $\top, \bot, A, \{a\}$ *basic concepts* because they are not composed by other concepts or roles. Given a KB Σ, we use CN_Σ (RN_Σ, IN_Σ) to denote the set of basic concepts (atomic roles, individuals) in Σ.

Target language $\mathcal{EL}^{++}$ supports

$$\top \mid \bot \mid A \mid C \sqcap D \mid \exists r.C \mid \{a\}.$$

[1] An $\mathcal{RO}$ concept is in NNF *iff* negation is applied only to atomic concepts and singletons. NNF of a given concept can be computed in linear time[4].

Both $\mathcal{RO}$ and $\mathcal{EL}^{++}$ support concept inclusions (CIs, e.g. $C \sqsubseteq D$), role inclusions (RIs, e.g. $r \sqsubseteq s$, $r_1 \circ \ldots \circ r_n \sqsubseteq s$), class assertions (e.g. $a : C$) and role assertions (e.g. $(a, b) : r$). If $C \sqsubseteq D$ and $D \sqsubseteq C$, we write $C \equiv D$. If C is non-atomic, $C \sqsubseteq D$ is a general concept inclusion (GCI). For more details about syntax and semantics of DLs, we refer the readers to [2]. Given a set of axioms Σ (a single axiom α), its signature, denoted by $Sig(\Sigma)$ ($Sig(\alpha)$) is the set of all the concept names (including $\top$ and $\bot$), role names and individual names appearing in Σ (α).

Traditionally in expressive and very expressive DLs, ABox reasoning is performed together with the TBox by the tableau algorithm [6]. The tableau algorithm [5] constructs a tableau (as a witness of a model of the ontology) as a graph in which each node x represents an individual and is labeled with a set of concepts it must satisfy, each edge $\langle x, y\rangle$ represents a pair of individuals satisfying a role that labels the edge.

Instance checking $\Sigma \models a : C$ is reduced to knowledge base consistence for the extended knowledge base $\Sigma' = \Sigma \cup \{a : \neg C\}$ [12]. To test this, a tableau is initialised with the concept and role assertions in Σ' and is then expanded by repeatedly applying the completion rules. Similar to other reasoning services, tableau-based instance checking has to deal with the non-determinism of GCI, which results in an exponential blowup of the search space.

Reasoning with $\mathcal{EL}^{++}$ is more efficient. [1] presents a set of TBox completion rules (Table 1) [2] to compute, given a normalised $\mathcal{EL}^{++}$ TBox $\mathcal{T}$, for each $A \in CN_{\mathcal{T}}$, a subsumer set $S(A) \subseteq CN_{\mathcal{T}}$ in which for each $B \in S(A)$, $\mathcal{T} \models A \sqsubseteq B$, and for each $r \in RN_{\mathcal{T}}$, a relation set $R(r) \subseteq CN_{\mathcal{T}} \times CN_{\mathcal{T}}$ in which for each $(A, B) \in R(r)$, $\mathcal{T} \models A \sqsubseteq \exists r.B$. These sets are initialised as: for each $A \in CN_{\mathcal{T}}$, $S(A) = \{A, \top\}$ and for each $r \in RN_{\mathcal{T}}$, $R(r) = \emptyset$. Reasoning with rules **R1-R8** is tractable.

Table 1. $\mathcal{EL}^{++}$ completion rules (no datatypes)

R1	If $A \in S(X)$, $A \sqsubseteq B \in \mathcal{T}$ and $B \notin S(X)$ then $S(X) := S(X) \cup \{B\}$
R2	If $A_1, A_2, \ldots, A_n \in S(X)$, $A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq B \in \mathcal{T}$ and $B \notin S(X)$ then $S(X) := S(X) \cup \{B\}$
R3	If $A \in S(X)$, $A \sqsubseteq \exists r.B \in \mathcal{T}$ and $(X, B) \notin R(r)$ then $R(r) := R(r) \cup \{(X, B)\}$
R4	If $(X, A) \in R(r)$ $A' \in S(A)$, $\exists r.A' \sqsubseteq B \in \mathcal{T}$ and $B \notin S(X)$ then $S(X) := S(X) \cup \{B\}$
R5	If $(X, A) \in R(r)$, $\bot \in S(A)$ and $\bot \notin S(X)$ then $S(X) := S(X) \cup \{\bot\}$
R6	If $\{a\} \in S(X) \cap S(A)$, $X \rightsquigarrow_R A$ and $S(A) \not\subseteq S(X)$ then $S(X) := S(X) \cup S(A)$
R7	If $(X, A) \in R(r)$, $r \sqsubseteq s \in \mathcal{T}$ and $(X, A) \notin R(s)$ then $R(s) := R(s) \cup \{(X, A)\}$
R8	If $(X, A) \in R(r_1)$, $(A, B) \in R(r_2)$, $r_1 \circ r_2 \sqsubseteq r_3 \in \mathcal{T}$, and $(X, B) \notin R(r_3)$ then $R(r_3) := R(r_3) \cup \{(X, B)\}$

[2] in **R6** $X \rightsquigarrow_R A$ iff there exists $C_1, \ldots, C_k \in CN_{\mathcal{T}}$ s.t. $C_1 = X$ or $C_1 = \{b\}$, $(C_j, C_{j+1}) \in R(r_j)$ for some $r_j \in RN_{\mathcal{T}} (1 \leq j \leq k)$ and $C_k = A$

When ABox $\mathcal{A}$ presents, an additional concept $C_{\mathcal{A}} := \bigsqcap_{a:C\in\mathcal{A}} \exists u.(\{a\} \sqcap C) \sqcap \bigsqcap_{(a,b):r\in\mathcal{A}} \exists u.(\{a\} \sqcap \exists r.\{b\})$, where u is a fresh role name, is introduced. To this end, instance checking $a : C$ can be reduced to subsumption checking $\{a\}\sqcap C_{\mathcal{A}} \sqsubseteq C$, which can be realised by **R1-R8**. However, this approach can not be directly applied on more expressive DLs.

To provide more scalable and efficient ABox reasoning service in expressive DLs, approximation approaches have been studied. However, most of these approaches heavily rely on existing reasoners. [14] presented approaches based on the idea of simplifying concept expressions in an ontology or a query to speed up the instance retrieval. The simplified ontology and query still needs to be processed by a heavy-weight reasoner, and the evaluation results showed that the number of subsumption tests can not always be reduced. *Semantic Approximation* [10] uses a heavy weighted reasoner to materialize the ontology and store in a database to speed up online query answering. But once the ABox changed, the entire procedure has to be performed again. [3, 13] present the `SCREECH` approach. It utilizes the KAON2 algorithm, which translates a $\mathcal{SHIQ}$ TBox into disjunctive datalog, and executes the rules together with a $\mathcal{SHIQ}$ ABox and a query by a datalog reasoning engine. By rewriting or eliminating all the disjunctive rules the data complexity can be reduced from coNP-complete of OWL DL to polynomial time. However this still relies on KAON2 to pre-translate the TBox. If the ontology is in a language beyond the capability of KAON2, e.g. $\mathcal{RO}$, this approach can not handle. Also, when the ABox contains complex concept expressions, this approach can not directly execute.

To sum up, tableau algorithms have difficulties to handle complex structured axioms; tractable DL algorithms can not support more expressive languages; while traditional approximation approaches still rely on existing DL reasoners. In what follows, we presented our approach which is motivated and inspired by these works, and show that it overcomes these difficulties.

3 The Approach

In this section, we first recall and extend the TBox approximation in [11] to support ontology approximation from $\mathcal{RO}$ to $\mathcal{EL}^{++}$. Then we discuss how the ABox can be reasoned internally and externally of the TBox. At the end, we discuss how the internal and external reasoning of ABox can possibly be integrated.

3.1 Approximate $\mathcal{RO}$ Ontologies to $\mathcal{EL}^{++}$

In approximation, we only consider concepts corresponding to the particular ontology in question. We use the notion *term* to refer to these "interesting" concept expressions. More precisely, a term is: (i) a concept expression in any axiom, or (ii) a singleton of any individual, or (iii) the complement of a term, or (iv) the syntactic sub-expression of a term. In order to represent terms that will be used in $\mathcal{EL}^{++}$ reasoning, we assign names to them.

Definition 1. ***(Name Assignment)*** *Given S a set of concept expressions, a* name assignment *fn is a function as for each $C \in S$, $fn(C) = C$ if C is a basic concept; otherwise, $fn(C)$ is a fresh name.*

Now we approximate an $\mathcal{RO}$ ontology to $\mathcal{EL}^{++}$ plus a complement table (CT). Its basic idea is to represent (non-$\mathcal{EL}^{++}$) terms with its name assignment:

Definition 2. ***($\mathcal{EL}^{++}_{\mathcal{C}}$ Transformation)*** *Given an $\mathcal{RO}$ Ontology $\mathcal{O} = (\mathcal{T}_{\mathcal{O}}, \mathcal{A}_{\mathcal{O}})$ and a name assignment fn, its $\mathcal{EL}^{++}_{\mathcal{C}}$* transformation *$A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O})$ is a triple $(\mathcal{T}, \mathcal{A}, CT)$ constructed as follows:*

1. *$\mathcal{T}, \mathcal{A}$ and CT are all initialised as $\emptyset$.*
2. *for each $C \sqsubseteq D$ ($C \equiv D$) in $\mathcal{T}$, $\mathcal{T} = \mathcal{T} \cup \{fn(C) \sqsubseteq fn(D)\}$ ($\mathcal{T} = \mathcal{T} \cup \{fn(C) \equiv fn(D)\}$).*
3. *for each $\beta \in RI_{\mathcal{T}}$, add β into $\mathcal{T}$.*
4. *for each $a : C \in \mathcal{A}$, $\mathcal{A} = \mathcal{A} \cup \{a : fn(C)\}$.*
5. *for each $(a, b) : r \in \mathcal{A}$, $\mathcal{A} = \mathcal{A} \cup \{(a, b) : r\}$.*
6. *for each term C in $\mathcal{O}$, $CT = CT \cup \{(fn(C), fn(\sim C))\}$, and*
   (a) *if C is the form $C_1 \sqcap \ldots \sqcap C_n$, then $\mathcal{T} = \mathcal{T} \cup \{fn(C) \equiv fn(C_1) \sqcap \ldots \sqcap fn(C_n)\}$,*
   (b) *if C is the form $\exists r.D$, then $\mathcal{T} = \mathcal{T} \cup \{fn(C) \equiv \exists r.fn(D)\}$,*
   (c) *otherwise $\mathcal{T} = \mathcal{T} \cup \{fn(C) \sqsubseteq \top\}$.*

Step 2 rewrites all the concept axioms; Step 3 preserves all the $\mathcal{EL}^{++}$ role axioms; Step 4 and 5 rewrite all the ABox axioms; Step 6 defines all the $\mathcal{EL}^{++}$ terms and constructs the complement table CT. We call this procedure an *$\mathcal{EL}^{++}_{\mathcal{C}}$ approximation*.The $\mathcal{EL}^{++}_{\mathcal{C}}$ approximation approximates an $\mathcal{RO}$ ontology into an $\mathcal{EL}^{++}$ ontology with a table maintaining the complements of all the basic concepts in linear time:

Proposition 1. ***($\mathcal{EL}^{++}_{\mathcal{C}}$ Approximation)*** *For an Ontology $\mathcal{O}$, let $A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O}) = (\mathcal{T}, \mathcal{A}, CT)$, we have: (1)$(\mathcal{T}, \mathcal{A})$ is an $\mathcal{EL}^{++}$ ontology; (2) $\mathcal{A}$ only contains basic concepts of $\mathcal{T}$; (3) for each $A \in CN_{\mathcal{T}}$, there exists $(A, B) \in CT$; (4) if $(A, B) \in CT$ then $A, B \in CN_{\mathcal{T}}$ and $(B, A) \in CT$.*

Proposition 2. *For any Ontology $\mathcal{O} = (\mathcal{T}_{\mathcal{O}}, \mathcal{A}_{\mathcal{O}})$ and $(\mathcal{T}, \mathcal{A}, CT)$ its $\mathcal{EL}^{++}_{\mathcal{C}}$ transformation, if $\mathcal{O}$ contains $n_{\mathcal{O}}$ terms, then $|\mathcal{T}| \leq n_{\mathcal{O}} + |\mathcal{T}_{\mathcal{O}}|$, $|\mathcal{A}| = |\mathcal{A}_{\mathcal{O}}|$ and $|CT| = n_{\mathcal{O}}$, where $|\mathcal{T}|(|\mathcal{A}|, |\mathcal{T}_{\mathcal{O}}|, |\mathcal{A}_{\mathcal{O}}|)$ is the number of axioms in $\mathcal{T}(\mathcal{A}, \mathcal{T}_{\mathcal{O}}, \mathcal{A}_{\mathcal{O}})$ and $|CT|$ is the number of pairs in CT.*

Given an $\mathcal{EL}^{++}_{\mathcal{C}}$ transformation $(\mathcal{T}, \mathcal{A}, CT)$, we normalise axioms of form $C \sqsubseteq D_1 \sqcap \ldots \sqcap D_n$ into $C \sqsubseteq D_1, \ldots, C \sqsubseteq D_n$, and recursively normalise role chain $r_1 \circ \ldots \circ r_n \sqsubseteq s$ with $n > 2$ into $r_1 \circ \ldots \circ r_{n-1} \sqsubseteq u$ and $u \sqsubseteq s$. This procedure can be done in linear time. In the following, we assume $\mathcal{T}$ to be always normalised. For convenience, we use a *complement function* $fc : CN_{\mathcal{T}} \mapsto CN_{\mathcal{T}}$ as: for each $A \in CN_{\mathcal{T}}$, $fc(A) = B$ such that $(A, B) \in CT$.

3.2 ABox Internalisation and Reasoning

Once we are able to approximate an $\mathcal{RO}$ ontology into $\mathcal{EL}^{++}$, it is straightforward to perform ABox reasoning by internalising the ABox into TBox. This can be done as in classical $\mathcal{EL}^{++}$ (cf. Sec. 2) by encoding the ABox as a concept. However this approach will introduce additional concept names in the normalisation phase, thus complicates the reasoning. Alternatively, we can do the following internalisation:

Definition 3. *($\mathcal{EL}^{++}_{\mathcal{C}}$ **ABox Internalisation**) Given an $\mathcal{RO}$ ontology $\mathcal{O}$, let $A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O}) = (\mathcal{T}', \mathcal{A}', CT')$, its $\mathcal{EL}^{++}_{\mathcal{C}}$* ABox internalisation *$AI(A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O}))$ is a triple $(\mathcal{T}, \emptyset, CT)$ constructed as follows:*

1. *$\mathcal{T}$ is initialised as $\mathcal{T}'$.*
2. *$CT = CT'$.*
3. *for each $a : C \in \mathcal{A}'$, $\mathcal{T} = \mathcal{T} \cup \{\{a\} \sqsubseteq C\}$.*
4. *for each $(a, b) : r \in \mathcal{A}'$, $\mathcal{T} = \mathcal{T} \cup \{\{a\} \sqsubseteq \exists r.\{b\}\}$.*

It's easy to show that such internalisation can be constructed in linear time and the triple $(\mathcal{T}, \emptyset, CT)$ still satisfy Proposition 1. Also, $\mathcal{T}$ is normalised if $\mathcal{T}'$ normalised. To this end, we reduce ABox reasoning to TBox reasoning on $\mathcal{T}$. To utilize the complementary relations in CT, we propose additional completion rules (Table 2) to $\mathcal{EL}^{++}$.

Table 2. Complement completion rules

R9	If $A, B \in S(X)$, $A = fc(B)$ and $\bot \notin S(X)$ then $S(X) := S(X) \cup \{\bot\}$
R10	If $A \in S(B)$ and $fc(B) \notin S(fc(A))$ then $S(fc(A)) := S(fc(A)) \cup \{fc(B)\}$
R11	If $A_1 \sqcap \ldots \sqcap A_i \sqcap \ldots \sqcap A_n \sqsubseteq \bot$, $A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_n \in S(X)$ and $fc(A_i) \notin S(X)$ then $S(X) := S(X) \cup \{fc(A_i)\}$

R9 realises axiom $A \sqcap \sim A \sqsubseteq \bot$. **R10** realises $A \sqsubseteq B \rightarrow \sim A \sqsubseteq \sim B$. **R11** builds up the relations between conjuncts of a conjunction, *e.g.* $A \sqcap B \sqsubseteq \bot$ implies $A \sqsubseteq \sim B$. The reasoning is tractable and soundness-preserving:

Theorem 1. ***(Complexity)** For any $\mathcal{EL}^{++}_{\mathcal{C}}$ internalisation $(\mathcal{T}, \emptyset, CT)$ ($\mathcal{T}$ normalised), TBox reasoning by **R1-R11** will terminate in polynomial time w.r.t. $|CN_{\mathcal{T}}|+|RN_{\mathcal{T}}|$.*

Theorem 2. ***(Concept Subsumption Checking)** Given an $\mathcal{RO}$ ontology $\mathcal{O} = (\mathcal{T}_{\mathcal{O}}, \mathcal{A}_{\mathcal{O}})$, its vocabulary $V_{\mathcal{O}}$ and $AI(A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O})) = (\mathcal{T}, \emptyset, CT)$, for any two concepts C and D constructed from $V_{\mathcal{O}}$, if $AI(A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\{C \sqsubseteq \top, D \sqsubseteq \top\})) = (\mathcal{T}', \emptyset, CT')$, then $\mathcal{O} \models C \sqsubseteq D$ if $fn(D) \in S(fn(C))$ can be computed by rules **R1-R11** on $(\mathcal{T} \cup \mathcal{T}', \emptyset, CT \cup CT')$.*

Concering ABox reasoning, this indicates that, $\mathcal{O} \models a : C$ if $fn(C) \in S(\{a\})$ can be computed. And $\mathcal{O} \models (a, b) : r$ if $fn(\exists r.\{b\}) \in S(\{a\})$ can be computed. When C is a term of $\mathcal{O}$, such computation can be performed directly on $(\mathcal{T}, \emptyset, CT)$.

3.3 TBox-irrelevant ABox Completion

The ABox internalisation absorbs the entire ABox into the TBox and reduce ABox reasoning to TBox reasoning. However, this approach has its limitation: (i) according to Theorem 1 when a large amount of individuals present, more computations are needed (individuals are converted into singletons); (ii) it yields some results useless in TBox and ABox reasoning. For example, $A \sqsubseteq \exists r.B, x : A$ will yield $(\{x\}, B) \in R(r)$ by **R3**. To optimize the performance we separate the reasoning of TBox and ABox.

We start from a simpler case, in which the approximated TBox contains no nominal. In this case, the ABox reasoning has no effect on the TBox reasoning, which can thus be pre-computed.

Given $A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O}) = (\mathcal{T}, \mathcal{A}, CT)$, after TBox reasoning of **R1-R11**, we present ABox completion rules (Table 3) to compute, for each $a \in IN_{\mathcal{A}}$, a class set $C(a) \subseteq CN_{\mathcal{T}} \cup CN_{\mathcal{A}}$ in which for each $A \in C(a)$, $\mathcal{T}, \mathcal{A} \models a : C$, and for each $r \in RN_{\mathcal{T}} \cup RN_{\mathcal{A}}$, a role set $RO(r) \subseteq IN_{\mathcal{A}} \times IN_{\mathcal{A}}$ in which for each $(a, b) \in RO(r)$, $\mathcal{T}, \mathcal{A} \models (a, b) : r$. These sets are initialised as: $A \in C(a)$ if $a : A \in \mathcal{A}$, $(a, b) \in RO(r)$ if $(a, b) : r \in \mathcal{A}$.

Table 3. TBox-independent $\mathcal{EL}^{++}_{\mathcal{C}}$ ABox completion rules (no datatypes)

AR1	If $A \in C(x)$, $B \in S(A)$ and $B \notin C(x)$ then $C(x) := C(x) \cup \{B\}$
AR2	If $A_1, A_2, \ldots, A_n \in C(x)$, $A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq B \in \mathcal{T}$ and $B \notin C(x)$ then $C(x) := C(x) \cup \{B\}$
AR3	If $(x, y) \in RO(r)$ $A \in C(y)$, $\exists r.A \sqsubseteq B \in \mathcal{T}$ and $B \notin C(x)$ then $C(x) := C(x) \cup \{B\}$
AR4	If $(x, y) \in RO(r)$, $\bot \in C(y)$ and $\bot \notin C(x)$ then $C(x) := C(x) \cup \{\bot\}$
AR5	If $(x, y) \in RO(r)$, $r \sqsubseteq s \in \mathcal{T}$ and $(x, y) \notin RO(s)$ then $RO(s) := RO(s) \cup \{(x, y)\}$
AR6	If $(x, y) \in RO(r_1)$, $(y, z) \in RO(r_2)$, $r_1 \circ r_2 \sqsubseteq r_3 \in \mathcal{T}$, and $(x, y) \notin RO(r_3)$ then $RO(r_3) := RO(r_3) \cup \{(x, z)\}$
AR7	If $A, B \in C(x)$, $A = fc(B)$ and $\bot \notin C(x)$ then $C(x) := C(x) \cup \{\bot\}$
AR8	If $A \in C(x)$, $fc(A) \in S(fc(B))$ and $B \notin C(x)$ then $C(x) := C(x) \cup \{B\}$
AR9	If $A_1 \sqcap \ldots \sqcap A_i \sqcap \ldots \sqcap A_n \sqsubseteq \bot$, $A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_n \in C(x)$ and $fc(A_i) \notin C(x)$ then $C(x) := C(x) \cup \{fc(A_i)\}$

It's easy to see that **AR1-AR6** are for $\mathcal{EL}^{++}$ while **AR7-AR9** are for $\mathcal{EL}^{++}_{\mathcal{C}}$ transformation. More precisely, **AR1-AR2** are analogues to **R1-R2**, **AR3-AR4** analogues to **R4-R5**, **AR5-AR6** analogues to **R7-R8**, **AR7-AR9** analogues to **R9-R11**. This indicates that similar algorithms can be applied and tractability and soundness are preserved. The fewer completion rules shall result in more efficient inference. Furthermore, **AR5-AR6** can be processed ahead of the other rules because other rules will not

generate any RO sets elements. This should further improve the efficiency and scalability. This TBox-ABox-separated approach should have the same results as the ABox-internalised approach.

3.4 Integrated ABox Approximation

ABox completion presented in Sec.3.3 has a restriction that the approximated TBox should contain no nominal. For example, $a : A, a : B, B \sqsubseteq \{b\}$, with internalisation we will infer $b : A$, which can not be computed by **AR1-AR9**. The question arises that whether the internalisation and ABox completion approach can be combined. In this section we discuss the possibility of relaxing the nominal-free restriction.

Our basic idea is to partition the approximated ABox $\mathcal{A}$ into two disjoint-union subsets $\mathcal{A}_I$ and $\mathcal{A}_E$ so that $\mathcal{A}_I$ should be internalised into the approximated TBox $\mathcal{T}$ to obtain an extended TBox $\mathcal{T}_I$ that can be classified by **R1-R11**, while $\mathcal{A}_E$ can be completed by **AR1-AR9** after TBox reasoning over $\mathcal{T}_I$. There can be different partitioning strategies. In what follows, we present a reachability-based partitioning. In general, any ABox axiom that contains concept or individual name that is directly or indirectly reachable to some nominal, should be internalised.

Definition 4. ***(Nominal-reachable Signature)*** *Let $A_{fn,\mathcal{EL}_c^{++}}(\mathcal{O}) = (\mathcal{T}, \mathcal{A}, CT)$, its* nominal-reachable signature $Sig_{NR}(A_{fn,\mathcal{EL}_c^{++}}(\mathcal{O}))$ *($Sig_{NR}(\mathcal{O})$ for short) is a minimal subset of $Sig(\mathcal{T}) \cup Sig(\mathcal{A})$ having the following properties:*

1. *for any $a \in IN_{\mathcal{T}}$, we have $a \in Sig_{NR}(\mathcal{O})$.*
2. *for any $A \in CN_{\mathcal{T}}$, we have $A, fc(A) \in Sig_{NR}(\mathcal{O})$ if there exists $C \sqsubseteq D \in \mathcal{T}$ s.t. $A \in Sig(C)$ and $Sig(D) \cap Sig_{NR}(\mathcal{O}) \neq \emptyset$.*
3. *for any $a \in IN_{\mathcal{T}}$, we have $a \in Sig_{NR}(\mathcal{O})$ if there exists $a : A \in \mathcal{A}$ ($(a, b) : r \in \mathcal{A}$) s.t. $A \in Sig_{NR}(\mathcal{O})$ ($b \in Sig_{NR}(\mathcal{O})$).*
4. *for any $A \in CN_{\mathcal{A}}$, $A, fc(A) \in Sig_{NR}(\mathcal{O})$ if there exists $a : A \in \mathcal{A}$ s.t. $a \in Sig_{NR}(\mathcal{O})$.*

Then the ABox can be partitioned into two parts, one's signature is nominal-reachable, the other's not. The nominal-reachable part of the ABox should be internalised into the TBox:

Definition 5. ***(Reachability-based Internalisation)*** *Given an $\mathcal{RO}$ ontology $\mathcal{O}$ and $A_{fn,\mathcal{EL}_c^{++}}(\mathcal{O}) = (\mathcal{T}', \mathcal{A}', CT')$, its* reachability-based internalisation $RbI(A_{fn,\mathcal{EL}_c^{++}}(\mathcal{O}))$ *is a triple $(\mathcal{T}, \mathcal{A}, CT)$ constructed as follows:*

1. *$CT = CT'$.*
2. *let $\mathcal{A}_I = \{\alpha \in \mathcal{A}' | Sig(\alpha) \cap Sig_{NR}(\mathcal{O}) \neq \emptyset\}$, and $AI((\mathcal{T}', \mathcal{A}_I, CT)) = (\mathcal{T}_I, \emptyset, CT)$, then*
   - *$\mathcal{A} = \mathcal{A}' \setminus \mathcal{A}_I$.*
   - *$\mathcal{T} = \mathcal{T}_I$.*

For example, let $\mathcal{T}_1 = \{B \sqsubseteq \{b\}\}$ and $\mathcal{A}_2 = \{a : A, a : B\}$, then both of the ABox axioms should be internalised. Similarly, let $\mathcal{T}_2 = \{A \sqsubseteq \{a\}, \exists r.B \sqsubseteq C\}$ and $\mathcal{A}_2 = \{b : B, (a, b) : r\}$, then the entire ABox should be internalised as well so that $A \sqsubseteq C$ can be inferred. Given $RbI(A_{fn,\mathcal{EL}^{++}_{\mathcal{C}}}(\mathcal{O})) = (\mathcal{T}, \mathcal{A}, CT)$, the reasoning can be performed as follows:

1. classify $(\mathcal{T}, \emptyset, CT)$ by **R1-R11**.
2. extend $\mathcal{A}$ as $\mathcal{A} = \mathcal{A} \cup \{a : A | A \in S(\{a\})\} \cup \{(a, b) : r | (\{a\}, \{b\}) \in R(r)\}$.
3. reason $(\mathcal{T}, \mathcal{A}, CT)$ by **AR1-AR9**.

This integration approach of internalisation and ABox completion should have the same results as the the internalisation approach. The tractability of the reasoning is also preserved.

There could be other way of partitioning the ABox. The advantage of our proposal is that it is purely syntactic thus can be performed efficiently.

4 Evaluation

We implemented the internalisation approach (cf. Sec.3.2) and the ABox completion approach (cf. Sec.3.3) separately in our REL reasoner. In our experiments, `REL_int` system implemented the approximation (Sec.3.1) and the internalisation approach; `REL_ext` implements the approximation and the ABox completion approach. To evaluate their performance in practice, we compared with mainstream reasoners Pellet 2.0.1 and FaCT++ 1.3.0.1. All experiments were conducted in an environment of Windows XP SP3 with 2.66 GHz CPU and 1G RAM allocated to JVM 1.6.0.07.

Our test suite consists of several real world or benchmark ontologies with various size and expressivity of TBox and ABox. The VICODI [3] ontology is developed to represent the history of Europe. SEMINTEC [4] ontology is developed for semantic web mining. WINE [5] ontology is an OWL-DL show case ontology, designed to exploit the expressive power of OWL-DL. LUBM (Lehigh University Benchmark) [6] is a benchmark for OWL-Lite query answering. VICODI and SEMINTEC have relatively simple TBox but large ABox. WINE has a rather complex TBox and a moderate ABox. LUBM contains a moderately complex TBox and its ABox can be generated as large as needed. In our evaluation, we generated 1 university. Only WINE has nominals. We also converted datatype properties into object properties. As for WINE ontology, the expressivity is beyond $\mathcal{RO}$, but REL directly approximate those constructs, e.g. cardinality restrictions, inverse roles, with names.

For each ontology, we retrieve the types of all the individuals and the relations between all pairs of individuals. Each reasoner was given 10 minutes on each task. Recall of REL is calculated against the others. Thus the time shown in our evaluation includes approximation time (for REL), reasoning time, type and relation retrieval and

[3] `http://www.vicodi.org/about.htm`
[4] `http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm`
[5] `http://www.w3.org/TR/owl-guide/wine.rdf`
[6] `http://swat.cse.lehigh.edu/projects/lubm/`

counting time. Time unit is second. `REL_ext` is tested for all the ontologies. `REL_int` is tested only for WINE ontology as the others contain no nominal. The results are presented in Table 4, in which "e/o" indicates that the reasoner exited with an error; "t/o"indicates that the reasoner failed to finish the task in 10 minutes.

Table 4. Evaluation Results

Ontology	retrieval	Pellet	FaCT++	REL_ext		REL_int	
				time	recall	time	recall
VICODI	concept	7.828	17.515	3.86	100%	-	-
	role	9.656	t/o	3.828	100%	-	-
SEMINTEC	concept	4.5	4.422	2.484	100%	-	-
	role	7.062	t/o	2.485	100%	-	-
WINE	concept	17.813	e/o	1.266	85.1%	1.641	98.2%
	role	26.9	e/o	1.266	40.0%	1.453	91.5%
LUBM$\times$1	concept	10.937	17.359	8.531	100%	-	-
	role	26.891	t/o	8.625	100%	-	-

As we can see from Tabel 4. REL is (2 times to more than 20 times) faster than all the other reasoners on all the ontologies, which indicates an improvement on the efficiency of reasoning. For simple ontologies such as VICODI, SEMINTEC and LUBM, the advantage of REL is not significant. While when the ontology has a relatively complex TBox, especially when the TBox and ABox are connected, the benefits of approximate reasoning become substantial. Note that for these two tasks, the time of REL was almost the same: because our completion rules compute the instances of all the atomic concepts and atomic roles together.

Concerning the completeness, when the ontology is simple, the recall of `REL_ext` is 100%. When the ontology TBox gets complex and contains nominals. Separate reasoning of TBox and ABox becomes not satisfying. By internalising the ABox into TBox the recall was significantly improved. It's interesting to see that the time of `REL_int` was not much longer than `REL_ext` on the WINE. That is because WINE ontology contains about 208 individuals, which is not a large number. It will be necessary to implement the integrated solution as we discussed in Sec.3.4, when the ABox goes large and TBox contains nominals.

To sum up, the evaluation showed that even naive implementations of our approach can provide efficient and rather complete ABox reasoning services. Particularly, when the ontology is complex and large, the efficiency can still be retained while the completeness is not sacrificed too much.

5 Conclusion & Future Work

In this paper, we presented an approximate reasoning approach to address the issue of ABox reasoning over ontologies of expressive DL $\mathcal{RO}$, a fragment of OWL2-DL supporting $\mathcal{ALC}$ GCIs, nominals and role chains. Our approach first approximates an

$\mathcal{RO}$ ontology to an $\mathcal{EL}^{++}$ ontology plus a complement table (CT) maintaining the complementary relations between named concepts (including $\top$ and $\bot$) and singletons. Then we presented an internalisation approach to reducing ABox reasoning into TBox reasoning, and presented additional completion rules to utilize the CT. For ontology with no nominal in TBox, we presented an optimized ABox Completion approach. We further discussed the possibility of combining the two approaches and presented one of the possible solution.

Our approximate reasoning strategy is soundness-preserving and can be realised in PTIME. Although we don't guarantee completeness, our preliminary evaluation showed that naive implementations of our approach can improve the efficiency of reasoning over real world and benchmark ontologies, while maintaining a high recall.

In the future, we would like to further improve the completeness by exploiting more reasoning patterns, to future improve the scalability by combining with relational databases, to further improve the efficiency by optimising the implementations. The lack of expressive benchmark in our evaluation also motivates us creating our own benchmarks.

References

1. F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ Envelope. In *Proceedings IJCAI-05*, 2005.
2. Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
3. Pascal Hitzler and Denny Vrandecic. Resolution-Based Approximate Reasoning for OWL DL. In *Proceedings of ISWC 2005*, 2005.
4. Bernhard Hollunder, Werner Nutt, and Manfred Schmidt-Schau. Subsumption Algorithms for Concept Description Languages. In *ECAI-90*, pages 348–353. Pitman Publishing, 1990.
5. Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical Reasoning for Very Expressive Description Logics. *Logic Journal of the IGPL*, 8:2000, 2000.
6. Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Reasoning with individuals for the description logic shiq. In *CADE-17*, 2000.
7. Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Reducing SHIQ-Description Logic to Disjunctive Datalog Programs. In *KR 2004*, pages 152–162, 2004.
8. B. Motik. Practical DL Reasoning over Large ABoxes with KAON2. 2006.
9. Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. OWL 2 Web Ontology Language: Profiles. W3c working draft, W3C, October 2008.
10. Jeff Z. Pan and Edward Thomas. Approximating OWL-DL Ontologies. In *AAAI-2007*, pages 1434–1439, 2007.
11. Yuan Ren, Jeff Z. Pan, and Yuting Zhao. Soundness Preserving Approximation for TBox Reasoning in R. In *Description Logics 2009*, 2009.
12. A. Schaerf. Reasoning With Individuals in Concept Languages. *Data and Knowledge Engineering*, 13(2):141–176, 1994.
13. Tuvshintur Tserendorj, Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. Approximate OWL-Reasoning with Screech. In *Proceedings of RR 08*, 2008.
14. Holger Wache, Perry Groot, and Heiner Stuckenschmidt. Scalable Instance Retrieval for the Semantic Web by Approximation. In *WISE Workshops-05*, 2005.