Module Extraction for Efficient Object Query over Ontologies with Large ABoxes

Jia Xu[a,*], Patrick Shironoshita[a], Ubbo Visser[b], Nigel John[a], Mansur Kabuka[a]

[a] *Department of Electrical and Computer Engineering,*
[b] *Department of Computer Science,*
University of Miami, Coral Gables, FL 33146, USA

Abstract

The extraction of logically-independent fragments out of an ontology ABox can be useful for solving the tractability problem of querying ontologies with large ABoxes. In this paper, we propose a formal definition of an ABox module, such that it guarantees complete preservation of facts about a given set of individuals, and thus can be reasoned independently w.r.t. the ontology TBox. With ABox modules of this type, isolated or distributed (parallel) ABox reasoning becomes feasible, and more efficient data retrieval from ontology ABoxes can be attained. To compute such an ABox module, we present a theoretical approach and also an approximation for $\mathcal{SHIQ}$ ontologies. Evaluation of the module approximation on different types of ontologies shows that, on average, extracted ABox modules are significantly smaller than the entire ABox, and the time for ontology reasoning based on ABox modules can be improved significantly.

Keywords:
Ontology, Reasoning, ABox Module, $\mathcal{SHIQ}$

1. Introduction

Description logics (DLs), as a decidable fragment of first-order logic, are a family of logic based formalisms for knowledge representation, and the mathematical underpinning for modern ontology languages such as OWL (Bechhofer et al., 2004; Horrocks et al., 2003) and OWL 2 (Cuenca Grau et al., 2008). A DL ontology (or knowledge base) consists of a terminological part (TBox) $\mathcal{T}$ that defines terminologies such as concepts and roles of a given domain, and an assertional part (ABox) $\mathcal{A}$ that describes instances of the conceptual knowledge. Similar to a database, the ontology TBox usually represents the data schema, and the ontology ABox corresponds to the actual data set.

In recent years, DL ontologies have been increasingly applied in the development of DL-based information systems in diverse areas, including biomedical research (Demir

*Corresponding author.
Email addresses: j.xu11@umiami.edu (Jia Xu), patrick@infotechsoft.com (Patrick Shironoshita), visser@cs.miami.edu (Ubbo Visser), nigel.john@miami.edu (Nigel John), m.kabuka@miami.edu (Mansur Kabuka)

et al., 2010; Visser et al., 2011), health care (Bhatt et al., 2009; Iqbal et al., 2011), decision support (Haghighi et al., 2012; Lee et al., 2008), and many others; see (Horrocks, 2008) for a review of existing applications. Most of these DL applications involve intensive querying of the underlying knowledge data that requires reasoning over ontologies.

Standard DL reasoning services include *subsumption testing* (i.e. testing if one concept is more general than the other), and *instance checking* (i.e. checking if an individual is one instance of a given concept). The former is considered TBox reasoning, and the latter is considered ABox reasoning as well as the central reasoning task for information retrieval from ontology ABoxes (Schaerf, 1994). These reasoning services are highly complex tasks, especially for ontologies with an expressive DL, and can have up to exponential worst-case complexity w.r.t. the size of the ontology (Baader et al., 2007; Tobies, 2001). For example, instance checking in a $\mathcal{SHIQ}$ ontology is in EXPTIME (Tobies, 2001). Consequently, query-answering over ontologies that rely on instance checking (Horrocks and Tessaris, 2000) can also have high computational complexity (Calvanese et al., 2007; Glimm et al., 2008; Ortiz et al., 2008).

Existing systems that provide standard DL reasoning services include HermiT (Motik et al., 2007), Pellet (Sirin et al., 2007), FaCT++ (Tsarkov and Horrocks, 2006), and Racer (Haarslev and Möller, 2001). They are all based on the (*hyper*)*tableau* algorithms that are proven sound and complete. Despite highly optimized implementations of reasoning algorithms in these systems, they are still confronted with the scalability problem in practical DL applications, where although the TBox could be relatively small and manageable, the ABox tends to be extremely large (e.g. in a setting of semantic webs), which can result in severe tractability problems (Horrocks et al., 2004; Motik and Sattler, 2006).

Due to the central role that is played by ontology querying in data-intensive DL-applications, various approaches have been proposed to cope with this ABox reasoning problem. For example, there are optimization strategies proposed by (Haarslev and Möller, 2002) for instance retrieval, and also techniques developed for ABox reduction (Fokoue et al., 2006) and partition (Guo and Heflin, 2006; Wandelt and Möller, 2012). (Hustadt et al., 2004) proposed to reduce DL ontology into a disjunctive datalog program, so that existing techniques for querying deductive databases can be reused; and (Grosof et al., 2003), (Motik et al., 2005), and (Royer and Quantz, 1993) have suggested the combination of ontology reasoning with inference rules.

In this paper, conceiving that a large ABox may consist of data with great diversity and isolation, we explore the modularity of an ontology ABox and expect ABox reasoning to be optimized by utilizing logical properties of ABox modules. Analogous to modularity defined for the ontology TBox (Cuenca Grau et al., 2007a,b), the notion of modularity for the ABox proposed in this paper is also based on the semantics and implications of ontologies; and an ABox module should be a *closure* or *encapsulation* of logical implications for a given set of individuals (that is, it should capture all facts, both explicit and implicit, of the given entities), such that each module can be reasoned *independently* w.r.t. $\mathcal{T}$.

This property of an ABox module is desired for efficient ABox reasoning under situations, either when querying information of a particular individual, such as instance checking (i.e. test if $\mathcal{K} \models C(a)$) or arbitrary property assertion checking (i.e. test if $\mathcal{K} \models R(a, b)$), or when performing instance retrieval and answering (conjunctive) queries over ontologies (Horrocks and Tessaris, 2000; Glimm et al., 2008). For the first case,

an independent reasoning on the ABox module for that particular individual will be sufficient, while for the latter, the property of ABox modules would allow the ABox reasoning to be parallelized, and thus able to take advantage of existing parallel-processing frameworks, such as MapReduce (Dean and Ghemawat, 2008).

For illustration, consider a real-world ontology that models and stores massive biomedical data in an ABox, and a query-answering system that is based on this ontology. A biomedical researcher may want to obtain information of a particular gene instance, say `CHRNA4`, to see if it is involved in any case of diseases, and in what manner. To answer such queries submitted to this ontology-based system, a reasoning procedure is normally invoked and applied to the ontology. Note however, given the fact that in this case the entire ontology ABox is extremely large, complete reasoning can be prohibitively expensive (Glimm et al., 2008; Ortiz et al., 2008), and it may take a lengthy period to reach any conclusions. Therefore, it would be preferable to perform reasoning on a smaller subject-related (in this case, `CHRNA4`-related) module. In particular, this `CHRNA4`-related ABox module should be a closure of all facts about the individual `CHRNA4`, so that reasoning on this module w.r.t. to $\mathcal{T}$ can achieve the same conclusions about `CHRNA4` as if the reasoning would be applied to the entire ontology. In addition, the reasoning time should be significantly decreased, provided the module is precise and much smaller than the entire ABox. Another case of querying over this biomedical ontology could be to retrieve all genes in the ABox that belong to the same pathological class of a certain disease. Answering such queries requires to perform instance retrieval, and a simple strategy based on the ABox module here is to partition the ABox based on groups of individuals and to distribute each independent reasoning task into a cluster of computers.

Our main contributions in this paper are summarized as follows:

1. To capture the notion of ABox modularity, we provide a set of formal definitions about the ABox module, specifying logical properties of an ABox module such that it guarantees preservation of sound and complete facts of a given set of individuals, and such that it can be independently reasoned over w.r.t. the TBox;
2. To extract such an ABox module, we develop a theoretical approach for DL $\mathcal{SHIQ}$ based on the idea of *module-essential* assertions. This approach gives an exposition for the problem of ABox-module extraction from a theoretical perspective, allowing a better understanding of the problem, and providing strategies that can be built into a general framework for computation of ABox modules;
3. To cope with the complexity for checking module-essentiality using a DL-reasoner and to make the techniques applicable to practical DL applications, we also present a simple and tractable syntactic approximation.

Additionally, we present an evaluation of our approximation on different ontologies with large ABoxes, including those generated by existing benchmark tools and realistic ones that are used in some biomedical projects. We also present and compare several simple optimization techniques for our approximation. And finally, we show the efficiency of ontology reasoning that can be achieved when using the ABox modules.

2. Preliminaries

2.1. Description Logic $\mathcal{SHIQ}$

The Description Logic $\mathcal{SHIQ}$ is extended from the well-known logic $\mathcal{ALC}$ (Schmidt-Schauß and Smolka, 1991), with added supports for role hierarchies, inverse roles, transitive roles, and qualified number restrictions (Horrocks et al., 2000). A $\mathcal{SHIQ}$ ontology defines a set $\mathbf{R}$ of role names, a set $\mathbf{I}$ of individual names, and a set $\mathbf{C}$ of class (or concept) names.

Definition 2.1 ($\mathcal{SHIQ}$**-Role).** *A $\mathcal{SHIQ}$-role can be an atomic (named) role $R \in \mathbf{R}$, or an inverse role R^- with $R \in \mathbf{R}$. The complete role set in a $\mathcal{SHIQ}$ ontology is denoted $\boldsymbol{R}^*=\boldsymbol{R} \cup \{R^- | R \in \boldsymbol{R}\}$. To avoid R^{--}, the function Inv$(\cdot)$ is defined, such that Inv$(R)=R^-$ and Inv$(R^-)=R$.*

A role hierarchy H_R can be defined in an ontology by a set of role inclusion axioms, each of which is expressed in the form of $R_1 \sqsubseteq R_2$, with $R_1, R_2 \in \mathbf{R}^*$. We call R_1 a *subrole* of R_2, if $R_1 \sqsubseteq R_2 \in H_R$ or if there exist $S_1, \ldots, S_n \in \mathbf{R}^*$ with $R_1 \sqsubseteq S_1, S_1 \sqsubseteq S_2, \ldots, S_n \sqsubseteq R_2 \in H_R$. Here, $\sqsubseteq$ is reflexive and transitive.

A role $R \in \mathbf{R}$ is *transitive*, denoted Trans(R), if $R \circ R \sqsubseteq R$ or Inv$(R) \circ$ Inv$(R) \sqsubseteq$ Inv(R). Finally, a role is called *simple* if it is neither transitive nor has any transitive subroles (Baader et al., 2007; Horrocks et al., 2000).

Definition 2.2 ($\mathcal{SHIQ}$**-Class).** *A $\mathcal{SHIQ}$-class is either an atomic (named) class or a complex class that can be defined using the following constructors recursively*

$$C, D ::= A | \neg C | C \sqcap D | C \sqcup D | \forall R.C | \exists R.C | \leq nS.C | \geq nS.C$$

where A is an atomic class, $R, S \in \boldsymbol{R}^*$ with S being simple, and n is a non-negative integer. The universal concept $\top$ and the bottom concept $\bot$ can be viewed as $A \sqcup \neg A$ and $A \sqcap \neg A$ respectively.

Definition 2.3 ($\mathcal{SHIQ}$ **Ontology).** *A $\mathcal{SHIQ}$ ontology is a tuple, denoted $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is the terminology representing general knowledge of a specific domain, and $\mathcal{A}$ is the assertional knowledge representing a particular state of the terminology.*

The terminology $\mathcal{T}$ of ontology $\mathcal{K}$ is the disjoint union of a finite set of role inclusion axioms (i.e. $R_1 \sqsubseteq R_2$) and a set of concept inclusion axioms in the form of $C \equiv D$ and $C \sqsubseteq D$, where C and D are arbitrary $\mathcal{SHIQ}$-classes. Statements $C \sqsubseteq D$ are called *general concept inclusion* axioms (GCIs), and $C \equiv D$ can be trivially converted into two GCIs as $C \sqsubseteq D$ and $D \sqsubseteq C$.

The assertional part $\mathcal{A}$ of $\mathcal{K}$ is also known as a $\mathcal{SHIQ}$-ABox, consisting of a set of assertions (facts) about individuals, in the form of

$C(a)$	class assertion
$R(a, b)$	role or property assertion
$a \not\approx b$	inequality assertion

where C is a $\mathcal{SHIQ}$ class, $R \in \mathbf{R}^*$, and $a, b \in \mathbf{I}$.

Note that explicit assertion of $a \not\approx b$ is supported in $\mathcal{SHIQ}$, while, conversely, explicit assertion of *equality*, i.e. $a \approx b$, is not supported, since its realization relies on equivalence between nominals (Baader et al., 2007; Hitzler et al., 2009), i.e. $\{a\} \equiv \{b\}$, which is illegal in $\mathcal{SHIQ}$.

Definition 2.4 ($\mathcal{SHIQ}$ **Semantics).** *The meaning of an ontology is given by an interpretation denoted* $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$*, with* $\Delta^{\mathcal{I}}$ *referred to as a non-empty domain and* $.^{\mathcal{I}}$ *referred to as an interpretation function. This interpretation function* $.^{\mathcal{I}}$ *maps:*

1. *every atomic class* $A \in \boldsymbol{C}$ *to a set* $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$*,*
2. *every individual* $a \in \boldsymbol{I}$ *to an element* $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ *and,*
3. *every role* $R \in \boldsymbol{R}$ *to a binary relation on the domain, i.e.* $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$*.*

Interpretation for other classes and inverse roles are given below:

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
\bot^{\mathcal{I}} &= \emptyset \\
\neg C^{\mathcal{I}} &= \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}} \\
(R^{-})^{\mathcal{I}} &= \{(y,x) \mid (x,y) \in R^{\mathcal{I}}\} \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &= \{x \mid \exists y.(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\forall R.C)^{\mathcal{I}} &= \{x \mid \forall y.(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \\
(\leq nR.C)^{\mathcal{I}} &= \{x \mid |\{y|(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}| \leq n\} \\
(\geq nR.C)^{\mathcal{I}} &= \{x \mid |\{y|(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}| \geq n\}
\end{aligned}$$

where $|\cdot|$ *represents the cardinality of a given set.*

An interpretation $\mathcal{I}$ *satisfies* an axiom $\alpha : C \sqsubseteq D$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Such interpretation is called a *model* of axiom α. An interpretation $\mathcal{I}$ satisfies an axiom or assertion α:

$$\begin{aligned}
R_1 \sqsubseteq R_2 \quad & \textit{iff} \quad R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}} \\
C(a) \quad & \textit{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}} \\
R(a,b) \quad & \textit{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}} \\
a \not\approx b \quad & \textit{iff} \quad a^{\mathcal{I}} \not\approx b^{\mathcal{I}}
\end{aligned}$$

For an ontology $\mathcal{K}$, if an interpretation $\mathcal{I}$ satisfies every axiom in $\mathcal{K}$, $\mathcal{I}$ is a *model* of $\mathcal{K}$, written $\mathcal{I} \models K$. In turn, ontology $\mathcal{K}$ is said *satisfiable* or *consistent* if it has at least one model; otherwise, it is *unsatisfiable* or *inconsistent*, and there exists at least one contradiction in $\mathcal{K}$.

Definition 2.5 (**Logic Entailment).** *Given an ontology* $\mathcal{K}$ *and an axiom* α*,* α *is called a logic entailment of* $\mathcal{K}$*, denoted* $\mathcal{K} \models \alpha$*, if* α *is satisfied in every model of* $\mathcal{K}$*.*

Definition 2.6 (**Instance checking).** *Given an ontology* $\mathcal{K}$*, a* $\mathcal{SHIQ}$*-class* C *and an individual* $a \in \mathbf{I}$*, instance checking is defined as testing whether* $\mathcal{K} \models C(a)$ *holds.*

Notice that instance checking is considered the central reasoning task for information retrieval from ontology ABoxes and a basic tool for more complex reasoning services (Schaerf, 1994). Instance checking can also be viewed as "*classification*" of an individual, that is, checking if an individual can be classified as a member of some defined DL class.

In most logic-based approaches, realizations of reasoning services are based on a so-called *refutation-style proof* (Baader et al., 2007; Horrocks and Patel-Schneider, 2004), that is, to convert a inference problem to the test of ontology satisfiability. For example,

to decide if $\mathcal{K} \models C(a)$, one can check if $\mathcal{K} \cup \{\neg C(a)\} \not\models \bot$ instead, and the answer is *yes* iff $\mathcal{K} \cup \{\neg C(a)\}$ is unsatisfiable, and the answer is *no* otherwise.

Ontology reasoning algorithm in current well-known systems (e.g. Pellet, HermiT, FaCT++, and Racer.) are all based on (hyper)tableau algorithms (Haarslev and Möller, 2001; Motik et al., 2007; Sirin et al., 2007; Tsarkov and Horrocks, 2006) that try to build a universal model for the given ontology based on a set of tableau expansion rules. For details of tableau expansion rules and description of a standard tableau algorithm for $\mathcal{SHIQ}$, we refer readers to the work in (Horrocks et al., 2000).

2.2. Other Definitions

We adopt notations from tableaux (Horrocks et al., 2000) for referring to individuals in $R(a, b)$, such that a is called an *R-predecessor* of b, and b is an *R-successor* (or R^-*-predecessor*) of a. If b is an *R-successor* of a, b is also an *R-neighbor* of a.

Definition 2.7 (**Signature**). *Given an assertion γ in ABox $\mathcal{A}$, the signature of γ, denoted $Sig(\gamma)$, is defined as a set of named individuals occurring in γ. This function is trivially extensible to $Sig(\mathcal{A})$ for the set of all individuals in ABox $\mathcal{A}$.*

Definition 2.8 (**Role path**). *We say there is a role path between individual a_1 and a_n, if for individuals $a_1, \ldots, a_n \in \boldsymbol{I}$ and $R_1, \ldots, R_{n-1} \in \boldsymbol{R}$, there exist either $R_i(a_i, a_{i+1})$ or $R_i^-(a_{i+1}, a_i)$ in $\mathcal{A}$ for all $i = 1, \ldots, n-1$.*

The role path from a_1 to a_n may involve inverse roles. For example, given $R_1(a_1, a_2)$, $R_2(a_3, a_2)$, and $R_3(a_3, a_4)$, the role path from a_1 to a_4 is $\{R_1, R_2^-, R_3\}$, while the opposite from a_4 to a_1 is $\{R_3^-, R_2, R_1^-\}$.

Definition 2.9 (**Simple-Form Concept**). *A concept is said in simple form, if the maximum level of nested quantifiers in the concept is less than 2.*

For example, given an atomic concept A, both A and $\exists R.A$ are simple-form concepts, while $\exists R_1.(A \sqcap \exists R_2.A)$ is not, since its maximum level of nested quantifiers is two. Notice however, an arbitrary concept can be *linearly* reduced to the simple form by assigning new concept names for fillers of the quantifiers, for example, $\exists R_1.\exists R_2.C$ can be converted to $\exists R_1.D$ by letting $D \equiv \exists R_2.C$ where D is a new class name.

3. Definition of an ABox Module

The notion of ABox module here is to be formalized w.r.t. ontology semantics and entailments that are only meaningful when the ontology is consistent. In this study, however, instead of restricting ontologies to be consistent, we aim to discuss the problem in a broader sense such that the theoretical conclusions hold regardless the consistency state of an ontology. For this purpose, we introduce the notion of *justifiable entailment* as defined below.

Definition 3.1 (**Justifiable Entailment**). *Let $\mathcal{K}$ be an ontology, α an axiom, and $\mathcal{K} \models \alpha$. α is called a justifiable entailment of $\mathcal{K}$, iff there exists a consistent fragment $\mathcal{K}' \subseteq K$ entailing α, i.e. $\mathcal{K}' \not\models \bot$ and $\mathcal{K}' \models \alpha$.*

It is not difficult to see that justifiable entailments of an ontology simply make up a subset of its logical entailments. More precisely, for a consistent ontology, the set of its justifiable entailments is exactly the set of its logical ones according to the definition above; while for an inconsistent ontology, justifiable entailments are those *sound* logical ones that have consistent bases (Huang et al., 2005). For example, given an inconsistent ontology $\mathcal{K}$=$\{C(a), \neg C(a)\}$, both $C(a)$ and $\neg C(a)$ are justifiable entailments of $\mathcal{K}$, but $R(a, b)$ is not.

Remark 3.1. *Unless otherwise stated, we take every entailment mentioned in this paper, denoted $\mathcal{K} \models \alpha$, to mean a justifiable entailment.*

Though complete reasoning on a large ABox may cause intractabilities, in realistic applications, a large ABox may consist of data with great diversity and isolation, and there are situations where a complete reasoning may not be necessary. For example, when performing instance checking of a given individual, say instance `CHRNA4` in the biomedical ontology example in Section 1, the ABox contains a great portion of other unrelated matters.

An ideal solution to this ABox reasoning problem is thus to extract a subject-related module and to have the reasoning applied to the module instead. Particularly, to fulfill soundness and completeness, this subject-related module should be a closure of entailments about the given entities, which in DL, should include class and property assertions. This leads to our formal definition of an ABox module as follows:

Definition 3.2 (ABox Module). *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be an ontology, and a set $\boldsymbol{S}$ of individuals be a signature. $\mathcal{M}_{\mathcal{S}}$ with $\mathcal{M}_{\mathcal{S}} \subseteq \mathcal{A}$ is called an ABox module for signature $\boldsymbol{S}$, iff for any assertion γ (either a class or a property assertion) with $Sig(\gamma) \cap \boldsymbol{S} \neq \emptyset$, $(\mathcal{T}, \mathcal{M}_{\mathcal{S}}) \models \gamma$ iff $\mathcal{K} \models \gamma$.*

This definition provides a formal criteria in terms of necessary and sufficient conditions for being an ABox module. It guarantees that sound and complete entailments (represented by γ) of individuals in signature $\mathbf{S}$ can be achieved by independent reasoning on the ABox module w.r.t $\mathcal{T}$. Class assertions preserved by an ABox module here are limited to atomic classes defined in $\mathcal{T}$.

Remark 3.2. *Limiting the preserved class assertions to only atomic classes simplifies the problems to deal with in this paper. While on the other hand, it should not be an obstacle in principle to allow this notion of ABox module to be applied for efficient instance retrieval or answering conjunctive queries with arbitrary classes. This is because we can always assign a new name A for an arbitrary query (possibly complex) class C by adding axiom $C \equiv A$ into $\mathcal{T}$.*

Definition 3.2 does not guarantee, however, uniqueness of an ABox module for signature $\mathbf{S}$, since any super set of $\mathcal{M}_{\mathcal{S}}$ is also a module for $\mathbf{S}$, due to the *monotonicity* of DLs (Baader et al., 2007). For example, given any signature $\mathbf{S} \subseteq \mathbf{I}$, the whole ABox $\mathcal{A}$ is always a module for $\mathbf{S}$.

Note however, the objective of this paper is to extract a precise ABox module and to select only *module-essential* assertions for a signature, so that the resulting module ensures completeness of entailments while keeping a relatively small size by excluding unrelated assertions. Intuitively, for assertions to be module-essential for a signature

S, they must be able to affect logical consequences of any individual in **S**, so that by having all these assertions included, the resulting ABox module can preserve all facts of the given entities. This criteria for being a module-essential assertion can be formalized based on the notion of *justification* (Kalyanpur et al., 2007) as given below.

Definition 3.3 (**Justification**). *Let $\mathcal{K}$ be an ontology, α an axiom, and $\mathcal{K} \models \alpha$. We say a fragment $\mathcal{K}' \subseteq K$ is a justification for axiom α, denoted $Just(\alpha, \mathcal{K})$, iff $\mathcal{K}' \models \alpha$ and $\mathcal{K}'' \not\models \alpha$ for any $\mathcal{K}'' \subset K'$.*

Definition 3.4 (**Module-essentiality**). *Let $\mathcal{K}$ be an ontology, a be an individual name, and γ an ABox assertion. γ is called module-essential for $\{a\}$, iff*

$$\gamma \in Just(\alpha, \mathcal{K}) \ \wedge \ \mathcal{K} \models \alpha,$$

for any assertion α of a (either $C(a)$ or $R(a, b)$, with $C \in \mathbf{C}$ and $R \in \mathbf{R}^*$) that can be derived from $\mathcal{K}$.

A justification $Just(\alpha, \mathcal{K})$ for axiom α is in fact a *minimum* fragment of the knowledge base that implies α, and every axiom (or assertion) in $Just(\alpha, \mathcal{K})$ is thus essential for this implication. Following from this point, an assertion γ is considered able to affect logical consequences of some individual in signature **S**, if and only if it appears in some justification for either (i) property assertion or (ii) class assertion of that individual. If so, γ is considered module-essential for **S** and should be included in $\mathcal{M}_{\mathcal{S}}$. With the notion of module-essentiality, we can now impose more restrictions on an ABox module, and the definition of an *Exact ABox Module* is derived as follows:

Definition 3.5 (**Exact ABox Module**). *Let **S** be a signature, $\mathcal{A}$ be an ABox, and $\mathcal{M}_{\mathcal{S}} \subseteq A$ be an ABox module for **S**. $\mathcal{M}_{\mathcal{S}}$ is called an Exact ABox Module for **S**, iff every assertion in $\mathcal{M}_{\mathcal{S}}$ is module-essential for **S** and every assertion in $\mathcal{A} \backslash M_S$ is not.*

In the following sections, a theoretical approach and an approximation are presented for the computation of an exact ABox module. Without loss of generality, we assume all ontology classes are in simple form as defined previously, and class terms in all class assertions are atomic.

We will show how to compute an ABox module in two steps: we begin by showing module extraction in an *equality-free* $\mathcal{SHIQ}$ ontology, and later we show how the basic technique can be extended to deal with equality.

4. ABox Modules in Equality-Free Ontologies

An ontology is called equality-free if it does not entail any equality (i.e. $\mathcal{K} \not\models a \approx b$, for any individual a, b in $\mathcal{K}$). In this section, we concentrate on a method that computes exact ABox modules in ontologies of this type. To further simplify the problem, we will consider module extraction for a single individual instead of an arbitrary signature **S**, since the *union* of modules of individuals in **S** yields a module for **S**, as indicated by the following proposition.

Proposition 4.1. *Let **S** be a signature, $\mathcal{M}_{\{i\}}$ be an ABox module for each individual $i \in$ **S**, and*

$$\mathcal{M}_{\mathcal{S}} = \bigcup_{i \in S} \mathcal{M}_{\{i\}}.$$

$\mathcal{M}_{\mathcal{S}}$ is an ABox module for $\mathbf{S}$.

Proof. Assume $\mathcal{M}_{\mathcal{S}}$ is not a module for $\mathbf{S}$, then there exists an assertion γ, with $Sig(\gamma) \cap \mathbf{S} \neq \emptyset$, and either (i) $(\mathcal{T}, \mathcal{M}_{\mathcal{S}}) \models \gamma$ and $\mathcal{K} \not\models \gamma$ or, (ii) $\mathcal{K} \models \gamma$ and $(\mathcal{T}, \mathcal{M}_{\mathcal{S}}) \not\models \gamma$.

(i) clearly contradicts DL monotonicity. For (ii), let individual $a \in Sig(\gamma) \cap \mathbf{S}$, $\mathcal{M}_{\{a\}}$ with $\mathcal{M}_{\{a\}} \subseteq \mathcal{M}_{\mathcal{S}}$ be the module for a. Then, by the definition of a module, we have $(\mathcal{T}, \mathcal{M}_{\{a\}}) \models \gamma$, which again conflicts with the monotonicity of DLs, since $\mathcal{M}_{\{a\}}$ is subsumed by $\mathcal{M}_{\mathcal{S}}$. Hence, the proposition holds. ■

4.1. Strategy

To compute an exact ABox module for a given individual a, basically we have to test every assertion in $\mathcal{A}$ to see if it is module-essential for a, that is, for every assertion we have to test whether it contributes in deducing any property assertion or class assertion of individual a.

As a fact of $\mathcal{SHIQ}$, deducing class assertions (classification) of an individual usually depends on both of its class and property assertions. While conversely, the tableau-expansion procedure for $\mathcal{SHIQ}$ indicates that deduction of a property assertion between different named individuals in $\mathcal{A}$ should not be affected by their class assertions, except via *individual equality* (Horrocks et al., 2000) as discussed in Section 5. This is consistent with the tree-model property of the description logic that, if nominals are not involved in the knowledge base, no tableau rules can derive connection (i.e. property assertion) from individual a to an arbitrary named individual, except to itself in the presence of (local) reflexivity (Motik et al., 2009).

Therefore, given an equality-free ontology, the above observation allows us to deduce property assertions from the ABox w.r.t. only role hierarchies[1] defined in $\mathcal{T}$ (Horrocks et al., 2000), and a strategy for extracting an ABox module can then be devised.

Proposition 4.2. *Given an equality-free $\mathcal{SHIQ}$ ontology, computation of an ABox module for individual a can be divided into two steps:*

1. *compute a set of assertions (denoted $\mathcal{M}^{\mathcal{P}}_{\{a\}}$) that preserves any property assertion $R(a, b)$ of a, with $a \not\approx b$ and $R \in \mathbf{R}^*$,*
2. *compute a set of assertions (denoted $\mathcal{M}^{\mathcal{C}}_{\{a\}}$) that preserves any class assertion $C(a)$ of a, with $C \in \mathbf{C}$.*

For simplicity, we call $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ a *property-preserved module* and $\mathcal{M}^{\mathcal{C}}_{\{a\}}$ a *classification-preserved module* for $\{a\}$.

4.2. Property-Preserved Module

A property-preserved module of individual a is essentially a set of assertions in ontology $\mathcal{K}$, which affect the deduction of a's property assertions. This set is denoted $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ such that

$$\mathcal{M}^{\mathcal{P}}_{\{a\}} = \{\gamma \mid \gamma \in Just(R(a, b), \mathcal{K}) \ \wedge \ \mathcal{K} \models R(a, b)\},$$

[1] Note that transitive roles can be expressed as $R \circ R \sqsubseteq R$.

where γ is an ABox assertion, and $Just(R(a,b),\mathcal{K})$ is any justification for $R(a,b)$, with $a \not\approx b$.

In an equality-free $\mathcal{SHIQ}$ ontology, since property assertions between different individuals can be deduced from the ABox $\mathcal{A}$ w.r.t. role hierarchies, the computation for $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ is then straightforward based on the following fact: for any $R \in \mathbf{R}^*$, if $\mathcal{K} \models R(a,b)$ with $a \not\approx b$, there are two possibilities on an equality-free $\mathcal{SHIQ}$ ontology, according to (Horrocks et al., 2000):

1. assertion $R_0(a,b)$ or $Inv(R_0)(b,a) \in \mathcal{A}$ with $R_0 \sqsubseteq R$,
2. assertions involved in a role path from a to b, with all roles having a common transitive parent R_0 and $R_0 \sqsubseteq R$. e.g. $R_1(a,a_1)$, $R_2(a_2,a_1)$, $R_3(a_2,b) \in \mathcal{A}$, with $R_1, R_2^-, R_3 \sqsubseteq R_0$ and R_0 is transitive.

Abstracting from a particular R_0, these two possibilities can be generalized into a formal criteria to select assertions to include in $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ for individual a:

$C1$. property assertions in $\mathcal{A}$ that have a as either subject or object, or

$C2$. property assertions in $\mathcal{A}$ that are involved in a role path from a to some b, with all roles in the path having a common transitive parent.

Proposition 4.3. *On an equality-free $\mathcal{SHIQ}$ ontology, the set of property assertions satisfying criteria $C1$ or $C2$ forms a property-preserved module $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ for individual a.*

Proof. Correctness of this proposition can be verified by observation of the tableau-constructing procedure for $\mathcal{SHIQ}$ presented in (Horrocks et al., 2000). Let a tableau $T = (\Delta, \mathcal{L}, \mathcal{E}, .^{\mathcal{I}})$ be an interpretation for $\mathcal{K}$ as defined in (Horrocks et al., 2000), where Δ is a non-empty set, $\mathcal{L}$ maps each element in Δ to a set of concepts, $\mathcal{E}$ maps each role to a set of pairs of elements in Δ, and $.^{\mathcal{I}}$ maps individuals in $\mathcal{A}$ to elements in Δ.

They have proven that, for tableau T to be a model for $\mathcal{K}$, if $\mathcal{K} \models R(a,b)$, there must be either $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in \mathcal{E}(R)$ or a path $(a^{\mathcal{I}}, s_1), (s_1, s_2), \ldots, (s_n, b^{\mathcal{I}}) \in \mathcal{E}(R_0)$ with $R_0 \sqsubseteq R$ and R_0 being transitive. The second scenario is consistent with criteria $C2$; while for the first one, i.e. $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in \mathcal{E}(R)$, there are only two possibilities according to the tableau-constructing procedure: (i) $R_0(a,b)$ or $R_0^-(b,a) \in \mathcal{A}$ with $R_0 \sqsubseteq R$ that triggers initialization of $\mathcal{E}(R_0)$; (ii) $R_0(a,b)$ or $R_0^-(b,a)$ is obtained through the $\leq_r$-rule for identical named individuals (Horrocks et al., 2000) with $R_0 \sqsubseteq R$. (i) reflects exactly the criteria $C1$, while (ii) does not apply here for equality-free ontologies. Therefore, the proposition holds. ■

4.3. Exact ABox Module

To compute an exact ABox module $\mathcal{M}_{\{a\}}$, we need to further decide a set of assertions that affect classifications of the individual, and this set is denoted $\mathcal{M}^{\mathcal{C}}_{\{a\}}$ such that

$$\mathcal{M}^{\mathcal{C}}_{\{a\}} = \{\gamma \mid \gamma \in Just(A(a),\mathcal{K}) \ \wedge \ \mathcal{K} \models A(a)\},$$

where γ is an ABox assertion, A is an atomic class, and $Just(A(a),\mathcal{K})$ is any justification for $A(a)$.

As previously stated, in $\mathcal{SHIQ}$ an individual is usually classified based on both its class and property assertions in $\mathcal{A}$. It is obvious that explicit class assertions of a form

an indispensable part of $\mathcal{M}^{\mathcal{C}}_{\{a\}}$. Then, to decide any property assertion of a that affects its classification, we examine each assertion captured in $\mathcal{M}^{\mathcal{P}}_{\{a\}}$.

The decision procedure here is based on the idea that instance checking is reducible to concept subsumption (Donini and Era, 1992; Donini et al., 1994; Nebel, 1990), i.e.

> *Given an ontology $\mathcal{K} = (T, A)$, an individual a and a $\mathcal{SHIQ}$-class C, a can be classified into C, if the class term behind a's assertions in the ABox is subsumed by C w.r.t. $\mathcal{T}$.*

This idea automatically lends itself as a methodology, such that to determine any assertion of an individual that contributes to its classification, we have to decide if the class term behind this assertion is subsumed by some class w.r.t. $\mathcal{T}$.

Consider the following example: Let an ontology $\mathcal{K} = (T, A)$ be

$$(\{\exists R_0.B \sqsubseteq A\},\ \{R_0(a, b), B(b)\}),$$

and let us ask whether $R_0(a, b)$ is essential for individual a's classification or not. To answer that, we need to decide if the class term behind this property assertion is subsumed by some class w.r.t. $\mathcal{T}$, i.e. to test if

$$\mathcal{K} \models \exists R.C_1 \sqsubseteq C_2 \tag{1}$$

for some named class C_2, with $R_0 \sqsubseteq R$ and $b \in C_1$. It is easy to see (1) is satisfied in this example by substituting B for C_1 and A for C_2. We can thus determine $R_0(a, b)$ is one of the causes for the entailment $C_2(a)$ (i.e. it is in some justification for $C_2(a)$), and should be an element of $\mathcal{M}^{\mathcal{C}}_{\{a\}}$. Moreover, assertions in $Just(C_1(b), \mathcal{K})$ should also be elements of $\mathcal{M}^{\mathcal{C}}_{\{a\}}$, since $C_1(b)$ is another important factor to the classification $C_2(a)$.

The above example illustrates a simple case of a single property assertion affecting classification of an individual. Classification of an individual can also be caused by multiple assertions. For example, let $\mathcal{K}$ be

$$\begin{array}{c}(\{\exists R_0.B \sqcap \exists R_1.C \sqsubseteq A\},\\ \{R_0(a, b), R_1(a, c), B(b), C(c))\}).\end{array}$$

Here, $R_0(a, b)$ is still essential for the deduction $A(a)$. But when testing the subsumption in (1) for $R_0(a, b)$, it will be found unsatisfied.

Thus, in order to comprehensively and completely include other information about the individual, condition (1) should be generalized into:

$$\mathcal{K} \models \exists R.C_1 \sqcap C_3 \sqsubseteq C_2 \tag{2}$$

where all other information of individual a is summarized and incorporated into a class C_3 with $C_3 \not\sqsubseteq C_2$.

Moreover, taking the number restrictions in $\mathcal{SHIQ}$ into consideration, condition (2) can be further generalized as:

$$\mathcal{K} \models \geq nR.C_1 \sqcap C_3 \sqsubseteq C_2 \ \wedge\ |R^{\mathcal{K}}(a, C_1)| \geq n. \tag{3}$$

Note that, $\exists R.C_1$ is only a special case of $\geq nR.C_1$, and $R^{\mathcal{K}}(a, C_1) = \{b_i \in \mathbf{I} \mid \mathcal{K} \models R(a, b_i) \wedge C_1(b_i)\}$ denotes the set of decidable and distinct R-neighbors of individual a in C_1.

1. Compute a property-preserved module $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ for the given individual a, by following the criteria $C1$ and $C2$ given in Section 4.2.
2. Add all explicit class assertions of a into $\mathcal{M}^{\mathcal{C}}_{\{a\}}$.
3. For every $R_0(a, b)$ ($R_0 \in \mathbf{R}^*$) captured in $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ and any R with $R_0 \sqsubseteq R$, test if the corresponding condition (3) is satisfied. If it is yes, add $R_0(a, b)$, assertions in $Just(C_1(b), \mathcal{K})$s, and any inequality assertions between individuals in $R^{\mathcal{K}}(a, C_1)$ into $\mathcal{M}^{\mathcal{C}}_{\{a\}}$.
4. Unite the sets, $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ and $\mathcal{M}^{\mathcal{C}}_{\{a\}}$, to form the exact ABox module for a.

Figure 1: Steps for computation of an exact ABox module for individual a.

In a general way, condition (3) indicates that in an equality-free $\mathcal{SHIQ}$-ontology, for any property assertion $R(a, b)$ to affect classification of individual a, the corresponding class term must be subsumed by some named class, and for any (qualified) number restrictions, the number of distinct R-neighbors of a should be no less than the cardinality required. With this condition derived, we are now in a position to present a procedure for computation of $\mathcal{M}^{\mathcal{C}}_{\{a\}}$ and also an exact ABox module for individual a, which is summarized in Figure 1.

4.4. Approximation of an Exact ABox Module

Computation of $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ depends on the complete role hierarchy, which should be computable using a DL-reasoner, since roles in $\mathcal{SHIQ}$ are atomic and, most importantly, the size of $\mathcal{T}$ should be much smaller than $\mathcal{A}$ in realistic applications (Motik and Sattler, 2006). On the other hand, it is difficult to compute $\mathcal{M}^{\mathcal{C}}_{\{a\}}$, since it demands computation of both concept subsumption (i.e. condition (3)) and justifications for class assertions (i.e. $Just(C_1(b), \mathcal{K})$). Simple approximations for both are given in this section as follows.

Definition 4.1 (Approximation of (3)). *A syntactic approximation for condition (3) for $R_0(a, b)$ is that: to test if $\mathcal{K}$ contains any formula in the form as listed below:*

$$\begin{array}{l} \exists R.C_1 \bowtie C_3 \sqsubseteq C_2 \\ \mid C_1 \sqsubseteq \forall R^-.C_2 \bowtie C_3 \\ \mid \ \geq nR.C_1 \bowtie C_3 \sqsubseteq C_2 \ \wedge \ |\{b_i \mid R(a, b_i) \in \mathcal{A}\}| \geq n \end{array} \tag{4}$$

where $R_0 \sqsubseteq R$, $C_i \in \mathbf{C}$, $\bowtie$ is a place holder for $\sqcup$ and $\sqcap$, and concepts are checked in negation normal form (NNF) (Baader et al., 2007). Also note the following equivalences:

$$\begin{array}{rcl} \exists R.C \sqsubseteq D & \equiv & \neg D \sqsubseteq \forall R.\neg C \\ \geq nR.C \sqsubseteq D & \equiv & \neg D \sqsubseteq \leq (n-1)R.C. \end{array}$$

For assertion $R_0(a, b)$, the approximation for condition (3) is to check if any formula in $\mathcal{K}$ is in the form of any listed axioms in (4). If it is yes, $R_0(a, b)$ may *potentially* affect some logical entailment of individual a, and related assertions will be added into a's ABox module to ensure preservation of this potential entailment. Validity of this approximation is shown by the following proposition.

Proposition 4.4. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a $\mathcal{SHIQ}$ ontology with simple-form concepts only, $\geq nR_0.B$ and C be $\mathcal{SHIQ}$ concepts, and D a named concept. If*

$$\mathcal{K} \models \geq nR_0.B \sqcap C \sqsubseteq D \tag{5}$$

with $C \not\sqsubseteq D$, there must exist some formula in $\mathcal{T}$ in the form as listed in (4) for some R with $R_0 \sqsubseteq R$.

Proof. Since $\exists R.C_1$ is a special case of $\geq nR.C_1$, $\exists R.C_1 \sqsubseteq C_2$ is equivalent with $C_1 \sqsubseteq \forall R^-.C_2$, and together with those equivalences mentioned above, every role restriction in $\mathcal{T}$ can be converted to the form of $\geq nR.C_1$ by axiom manipulation. Then, the task here is reducible to proving that if (5) is satisfied, there must be some formula in $\mathcal{K}$ in the form of $\geq nR.C_1 \bowtie C_2 \sqsubseteq C_3$ for some R with $R_0 \sqsubseteq R$.

It is straightforward that, if such no R with $R_0 \sqsubseteq R$ is used in concept definition, $\geq nR.B$ is not comparable (w.r.t. subsumption) with any atomic class (except $\top$ and its equivalents). On the other hand, if $\geq nR.C_1$ is used in concept definition but occurs only in the right-hand side (r.h.s.) of GCIs, it is unable to indicate any atomic class as its subsumer, which can be confirmed by observation of a tableau-constructing procedure.

Let P, Q be two atomic classes, $Q \neq \top$, P and $\neg P$ not fillers in any restrictions, and all concepts of $\mathcal{T}$ in NNF. Assume $(*)$ P occurs only in r.h.s. of GCIs (or $\neg P$ in l.h.s.), and there is a consistent fragment $\mathcal{T}' \subseteq T$ that $\mathcal{T}' \models P \sqsubseteq Q$. It follows that:

(E1) $\mathcal{T}' \cup \{a\} \models \neg P \sqcup Q(a)$ for any individual a, since $P \sqsubseteq Q$ implies $\top \sqsubseteq \neg P \sqcup Q$.

(E2) $\mathcal{T}' \cup \{\neg Q(a)\} \models \neg P(a)$, because of (E1).

(E3) $\mathcal{T}' \cup \{P \sqcap \neg Q(a)\} \models \bot$, the so-called refutation-style proof for $P \sqsubseteq Q$.

(E1) implies that, in any tableau that is a model of $\mathcal{T}' \cup \{a\}$, there must be either $\neg P$ or Q in $\mathcal{L}(a^{\mathcal{I}})$ (i.e. the class set of $a^{\mathcal{I}}$ in the tableau), which can be shown by contradiction: suppose $\mathcal{I}_1$ is a model for $\mathcal{T}' \cup \{a\}$, where neither $\neg P$ nor Q is in $\mathcal{L}(a^{\mathcal{I}_1})$. Let $\mathcal{I}_2$ be another tableau such that $\mathcal{I}_2$ coincides with $\mathcal{I}_1$ except $\mathcal{L}(a^{\mathcal{I}_2})$ is extended with $\{P, \neg Q\}$, and $\mathcal{I}_2$ should be clash-free since both P and Q are atomic and no tableau rules can be applied. Thus, $\mathcal{I}_2$ turns out to be a model for $\mathcal{T}' \cup \{P \sqcap \neg Q(a)\}$ that violates (E3).

Analogously for (E2), there must be $\neg P$ in $\mathcal{L}(a^{\mathcal{I}})$ for any model of $\mathcal{T}' \cup \{\neg Q(a)\}$. Nevertheless, if P occurs only in r.h.s. of GCIs in $\mathcal{T}$, $\neg P$ can never exist after the NNF transformation of axioms in $\mathcal{T}$, and since P and $\neg P$ are not fillers in any restrictions, $\mathcal{L}(a^{\mathcal{I}})$ can never comprise $\neg P$ according to the tableau rules (Horrocks et al., 2000). Hence, the original assumption $(*)$ does not hold.

The above case essentially indicates that, if an atomic concept occurs only in the r.h.s. of GCIs in $\mathcal{T}$, its subsumer is undecidable. And the same general principle applies, if we consider all $\geq nR.C_1$ for any R with $R_0 \sqsubseteq R$ as a single unit. Thus, there must be some $\geq nR.C_1$ with $R_0 \sqsubseteq R$ occurring in l.h.s. of GCIs in $\mathcal{T}$, if (5) is true. ■

Proposition 4.4 shows the completeness of the approximation (4). We can thus conclude that an ABox module resulting from this approximation is still able to capture complete classifications (w.r.t. $\mathcal{T}$) of the given individual, which are derivable from its property assertions. The following statement is an immediate consequence of this conclusion: if $C_1 \neq \top$, an approximation for the union of assertions in $Just(C_1(b), \mathcal{K})$s is

1. Compute a property-preserved module $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ for the given individual a, by following the criteria $C1$ and $C2$ given in Section 4.2.
2. Add all explicit class assertions of a into $\mathcal{M}^{\mathcal{C}}_{\{a\}}$.
3. For every $R_0(a, b)$ captured in $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ and any R with $R_0 \sqsubseteq R$, test if $\mathcal{K}$ contains any formula in the form as listed in (4). If it is yes, add $R_0(a, b)$, all assertions in $\mathcal{M}_{\{b\}}$, and any inequality assertions between a's R-neighbors into $\mathcal{M}^{\mathcal{C}}_{\{a\}}$.
4. Unite the sets, $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ and $\mathcal{M}^{\mathcal{C}}_{\{a\}}$, to form an approximation for the exact ABox module for a.

Figure 2: Steps for approximation of an exact ABox module for individual a.

$\mathcal{M}_{\{b\}}$, the exact ABox module for b, which is approximated using the same strategies described here.

A procedure for approximating an exact ABox module is then summarized in Figure 2.

5. Module Extraction with Equality

In this section, we show how the outcome from the previous section can be utilized to tackle module extraction with individual equality.

In $\mathcal{SHIQ}$, individual equality is recognized through the $\leq_r$-rule defined in (Horrocks et al., 2000), which is briefly summarized as follows:

$\leq_r$-rule: If $\leq n.R.C \in \mathcal{L}(x^{\mathcal{I}})$, and $x^{\mathcal{I}}$ has more than n R-neighbors in C, then for two of these R-neighbors $y^{\mathcal{I}}$ and $z^{\mathcal{I}}$, where y, z are named individuals, if $y^{\mathcal{I}} \not\approx z^{\mathcal{I}}$ does not hold, then $y^{\mathcal{I}} \approx z^{\mathcal{I}}$ (thus $y \approx z$).

This rule indicates that determination of individual equality requires computation of both property and class assertions of related individuals, i.e. in the $\leq_r$-rule, in order to derive $y \approx z$ we need to know at least

$$\mathcal{K} \models \leq n.R.C(x) \wedge R(x, y) \wedge R(x, z) \wedge C(y) \wedge C(z).$$

Hence, with arbitrary equality, it is difficult to extract an ABox module, since it requires an expensive ontology reasoning process to detect equality. In addition, the strategy proposed in Proposition 4.2 becomes infeasible, since property assertions can be derived from equalities (e.g. given $y \approx z$, $R(y, w)$ simply implies $R(z, w)$). In other words, with equality, the computation of $\mathcal{M}^{\mathcal{P}}_{\{a\}}$ may be dependent on that of $\mathcal{M}^{\mathcal{C}}_{\{a\}}$.

To address this, we present a procedure for extraction of ABox modules, which first modularizes the ABox as if it were equality-free, and then resolves equality through post-processing.

Proposition 5.1. *Let individual $x \in \leq nR.C$ have more than n R-neighbors in C, of which two, y and z, can be determined to be equal (i.e. $y \approx z$). Let signature $\mathbf{S}$ consist of x and all its R-neighbors in C, and*

$$\mathcal{M}_{\mathcal{S}} = \bigcup_{i \in S} \mathcal{M}_{\{i\}}$$

where $\mathcal{M}_{\{i\}}$ is an ABox module for each individual $i \in \mathbf{S}$ in the "equality-free" ABox. $\mathcal{M}_{\mathcal{S}}$ preserves the equality $y \approx z$.

Proposition 5.1 suggests a strategy to retain equality between y and z, by combining "modules" of related individuals. With $y \approx z$ preserved, $\mathcal{M}_{\mathcal{S}}$ automatically preserves all facts of y and z that are derived from the equality. Subsequently, for neighbors of y and z (i.e. individuals in $Sig(\mathcal{M}^{\mathcal{P}}_{\{y\}})$ and $Sig(\mathcal{M}^{\mathcal{P}}_{\{z\}})$), modules of these entities should be combined with the $\mathcal{M}_{\mathcal{S}}$ obtained above, so that their facts derivable from $y \approx z$ can also be captured. This strategy to retain equality in ABox modules should be applied *recursively* for all identities.

Notice, however, that the strategy in Proposition 5.1 is based on the condition that individuals y and z be known to be equal in the first place, which cannot be assured without ABox reasoning. Nevertheless, conceiving that equality in $\mathcal{SHIQ}$ stems from number restrictions (Horrocks et al., 2000), a simple approximation for it is given in the definition below.

Definition 5.1. *Let x,y and z be named individuals, y, z be R-neighbors of x, and $y \not\approx z$ not hold explicitly*[2]*, y and z are considered **potential equivalents**, if their R-predecessor x:*

1. *has no potential equivalents, and has m R-neighbors with $m \gneq n$, or*
2. *has a set of potential equivalents, denoted X (which includes x), and there exists a set $S \in \binom{X}{m'-n'+1}$, such that*

$$\max_{S} |\{y_i \mid R(x_i, y_i) \in \mathcal{A} \ \wedge \ x_i \in S\}| = m \gneq n. \tag{6}$$

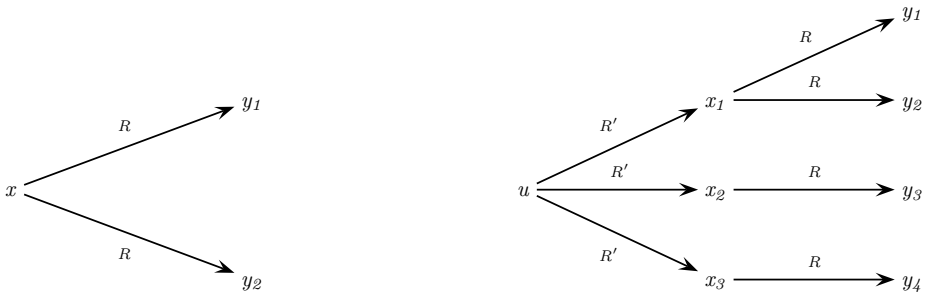
where R is used in number restrictions as in the axiom listed in (4), and n is the minimum of the set $\{k \mid \geq (k+1)R.C$ in l.h.s. of GCIs$\}$. $\binom{X}{m'-n'+1}$[3] denotes the set of all $(m'-n'+1)$-combinations of set X, and variables m' and n' are for identification of x that correspond to m and n above, respectively.

In this definition, for $\mathcal{K} \models \leq nR.C(x)$ to be possible it is required that $\leq nR.C$ or its isoform occur in the r.h.s. of GCIs in $\mathcal{T}$ (or $\geq (n+1)R.C$ in l.h.s. as in (4)), proof for which is similar to the one given for Proposition 4.4.

Moreover, if x itself has potential equivalents, individuals y and z should be elements of $\{y_i \mid R(x_i, y_i) \in \mathcal{A} \wedge x_i \in X\}$. For the counting of potential R-neighbors of x, instead of taking the entire set X, only the $(m'-n'+1)$-combination of X that maximizes the counting is considered (see (6)), since given m' R'-neighbors, the maximum possible number of decidable identical entities is $(m'-n'+1)$ out of m' according to (Horrocks et al., 2000). Examples for two cases of individual x in Definition 5.1 are illustrated in Figure 3.

Proposition 5.2. *Applying the strategy in Proposition 5.1 to potential equivalents generates modules that preserve individual equality.*

[2]Either $y \not\approx z$ is not explicitly given, or assertions $C(y)$ and $\neg C(z)$ do not occur simultaneously in $\mathcal{A}$.
[3]If $(m'-n'+1) \geq |X|$, it denotes $\{X\}$.



(a) $x \in \leq 1R.\top$ (b) $u \in \leq 2R'.\top,\ x_1 \in \leq 2R.\top$

Figure 3: Examples for two cases for individual x in Definition 5.1. In figure (a), y_1 and y_2 are inferred to be equal, if $y_1 \not\approx y_2$ doesn't hold. In (b), all x_i(s) are considered potential equivalents if none of them are explicitly asserted different from each other; on the other hand, only two of them can be inferred to be equal, when sufficient information is provided. If x_1 is equated with any one of the others, its R-neighbors (i.e. y_i(s)) may be further equated.

Proof. (Sketch) We prove by induction.

Basis: Assume initially, $y \approx z$ is the only equality in ABox that is entailed from the $\leq_r$-rule, then, there must be $R_1(x, y)$ and $R_2(x, z)$ in $\mathcal{M}^P_{\{x\}}$, for some $x \in \leq nR.C$, x having more than n R-neighbors, and $R_1, R_2 \sqsubseteq R$. According to the strategy in Proposition 5.1, modules in "equality-free" ABox of x and its R-neighbors are merged, and resulted $\mathcal{M}_{\mathcal{S}}$ thereby preserves all facts (except those derived from $y \approx z$) of these individuals, including assertion $R(x, i)$ and classification for each x's R-neighbor i, which are sufficient to entail $y \approx z$, according to the $\leq_r$-rule (Horrocks et al., 2000).

Inductive step: Assume ABox $\mathcal{A}$ with arbitrary individual equality is modularized, and all extracted modules are in compliance with Definition 3.2. $\mathcal{A}$ is then further extended by adding a new assertion $R(x', y')$ that causes fresh equality between individual y' and z'. If x' has no equivalent, this simply follows what we have discussed above. Otherwise, we take into account all x''s potential equivalents in set X' for its R-neighbors that comprise both y' and z'. There exists a set $S \subseteq X'$ of true positives such that $|\{y'_i | R(x'_i, y'_i), x'_i \in S\}|$ is greater than the cardinality restriction as required for individual identity, and S is at most a $(m' - n' + 1)$-combination of X'. Modules of x' and all its R-neighbors are then merged, and as discussed above equality between y' and z' is preserved.

■

6. Related Work

Modularization of the terminological part of an ontology has already been addressed by (Cuenca Grau et al., 2007a,b), where the author provided a well-defined idea of

ontology modularity and developed a logic framework for extracting modules for terminological concepts. On the other hand, for ABox reduction or partition, the problem has been addressed mainly by (Fokoue et al., 2006), (Guo and Heflin, 2006), (Du and Shen, 2007) and (Wandelt and Möller, 2012).

The idea of (Fokoue et al., 2006) is to reduce large ABoxes by combining similar individuals (w.r.t. individual description) to develop a summary ABox as the proxy, which is small enough for scalable reasoning. This ABox summary can be useful in some scenarios that involve general testing of consistency of an ontology, but it has a limited capability to support ontology queries such as instance retrieval.

In (Guo and Heflin, 2006), the authors proposed a method for computing independent ABox partitions for $\mathcal{SHIF}$ ontologies, such that complete reasoning can be achieved if combining results of independent reasoning on each partition. This method for ABox partitioning is based on a set of inference rules in (Royer and Quantz, 1993), and it encapsulates assertions that are antecedents of a inference rule as an ABox partition. This technique is also used by (Williams et al., 2010) to handle reasoning on large data sets.

ABox partition is also used in (Du and Shen, 2007), where the authors proposed an algorithm for $\mathcal{SHIQ}$ ontologies. Based on the technique in (Hustadt et al., 2004) that converts DL ontologies to disjunctive datalog programs, their algorithm further converts the disjunctive datalog program into a plain datalog program, to generate rules that can be employed as guidelines for ABox partition.

Both approaches above for ABox partition, however, failed to impose any logical restrictions on a single partition. An immediate consequence is that every single assertion can turn out to be a partition if the ontology is too simple, i.e. has no transitive roles nor concepts defined upon restrictions. Besides, to get complete entailments of an individual, one still has to reason over all partitions of the ontology.

The work most related to our proposed techniques is presented in (Wandelt and Möller, 2012), which focuses on $\mathcal{SHI}$ ontologies. In their paper, instead of imposing any specification on a single ABox module, the authors provided a formal definition directly for ABox modularization that extends the notion of ABox partition from (Guo and Heflin, 2006). A briefly summarized version of their definition is given below.

Definition 6.1 (**ABox Modularization** (Wandelt and Möller, 2012))**.** *Given ontology $\mathcal{K} = (T, A)$, an ABox modularization M is defined as a set of ABoxes $\{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$, where each $\mathcal{A}_i \subseteq \mathcal{A}$ is an ABox module. M is said sound and complete for instance retrieval if, given any class assertion $C(a)$ (C is atomic), $\exists \mathcal{A}_i \in M$ that $\mathcal{T} \cup \mathcal{A}_i \models C(a)$ iff $\mathcal{K} \models C(a)$.*

In (Wandelt and Möller, 2012), the general idea for ABox modularization is based on connected components in an ABox graph. The authors presented an *ABox-Split* based approximation, such that after applying ABox-split on each $R(a, b) \in \mathcal{A}$, every connected component in the resulted ABox graph forms an ABox module, and the set of all connected components forms the modularization defined above.

More precisely, this ABox-split technique checks every property assertion $R(a, b)$ in an ABox and will replace it with two generated ones, $R(a', b)$ and $R(a, b')$, if all the conditions below are satisfied for $R(a, b)$ (Wandelt and Möller, 2012):

1. R is neither transitive nor has any transitive parent.

Table 1: ABox modules in different ontologies

Ontology	Exp.	#Ast.	#Ind.	Max. #Ast./#Ind.	Avg. #Ast./#Ind.	Avg. Extraction Time
LUBM-1	$\mathcal{SHI}$	67,465	17,175	2,921/593	13.1/2.4	1.12 ms
LUBM-2	$\mathcal{SHI}$	319,714	78,579	2,921/593	14.4/2.5	1.22 ms
VICODI	$\mathcal{ALH}$	53,653	16,942	8,591/1	5.3/1	0.28 ms
AT	$\mathcal{SHIN}$	117,405	42,695	54,561/10,870	6.9/1.7	1.06 ms
CE	$\mathcal{SHIN}$	105,238	37,162	49,914/9,315	7.1/1.7	0.60 ms
DBpedia*_1	$\mathcal{SHIQ}$	402,062	273,663	94,862/18,671	2.9/1.1	0.62 ms
DBpedia*_2	$\mathcal{SHIQ}$	419,505	298,103	160,436/17,949	2.8/1.1	0.59 ms
DBpedia*_3	$\mathcal{SHIQ}$	388,324	255,909	140,050/35,720	3.1/1.2	1.80 ms
DBpedia*_4	$\mathcal{SHIQ}$	398,579	273,917	139,422/18,208	2.9/1.1	0.51 ms

2. For every $C \in \mathit{extinfo}^{\forall}_{\mathcal{T}}(R)$, it satisfies that: (i) $C \equiv \bot$, or (ii) there exists $D(b) \in \mathcal{A}$ such that $\mathcal{K} \models D \sqsubseteq C$, or (iii) there exists $D(b) \in \mathcal{A}$ such that $\mathcal{K} \models D \sqcap C \sqsubseteq \bot$.
3. For every $C \in \mathit{extinfo}^{\forall}_{\mathcal{T}}(R^-)$, it satisfies that: (i) $C \equiv \bot$, or (ii) there exists $D(a) \in \mathcal{A}$ such that $\mathcal{K} \models D \sqsubseteq C$, or (iii) there exists $D(a) \in \mathcal{A}$ such that $\mathcal{K} \models D \sqcap C \sqsubseteq \bot$.

where $\mathit{extinfo}^{\forall}_{\mathcal{T}}(R)$ is the set of classes in $\mathcal{T}$ that are used as fillers of R and are able to propagate through the $\forall$-rule (Horrocks et al., 2000) in the tableau algorithm.

In general, this ABox-split based approximation will replace an original property assertion $R(a, b)$ with two generated ones, if it can be determined that, either (i) $R(a, b)$ has no influence on hidden implications of a nor b, or (ii) its influence relies on individual's classification that has already been explicitly given. This approximation hence "splits" assertion $R(a, b)$ and separates a and b into different connected components (ABox modules) in ABox graph. For example, given ontology $\mathcal{K} = \{R(a, b), \forall R.B(a), B(b)\}$, $B(b)$ is entailed by $\mathcal{K}$ by either the explicitly given assertion or $\{R(a, b), \forall R.B(a)\}$. Nevertheless, since $B(b)$ is explicitly given, this approximation can separate individual a and b into two ABox modules.

To sum up, the method by (Wandelt and Möller, 2012) aims at computing ABox modules as small as possible (different from the defined Exact ABox Modules in this paper) for $\mathcal{SHI}$-ontologies; and it makes use of information from the class hierarchy (including concept subsumption and concept disjointness, where complex classes may be involved) to rule out assertions for duplicate or impossible implications, which thus requires invocation of a DL reasoner. Moreover, this approach requires a consistent ontology as the prerequisite.

Table 2: Small and simple ABox modules in ontologies

Ontology	Total #Modules	#Module with #Ast $\leq$ 10 (%)	#Module with Signature Size = 1 (%)
LUBM-1	7,264	7,210 (99.3)	7,209 (99.2)
LUBM-2	31,361	30,148 (96.1)	31,103 (99.2)
VICODI	16,942	16,801 (99.2)	16,942 (100)
AT	24,606	23,464 (95.4)	23,114 (93.9)
CE	21,305	20,010 (93.9)	19,455 (91.3)
DBpedia*_1	239,758	233,231 (97.3)	237,676 (99.1)
DBpedia*_2	264,079	257,555 (97.5)	261,973 (99.2)
DBpedia*_3	208,401	201,905 (96.9)	206,377 (99.0)
DBpedia*_4	241,451	234,903 (97.3)	239,346 (99.1)

7. Empirical Evaluation

We implemented our approximation using Manchester's OWL API[4], and evaluate it on a lab PC with Intel(R) Xeon(R) 3.07 GHz CPU, Windows 7, and 1.5 GB Java Heap. For test data, we collected a set of ontologies with large ABoxes:

1. VICODI[5] is an ontology that models European history;
2. LUBM-1 and LUBM-2 are benchmark ontologies generated using tools provided by (Guo et al., 2005);
3. Arabidopsis thaliana (AT) and Caenorhabditis elegans (CE) are two biomedical ontologies[6] based on a common TBox[7] that models biological pathways; and
4. DBpedia* ontologies are acquired from the original DBpedia ontology [8] (Auer et al., 2007). They have a common terminological part $\mathcal{T}$, DL expressivity of which is extended from $\mathcal{ALF}$ to $\mathcal{SHIQ}$ by adding complex roles and classes defined on role restrictions, and their ABoxes are generated by randomly sampling the original ABox.

Details of these ontologies are summarized in Table 1, in terms of expressiveness (Exp.), number of assertions (#Ast.), and number of individuals (#Ind.).

7.1. Evaluation of Extracted Modules

These collected ontologies were modularized using the approximation for module extraction for every named individual. As discussed previously, extracting ABox module for a single individual results in a module whose signature is constituted by a set of entities because of module combination.

[4] http://sourceforge.net/projects/owlapi
[5] http://www.vicodi.org
[6] http://www.reactome.org/download
[7] http://www.biopax.org
[8] http://wiki.dbpedia.org/Ontology?v=194q

In Table 1, we show the statistics of maximum and average module size in terms of number of assertions (#Ast.) and size of signature (#Ind.). It can be observed that, in average, modules of all these ontologies are significantly smaller as compared with the entire ABox. In some ontologies, the maximum module size is relatively large, either because there is a great number of assertions of a single individual, or because there are indeed intricate relationships between individuals that may affect classification of each other.

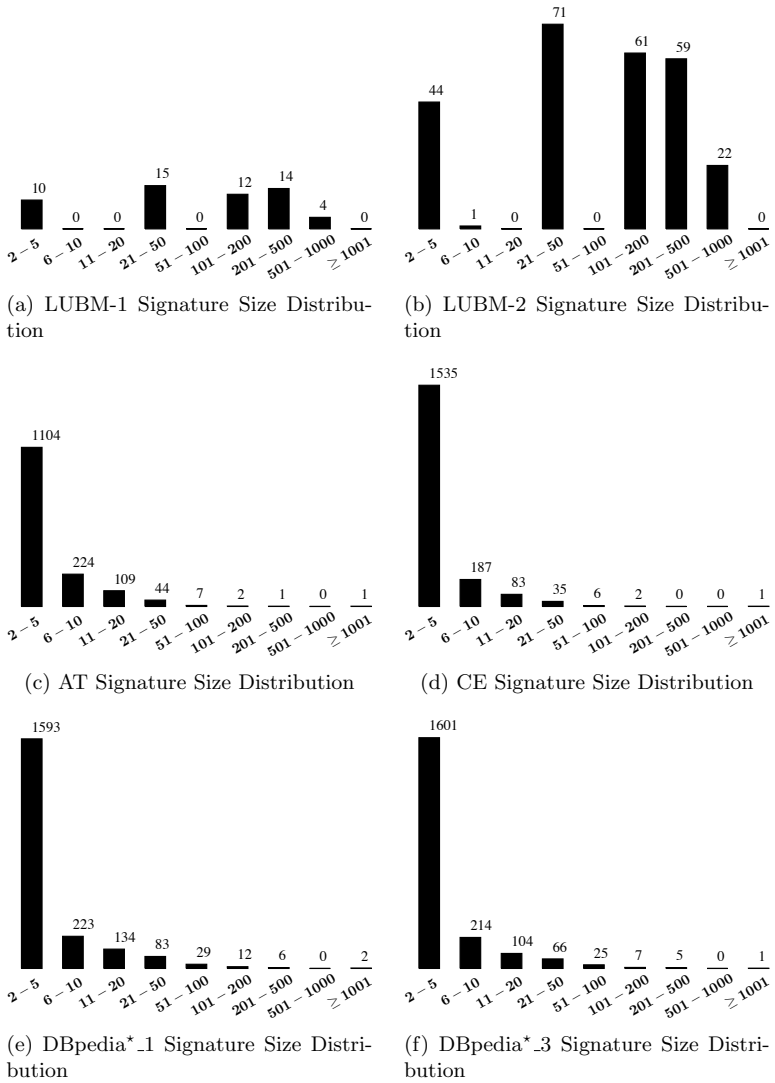
VICODI is a simple ontology that has no property assertions in the ABox with condition (4) satisfied, and thus, the signature of every ABox module is constituted by only a single entity. However, its maximum ABox module consists of more than 8000 assertions for a single individual.

For the biomedical ontologies AT and CE, their maximum ABox modules are large and complex, mainly because: (*i*) the terminological part of these ontologies is highly complex, with 33 out of 55 object properties either functional/inverse functional or used as restrictions for concept definition; and (*ii*) these ontologies also have single individuals that each has a great number of property assertions (e.g. AT has one individual with 8520 assertions). Thus, connections between these individuals by any property assertions satisfying condition (4) result in large ABox modules.

For LUBM-1 and LUBM-2, the sizes of their maximum modules are between of these two categories above, due to moderate complexity of the ontologies and the fact that only 4 out of 25 object properties are used for class definition.

For DBpedia* ontologies, though their terminological part $\mathcal{T}$ is extended from the original $\mathcal{ALF}$ to $\mathcal{SHIQ}$, the terminology is still simple, since the extension to $\mathcal{SHIQ}$ is limited and most of its classes are not defined on restrictions. Nonetheless, maximum ABox modules of these ontologies are even larger, and it is mainly because of those individuals that each has huge number of property assertions (e.g. DBpedia*_1 has one individual with 40773 assertions, and DBpedia*_2 has one individual with 60935 assertions, etc.), and some of their property assertions happen to satisfy the condition (4) that cause a great number of module combinations.

As shown in Table 2, most of the ABox modules (greater than 90%) in these ontologies, are small and simple with no greater than ten assertions or with a single individual in signature. For modules with more than one individual in signature, we plot distributions of signature sizes of these ABox modules in Figure 4 for LUBM-1,LUBM-2, AT, CE, DBpedia*_1 and DBpedia*_3. The X-axis gives the size range and the Y-axis gives the number of modules in each size range. Because of a large span and uneven distribution of module sizes, we use non-uniform size ranges, so that we can have a relatively simple while detailed view of distributions of small, medium, and large modules. It can be observed from the figure that: (*i*) for all these ontologies, the majority of ABox modules are still small with no more than five individuals in signature, (*ii*) LUBMs have more medium modules that have signature size between 50 and 600 due to moderate ontology complexity, and (*iii*) the biomedical ontologies have almost all modules with signature size below 200 but one with very large signature (more than 1,000 individuals), and similar situations are also found in DBpedia* ontologies. Those large ABox modules in both biomedical and DBpedia* ontologies could be caused by complexity of the ontology as discussed above.



(a) LUBM-1 Signature Size Distribution

(b) LUBM-2 Signature Size Distribution

(c) AT Signature Size Distribution

(d) CE Signature Size Distribution

(e) DBpedia*_1 Signature Size Distribution

(f) DBpedia*_3 Signature Size Distribution

Figure 4: Distributions of signature sizes of ABox modules. X-axis is the size range, and Y-axis is the number of modules.

7.2. Optimization and Comparison

While the notion of *Exact ABox Module* defined here can be used to characterize the modularity of ontology ABoxes, in real-world applications, smaller modules are needed for more efficient reasoning and query-answering. Therefore, we discuss following several simple optimization techniques that can be applied to further reduce the size of an ABox module.

As defined previously, an exact ABox module for signature $\mathbf{S}$ consists of *all* justifications for entailments about individuals in $\mathbf{S}$. Considering there may be more than

Table 3: Evaluation and Comparison of Optimized Approximations

	Opt1		Opt2 (Wandelt and Möller, 2012)		Opt3	
Ontology	Max. #Ast./#Ind.	Avg. #Ast./#Ind.	Max. #Ast./#Ind.	Avg. #Ast./#Ind.	Max. #Ast./#Ind.	Avg. #Ast./#Ind.
LUBM-1	773/2	6.8/1.0	773/2	6.8/1.0	732/1	6.8/1.0
LUBM-2	773/2	7.1/1.0	773/2	7.1/1.0	732/1	7.1/1.0
AT	54,561/10,870	6.9/1.7	54,561/10,870	6.9/1.7	54,561/10,870	6.9/1.7
CE	49,914/9,315	7.1/1.7	49,914/9,315	7.1/1.7	49,914/9,315	7.1/1.7
DBpedia*_1	52,539/13,977	2.9/1.1	87,721/15,258	2.9/1.1	228/25	2.7/1.0
DBpedia*_2	108,772/10,568	2.8/1.1	76,386/12,058	2.7/1.1	150/18	2.6/1.0
DBpedia*_3	106,344/32,063	3.1/1.2	63,092/10,385	2.9/1.1	113/29	2.8/1.0
DBpedia*_4	40,760/8,182	2.9/1.1	85,152/11,366	2.8/1.1	171/18	2.7/1.0

one justification for a single entailment (Kalyanpur et al., 2007), an intuition for the strategy to reduce sizes of ABox modules is to exclude redundant justifications for the same entailments.

For example, consider an ontology $\mathcal{K}$ that entails

$$\exists R.B \sqsubseteq A,\ B(b_1) \text{ and } B(b_2),$$

and the ABox contains

$$R(a, b_1) \text{ and } R(a, b_2).$$

To preserve the fact $A(a)$ while excluding redundant justifications for a single entailment, an ABox module for individual a should include either $R(a, b_1)$ with $Just(B(b_1), \mathcal{K})$ or $R(a, b_2)$ with $Just(B(b_2), \mathcal{K})$, but not both. Additionally, instead of considering all $Just(B(b_1), \mathcal{K})$s (*resp.* $Just(B(b_2), \mathcal{K})$s), taking a single justification for $B(b_1)$ (*resp.* for $B(b_2)$) should be sufficient.

Therefore, to exclude redundant justifications for the same entailments from an ABox module, beyond testing the condition (3), i.e.

$$\mathcal{K} \models \geq nR.C_1 \sqcap C_3 \sqsubseteq C_2 \ \wedge\ |R^{\mathcal{K}}(a, C_1)| \geq n$$

for every property assertion $R_0(a, b)$ when computing $\mathcal{M}^{\mathcal{C}}_{\{a\}}$, it is also necessary to consider:

*Case*1. if any justification for $C_2(a)$ has already been added into the module, or

*Case*2. if there is any justification for $C_1(b)$ that can be easily obtained.

In *Case*1, if the module already preserves the fact $C_2(a)$, redundant assertions for the same entailment will not be included; in *Case*2, instead of adding all $Just(C_1(b), \mathcal{K})$s to a's module, we should first consider if there is any one that is easy to compute. Based on these two cases, we implemented three simple optimized approximations for evaluation and comparison:

Opt1. Based on $Case1$, if $C_2(a)$ can be entailed from explicit class assertions of a, modules of a and b will not be merged.

Opt2. Based on $Case2$, if $B(b)$ can be entailed from explicit class assertions of b, instead of merging modules of a and b, only explicit class assertions of b will be added into a's module. If we preprocess every axiom in $\mathcal{T}$ to the form of $\geq nR.C_1 \sqcap C_3 \sqsubseteq C_2$ as described in the proof for Proposition 4.4, this optimization coincides with the one that can be obtained by extending the method in (Wandelt and Möller, 2012).

Opt3. The combination of Opt1 and Opt2.

It is obvious that the first two strategies, Opt1 and Opt2, should have no advantage over each other, and their performances will mostly depend on the actual ontology data to which they are applied. More precisely, if $Case1$ prevails in the ontology, Opt1 is expected to generate smaller ABox modules than that of Opt2, and otherwise if $Case2$ prevails. Nonetheless, Opt3 should always have better or no-worse performance than that of Opt1 and Opt2, as it takes advantage of both of them.

In general, these three optimized approximations are expected to outperform the original approximation and produce smaller ABox modules, since they all employ a DL-reasoner to rule out redundant assertions for the same entailments. Indeed, when an ontology is simple and contains considerable redundant implications, these methods are able to efficiently reduce sizes of ABox modules; while on the other hand, when ontologies are complex or with less explicit redundant information, these methods may not provide significant size reductions.

Evaluation and comparison for these three optimized approximations are shown in Table 3. For LUBM ontologies, all these optimized approximations efficiently reduce the size of ABox modules (especially the maximum one), and particularly, Opt3 reduces signature of every ABox module to include only a single individual. For DBpedia* ontologies, Opt1 or Opt2 achieve only limited size reduction for the maximum ABox modules, while their combination, i.e. Opt3, is able to decrease the size of maximum ABox modules significantly. Nevertheless, for the biomedical ontologies, which are complex and may contain few explicit duplicate entailments, none of the three optimization strategies produced reductions in module size.

7.3. Reasoning with ABox Modules

In this section, we show the efficiency of ontology reasoning gained when reasoning is based on ABox modules (*modular reasoning*), and compare it with that of the complete ABox reasoning. Notice however, the purpose here is not to compare modular reasoning with those developed optimization techniques (e.g. lazy unfolding, and satisfiability reuse etc. (Motik et al., 2007; Horrocks, 2007)) in existing reasoners, since they are in totally different categories and modular reasoning still relies on the reasoning services provided by current state-of-art reasoners. Instead, what we aim to show is in fact the improvement in time efficiency that can be achieved when using the modular reasoning for instance checking or retrieval on top of existing reasoning techniques. The reasoner we used here is HermiT (Motik et al., 2007, 2009), which is one of the fully-fledged OWL 2 reasoners.

For evaluation, we extract ABox modules using the Opt3 discussed in previous section and run the reasoning on each collected ontology: we first randomly select 10 atomic

Table 4: Modular v.s. Complete ABox reasoning

	Modular		Complete	
Ontology	IC Time (ms) Max./Avg.	IR Time Avg.	IC Time (ms) Avg.	IR Time Avg.
LUBM-1	17.00 / 1.93	33.08 (s)	733.21	3.50 (h)
LUBM-2	17.00 / 1.91	150.36 (s)	9,839.22	–
AT	3,120.00 / 344.53	4.09 (h)	11,378.63	–
CE	3,151.00 / 542.60	5.60 (h)	10,336.61	–
DBpedia*_1	1,326.00 / 19.10	1.45 (h)	6,187.01	–
DBpedia*_2	1,497.00 / 20.20	1.67 (h)	6,948.20	–
DBpedia*_3	3,189.00 / 19.89	1.41 (h)	6,087.23	–
DBpedia*_4	1,154.00 / 20.00	1.52 (h)	6,305.41	–

classes that are defined on role restrictions from the testing ontology, and for each one of them we perform the instance checking and retrieval using modular reasoning and complete reasoning, respectively. Table 4 details the reasoning time (not including resource loading time) for both instance checking (IC) and instance retrieval (IR). Particularly, for instance checking using modular reasoning, we show both maximum and average reasoning time over the 10 test cases, since sizes of ABox modules could vary greatly. For instance retrieval, we simply report the average time. Besides, we also set a threshold (24 hours) for the time-out, and if it happens to any one of the test, we simply put a "–" in the corresponding table entry.

Notice that, for fairness the modular reasoning for instance retrieval performed here is not parallelized, but instead running in an arbitrary sequential order of ABox modules. Nevertheless, we can still see the great improvement on time efficiency when using the modular reasoning for instance retrieval as shown in Table 4. For example, the average time for instance checking has been reduced significantly from seconds down to several milliseconds when using the modular reasoning on LUBM(s) and DBpedia*(s), and instance retrieval time on LUBM(s) (respectively on DBpedia*(s)) has been reduced from several hours (respectively several days) down to seconds (respectively less than two hours); even for those biomedical ontologies, the time for instance retrieval is also reduced from more than 130 hours down to less than 6 hours. The reason behind all these improvements is simply that the complexity of those tableau-based reasoning algorithms is up to exponential time w.r.t. the data size for expressive DLs (Donini, 2007; Tobies, 2001); once the data size is cut down, the reasoning time could be reduced significantly, and modular reasoning excels as if it turned the complexity of reasoning from exponential to (approximately) polynomial w.r.t. the data size.

One point to note here is that, when using the modular reasoning for answering (conjunctive) queries, the time for instance retrieval in fact should plus the time that is spent for modularizing the ABox. Nevertheless, as we can see from Table 1, the overhead on average is only around one millisecond for instance checking of each named individual.

8. Discussion and Outlook

In this paper, we have proposed a formal definition of an ABox module, such that each module ensures complete preservation of logical entailments for a give set of individuals. Utilizing this property, scalable object queries over big ontologies can be accomplished, by either conducting an isolated reasoning on a single ABox module when querying information about a particular group of individuals, or by distributing independent reasoning tasks on ABox modules into a cluster of computers when performing instance retrieval or answering conjunctive queries.

To characterize modularity of an ontology ABox, we further defined the notion of Exact ABox Module, which consists of all ABox assertions that are semantically "non-local" to a given signature. For object queries over ontologies, however, the exact ABox module may not be appealing when an ontology consists of a great amount of assertions for redundant logical entailments. Instead, the minimum ABox modules (with the least redundant assertions) would be desired for the most efficient ontology querying; while on the other hand, computation of such minimum ABox modules could be difficult and computationally more expensive than the ABox reasoning itself.

To extract an ABox module, we presented a theoretical approach, which is only aimed to give an exposition of the problem from a theoretical perspective and to provide strategies that are provable theoretically sound. For applicability in realistic ontologies, we provided a simple and tractable approximation, which is straightforward and easy to implement, but may include many unrelated assertions when the ontology is complex, thus resulting in modules that are larger than desired. The undesired bulk of ABox modules could be caused by: (*i*) the syntactic approximation (i.e. (4)) for semantic conditions (i.e. (3)) that could result in many false positives and hence cause the unnecessary combination of ABox modules; (*ii*) the simple approximation of individual equalities, which checks syntactically the possibility of an individual to be instance of some concept $\leq nR.C$ and approximates the number of its R-neighbors; and (*iii*) last but not least, the intrinsic complexity of ontologies that requires assertions to be grouped to preserve complete logical entailments.

Strategies for optimization can provide significant reduction in module size under some conditions, as has been shown in the results section. However, for highly complex ontologies like those biomedical ones, more advanced optimization techniques are demanded. One of the directions for future optimization is to provide a more rigorous approximation for condition (3) and individual equality, and a possible solution is to add affordable semantical verification of individual classifications, which may not only prevent unnecessary module combination but also rule out false individual equalities. Progress in this direction has already been made in our current projects.

References

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z., 2007. Dbpedia: A nucleus for a web of open data, in: Proceedings of ISWC, Springer. pp. 722–735.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (Eds.), 2007. The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F., Stein, L.A., 2004. Owl web ontology language. http://www.w3.org/TR/owl-ref/.

Bhatt, M., Rahayu, W., Soni, S., Wouters, C., 2009. Ontology driven semantic profiling and retrieval in medical information systems. Journal of Web semantics 7, 317–331.

Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R., 2007. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. Journal of Automated reasoning 39, 385–429.

Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U., 2007a. Just the right amount: Extracting modules from ontologies, in: Proceedings of WWW'07, pp. 717–726.

Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U., 2007b. A logical framework for modularity of ontologies, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), AAAI. pp. 298–304.

Cuenca Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U., 2008. OWL 2: The next step for owl. Journal of Web Semantics 6, 309–322.

Dean, J., Ghemawat, S., 2008. Mapreduce: simplified data processing on large clusters. Communications of the ACM 51, 107–113.

Demir, E., Cary, M., Paley, S., Fukuda, K., Lemer, C., Vastrik, I., Wu, G., D'Eustachio, P., Schaefer, C., Luciano, J., et al., 2010. The biopax community standard for pathway data sharing. Nature biotechnology 28, 935–942.

Donini, F., 2007. The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press. chapter Complexity of Reasoning.

Donini, F., Era, A., 1992. Most specific concepts for knowledge bases with incomplete information, in: Proceedings of CIKM, Baltimor, MD. pp. 545–551.

Donini, F., Lenzerini, M., Nardi, D., Schaerf, A., 1994. Deduction in concept languages: From subsumption to instance checking. Journal of logic and computation 4, 423–452.

Du, J., Shen, Y., 2007. Partitioning aboxes based on converting DL to plain datalog, in: Proceedings of Description Logics (DL-07), Citeseer. pp. 251–258.

Fokoue, A., Kershenbaum, A., Ma, L., Schonberg, E., Srinivas, K., 2006. The summary abox: Cutting ontologies down to size, in: Proceedings of ISWC-06, Springer. pp. 343–356.

Glimm, B., Horrocks, I., Lutz, C., Sattler, U., 2008. Conjunctive query answering for the description logic. Journal Artificial Intelligence Research 31, 157–204.

Grosof, B., Horrocks, I., Volz, R., Decker, S., 2003. Description logic programs: Combining logic programs with description logic, in: Proceedings of WWW, ACM. pp. 48–57.

Guo, Y., Heflin, J., 2006. A scalable approach for partitioning owl knowledge bases, in: International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS), Springer. pp. 636–641.

Guo, Y., Pan, Z., Heflin, J., 2005. LUBM: A benchmark for owl knowledge base systems. Journal of Web Semantics 3, 158–182.

Haarslev, V., Möller, R., 2001. RACER system description, in: Proceedings of the First International Joint Conference on Automated Reasoning, Springer. pp. 701–705.

Haarslev, V., Möller, R., 2002. Optimization strategies for instance retrieval, in: Proc. International Workshop on Description Logics (DL).

Haghighi, P., Burstein, F., Zaslavsky, A., Arbon, P., 2012. Development and evaluation of ontology for intelligent decision support in medical emergency management for mass gatherings. Decision Support Systems .

Hitzler, P., Krötzsch, M., Rudolph, S., 2009. Foundations of Semantic Web Technologies. Chapman Hall CRC.

Horrocks, I., 2007. The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press. chapter 9 Implementation and Optimization Techniques.

Horrocks, I., 2008. Ontologies and the semantic web. Communications of the ACM 51, 58–67.

Horrocks, I., Li, L., Turi, D., Bechhofer, S., 2004. The instance store: DL reasoning with large numbers of individuals, in: Proceedings of the Description Logic Workshop (DL-2004), pp. 31–40.

Horrocks, I., Patel-Schneider, P., 2004. Reducing owl entailment to description logic satisfiability. Journal of Web Semantics 1, 345–357.

Horrocks, I., Patel-Schneider, P., Van Harmelen, F., 2003. From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics 1, 7–26.

Horrocks, I., Sattler, U., Tobies, S., 2000. Reasoning with individuals for the description logic SHIQ, in: Proceedings of Conference on Automated Deduction (CADE), Springer. pp. 482–496.

Horrocks, I., Tessaris, S., 2000. A conjunctive query language for description logic aboxes, in: Proceedings of AAAI, pp. 399–404.

Huang, Z., Van Harmelen, F., Teije, A., 2005. Reasoning with inconsistent ontologies, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), AAAI. p. 454.

Hustadt, U., Motik, B., Sattler, U., 2004. Reducing SHIQ- description logic to disjunctive datalog programs, in: Proceedings of KR, pp. 152–162.

Iqbal, A., Shepherd, M., Abidi, S., 2011. An ontology-based electronic medical record for chronic disease management, in: Proceedings of Hawaii International Conference on System Sciences (HICSS), IEEE. pp. 1–10.

Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E., 2007. Finding all justifications of OWL DL entailments, in: Proceedings of International Semantic Web Conference (ISWC), pp. 267–280.

Lee, C., Wang, M., Chen, J., 2008. Ontology-based intelligent decision support agent for cmmi project monitoring and control. International Journal of Approximate Reasoning 48, 62–76.

Motik, B., Sattler, U., 2006. A comparison of reasoning techniques for querying large description logic aboxes, in: Proceedings of LPAR'06, pp. 227–241.

Motik, B., Sattler, U., Studer, R., 2005. Query answering for OWL-DL with rules. Journal of Web Semantics 3, 41–60.

Motik, B., Shearer, R., Horrocks, I., 2007. Optimized Reasoning in Description Logics using Hypertableaux, in: Pfenning, F. (Ed.), Proceedings of Conference on Automated Deduction (CADE), Springer, Bremen, Germany. pp. 67–83.

Motik, B., Shearer, R., Horrocks, I., 2009. Hypertableau reasoning for description logics. Journal of Artificial Intelligence Research 36, 165–228.

Nebel, B., 1990. Reasoning and revision in hybrid representation systems. volume 422. Springer-Verlag Germany.

Ortiz, M., Calvanese, D., Eiter, T., 2008. Data complexity of query answering in expressive description logics via tableaux. Journal of Automated Reasoning 41, 61–98.

Royer, V., Quantz, J., 1993. Deriving Inference Rules for Description Logics: a Rewriting Approach into Sequent Calculi. Technical Report. Technische Universitaet Berlin. Germany.

Schaerf, A., 1994. Reasoning with individuals in concept languages. Data and Knowledge Engineering 13, 141–176.

Schmidt-Schauß, M., Smolka, G., 1991. Attributive concept descriptions with complements. Artificial intelligence 48, 1–26.

Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y., 2007. Pellet: A practical owl-dl reasoner. Journal of Web Semantics 5, 51 – 53.

Tobies, S., 2001. Complexity Results and Practical Algorithms for Logics in Knowledge Representation. Ph.D. thesis. RWTH Aachen.

Tsarkov, D., Horrocks, I., 2006. FaCT++ description logic reasoner: System description. Automated Reasoning , 292–297.

Visser, U., Abeyruwan, S., Vempati, U., Smith, R., Lemmon, V., Schürer, S., 2011. BioAssay Ontology (BAO): a semantic description of bioassays and high-throughput screening results. BMC bioinformatics 12, 257.

Wandelt, S., Möller, R., 2012. Towards abox modularization of semi-expressive description logics. Applied Ontology 7, 133–167.

Williams, G., Weaver, J., Atre, M., Hendler, J., 2010. Scalable reduction of large datasets to interesting subsets. Journal of Web Semantics 8, 365–373.