Just the Right Amount: Extracting Modules from Ontologies

Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov and Ulrike Sattler
The University of Manchester
School of Computer Science
Manchester, M13 9PL, UK
{bcg,horrocks,ykazakov,sattler}@cs.man.ac.uk

ABSTRACT

The ability to extract meaningful fragments from an ontology is key for ontology re-use. We propose a definition of a module that guarantees to completely capture the meaning of a given set of terms, i.e., to include all axioms relevant to the meaning of these terms, and study the problem of extracting minimal modules. We show that the problem of determining whether a subset of an ontology is a module for a given vocabulary is undecidable even for rather restricted sub-languages of OWL DL. Hence we propose two "approximations", i.e., alternative definitions of modules for a vocabulary that still provide the above guarantee, but that are possibly too strict, and that may thus result in larger modules: the first approximation is semantic and can be checked using existing DL reasoners; the second is syntactic, and can be computed in polynomial time. Finally, we report on an empirical evaluation of our syntactic approximation which demonstrates that the modules we extract are surprisingly small.

Categories and Subject Descriptors

I.2.4 [**Knowledge Representation Formalisms and Methods**]: Miscellaneous

General Terms

Algorithms

Keywords

Ontologies, Description Logics, OWL, Semantic Web

1. INTRODUCTION

The design, maintenance, reuse, and integration of ontologies are highly complex tasks—especially for ontologies formulated in a logic-based language such as OWL. Like software engineers, "ontology engineers" need to be supported by tools and methodologies that help them to minimise the introduction of errors, i.e., to ensure that ontologies have appropriate consequences. In order to develop this support, important notions from software engineering, such as *module*, *black-box behavior*, and *controlled interaction*, need to be adapted so as to take into account the fact that an OWL ontology is, in essence, a logical theory; due to the expressive power of OWL, this turns out to be difficult.

In earlier work [3], we have studied modularity in the context of *collaborative ontology development* and *controlled integration*, and defined what it means for an ontology we are developing to be safely integrated with a "foreign" ontology; roughly speaking, such an integration is safe if it does not change the meaning of the terms in the foreign ontology.

In this paper, we focus on the use of modularity to support the *partial reuse* of ontologies: continuing with the above integration scenario, as a next step, we would like to *extract*, from the foreign ontology, a small fragment that captures the meaning of the terms we use in our ontology. For example, when building an ontology describing research projects, we may use terms such as $\mathsf{Cystic\_Fibrosis}$ and $\mathsf{Genetic\_Disorder}$ in our descriptions of medical research projects. In order to improve the precision of our ontology, we may want to add more detail about the meaning of these terms; for reasons of cost and accuracy, we would prefer to do this by reusing information from a medical ontology. Such ontologies are, however, typically very large, and importing the whole ontology would make the consequences of the additional information costly to compute and difficult for our ontology engineers (who are not medical experts) to understand. Thus, in practice, we need to extract a module that includes just the relevant information. Ideally, this module should be *as small as possible* while still *guaranteeing* to capture the meaning of the terms used; that is, when answering arbitrary queries against our projects ontology, importing the module would give us *exactly the same answers* as if we had imported the whole medical ontology. In this case, importing the module instead of the whole ontology will have no observable effect on our ontology—apart from allowing for more efficient reasoning.

With respect to the efficiency of reasoning, although modern reasoners perform well with realistic ontologies, reasoning with large ontologies is often still challenging. Even if the ontology under consideration can be processed, it may still be the case that the processing time involved is too high for ontology engineering, where fast response under changes in the ontology is required, or for deployment in applications, where fast response to queries is required. The ability to extract modules in the sense described above would address both these problems: it would allow us to identify a (hopefully small) part of the ontology that is affected by a given change or that is sufficient to answer a given query—and then to reason over this part only without losing any consequences.

Finally, the techniques we develop in this paper might also be useful for analyzing ontologies and identifying design problems. For example, they can be used to identify inter-dependencies between classes in the ontology, and to distinguish classes with "shallow" definitions from those that are defined in great detail. Moreover, ontologies that facilitate the extraction of small modules are likely to be easier to develop and maintain than those that do not.

The contributions of this paper are as follows:

1. We propose a definition of a *module* $\mathcal{Q}_1$ within a given on-

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

tology $\mathcal{Q}$ for a given vocabulary $\mathbf{S}$.

2. We take the above definition as a starting point, and investigate the problem of computing minimal modules. We show that none of the reasonable variants of this problem is solvable in general already for rather restricted sub-languages of OWL DL. In fact, it is even not possible to determine whether a subset $\mathcal{Q}_1$ of an ontology $\mathcal{Q}$ is a module in $\mathcal{Q}$ for $\mathbf{S}$.

3. Given these negative results, we propose two "approximations", i.e., alternative definitions of a module that still guarantee to completely capture the meaning of the terms in $\mathbf{S}$, but that are possibly too strict, and that may thus result in larger modules; these approximations are based on the notion of *locality* of an ontology with respect to a vocabulary, as first introduced in [3]. The first approximation is semantic, and can be checked using existing OWL reasoners; the second one is a restriction of the first one which can be computed in polynomial time. We propose an algorithm for computing the smallest module for each of these approximations.

4. Finally, we describe our implementation and present our experimental results on a set of real-world ontologies of varying size and complexity. We show that, using our syntactic approximation, we obtain modules that are much smaller than the ones computed using existing techniques, but still sufficient to capture the meaning of the specified vocabulary.

This paper comes with a Technical Report [4], available online, which contains the complete proofs for the results we discuss here.

2. PRELIMINARIES

In this section we introduce description logics (DLs) [2] which underly modern ontology languages, such as OWL DL. The *syntax* of a description logic L is given by a signature and a set of constructors. A *signature* (or *vocabulary*) $\mathbf{S}$ of a DL is the (disjoint) union of a set $\mathbf{C}$ of *atomic concepts* $(A, B, \dots)$ representing sets of elements, a set $\mathbf{R}$ of *atomic roles* $(r, s, \dots)$ representing binary relations between elements, and a set $\mathbf{I}$ of *individuals* $(a, b, c, \dots)$ representing single elements. Every DL provides *constructors* for defining the set $\mathsf{Rol}(\mathbf{S})$ of (general) *roles* $(R, S, \dots)$, the set $\mathsf{Con}(\mathbf{S})$ of (general) *concepts* $(C, D, \dots)$, and the set $\mathsf{Ax}(\mathbf{S})$ of *axioms* $(\alpha, \beta, \dots)$ for a signature $\mathbf{S}$ which is a union of *role axioms* (RBox), *terminological axioms* (TBox) and *assertions* (ABox).

$\mathcal{EL}$ [1] is a simple description logic which allows one to construct complex concepts using *conjunction* $C_1 \sqcap C_2$ and *existential restriction* $\exists R.C$ starting from atomic concepts A, roles R and the *bottom concept* $\bot$. $\mathcal{EL}$ provides no role constructors and no role axioms; thus, every role R in $\mathcal{EL}$ is atomic. The TBox axioms of $\mathcal{EL}$ can be either *concept definitions* $A \equiv C$ or *general concept inclusion axioms* (GCIs) $C_1 \sqsubseteq C_2$. $\mathcal{EL}$ assertions are either *concept assertions* $a : C$ or *role assertions* $r(a, b)$.

The *basic description logic* $\mathcal{ALC}$ [13] is obtained from $\mathcal{EL}$ by adding *complement of concepts* $\neg C$. We introduce some additional constructors as abbreviations: the *top concept* $\top$ is a shortcut for $\neg\bot$, the *disjunction of concepts* $C_1 \sqcup C_2$ stands for $\neg(\neg C_1 \sqcap \neg C_2)$, and the *value restriction* $\forall R.C$ stands for $\neg(\exists R.\neg C)$.

$\mathcal{S}$ is an extension of $\mathcal{ALC}$ where, additionally, some atomic roles can be declared to be *transitive* using a role axiom $\mathsf{Trans}(r)$.

Further extensions of description logics include *inverse roles* r^- (indicated by appending a letter $\mathcal{I}$), *role inclusion axioms* (RIs) also called *role hierarchies* $R_1 \sqsubseteq R_2$ $(+\mathcal{H})$, *functional roles* $\mathsf{Funct}(R)$ $(+\mathcal{F})$, *number restrictions* $(\geqslant n\, S)$ $(+\mathcal{N})$, *qualified number restrictions* $(\geqslant n\, S.C)$[1] $(+\mathcal{Q})$, and *nominals* $\{a\}$ $(+\mathcal{O})$. Nominals make it possible to construct a concept representing a singleton set $\{a\}$ (a *nominal* concept) from an individual a. These extensions can be used in different combinations, for example $\mathcal{ALCO}$ is an extension of $\mathcal{ALC}$ with nominals; $\mathcal{SHIQ}$ is an extension of $\mathcal{S}$ with role hierarchies, inverse roles and qualified number restrictions; and $\mathcal{SHOIQ}$ is the DL that uses all the constructors and axiom types we have presented.

Modern ontology languages, such as OWL, are based on description logics and, to a certain extent, are syntactic variants thereof. In particular, OWL DL corresponds to $\mathcal{SHOIN}$ [11]. In this paper, we assume an *ontology* $\mathcal{O}$ *based on a description logic* L to be a set of axioms in L. The *signature of an ontology* $\mathcal{O}$ (*of an axiom* α) is the set $\mathsf{Sig}(\mathcal{O})$ ($\mathsf{Sig}(\alpha)$) of atomic concepts, atomic roles and individuals that occur in $\mathcal{O}$ (respectively in α).

The main reasoning task for ontologies is *query answering*: given an ontology $\mathcal{O}$ and an axiom α, check if $\mathcal{O}$ implies α.

The logical entailment $\models$ is defined using the *usual Tarski-style semantics* for description logics as follows. Given a signature $\mathbf{S} = \mathbf{R} \cup \mathbf{C} \cup \mathbf{I}$, an $\mathbf{S}$-*interpretation* $\mathcal{I}$ is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set, called the *domain* of the interpretation, and $\cdot^{\mathcal{I}}$ is the *interpretation function* that assigns: to every $A \in \mathbf{C}$ a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, to every $r \in \mathbf{R}$ a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to every $a \in \mathbf{I}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$.

The interpretation function $\cdot^{\mathcal{I}}$ is extended to complex roles and concepts via DL-constructors in the standard way (see [2] or [4] for details). The *satisfaction* relation $\mathcal{I} \models \alpha$ between an interpretation $\mathcal{I}$ and a DL axiom α (read as $\mathcal{I}$ *satisfies* α) is also standard and can be found in [2] or [4]. An interpretation $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$ if $\mathcal{I}$ satisfies all axioms in $\mathcal{O}$. An ontology $\mathcal{O}$ *implies* an axiom α (written $\mathcal{O} \models \alpha$) if $\mathcal{I} \models \alpha$ for every model $\mathcal{I}$ of $\mathcal{O}$. An axiom α is a *tautology* if it is implied by the empty ontology.

Let $\mathbf{S}_1, \mathbf{S}$ be signatures such that $\mathbf{S}_1 \subseteq \mathbf{S}$. The *restriction of an* $\mathbf{S}$-*interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *to* $\mathbf{S}_1$ is an interpretation $\mathcal{I}|_{\mathbf{S}_1} = (\Delta^{\mathcal{I}_1}, \cdot^{\mathcal{I}_1})$ over $\mathbf{S}_1$ such that $\Delta^{\mathcal{I}_1} = \Delta^{\mathcal{I}}$ and $X^{\mathcal{I}_1} = X^{\mathcal{I}}$ for every $X \in \mathbf{S}_1$. An *expansion of an* $\mathbf{S}_1$-*interpretation* $\mathcal{I}_1$ *to* $\mathbf{S}$ is an $\mathbf{S}$-interpretation $\mathcal{I}$ such that $\mathcal{I}|_{\mathbf{S}_1} = \mathcal{I}_1$. A *trivial expansion of an* $\mathbf{S}_1$-*interpretation* $\mathcal{I}_1$ *to* $\mathbf{S}$ is an expansion of $\mathcal{I}_1$ to $\mathbf{S}$ such that $X^{\mathcal{I}} = \emptyset$ for every atomic concept and atomic role $X \in \mathbf{S} \setminus \mathbf{S}_1$.

3. MODULES FOR KNOWLEDGE REUSE

For exposition, suppose that an ontology engineer wants to build an ontology about research projects. The ontology defines different types of projects according to the research topics they focus on. Suppose that the ontology engineer defines two concepts Genetic_Disorder_Project and Cystic_Fibrosis_EUProject in his ontology $\mathcal{P}$. The first one describes projects about genetic disorders; the second one describes European projects about cystic fibrosis, as given by the axioms P1 and P2 in Figure 1.

The ontology engineer is supposed to be an expert on research projects: he knows, for example, that a EUProject is a Project (axiom P3). He is unfamiliar, however, with most of the topics the projects cover and, in particular, with the terms Cystic_Fibrosis and Genetic_Disorder mentioned in P1 and P2. In this case, he decides to reuse the knowledge about these subjects from a well-established and widely-used medical ontology

The most straightforward way to reuse these concepts is to import the medical ontology. This may be, however, a large ontology, which deals with other matters in which the ontology engineer is not interested, such as genes, anatomy, surgical techniques, etc.

[1]the dual constructors $(\leqslant n\, S)$ and $(\leqslant n\, S.C)$ are abbreviations for $\neg(\geqslant n\, S.\neg C)$ and $\neg(\geqslant n\, S.\neg C)$, respectively

	Ontology of medical research projects $\mathcal{P}$:
P1	$\mathsf{Genetic\_Disorder\_Project} \equiv \mathsf{Project} \sqcap \sqcap \exists \mathsf{has\_Focus}.\underline{\mathsf{Genetic\_Disorder}}$
P2	$\mathsf{Cystic\_Fibrosis\_EUProject} \equiv \mathsf{EUProject} \sqcap \sqcap \exists \mathsf{has\_Focus}.\underline{\mathsf{Cystic\_Fibrosis}}$
P3	$\mathsf{EUProject} \sqsubseteq \mathsf{Project}$
	Ontology of medical terms $\mathcal{Q}$:
M1	$\underline{\mathsf{Cystic\_Fibrosis}} \equiv \mathsf{Fibrosis} \sqcap \exists \mathsf{located\_In}.\mathsf{Pancreas} \sqcap \sqcap \exists \mathsf{has\_Origin}.\mathsf{Genetic\_Origin}$
M2	$\mathsf{Genetic\_Fibrosis} \equiv \mathsf{Fibrosis} \sqcap \sqcap \exists \mathsf{has\_Origin}.\mathsf{Genetic\_Origin}$
M3	$\mathsf{Fibrosis} \sqcap \exists \mathsf{located\_In}.\mathsf{Pancreas} \sqsubseteq \mathsf{Genetic\_Fibrosis}$
M4	$\mathsf{Genetic\_Fibrosis} \sqsubseteq \underline{\mathsf{Genetic\_Disorder}}$
M5	$\mathsf{DEFBI\_Gene} \sqsubseteq \mathsf{Immuno\_Protein\_Gene} \sqcap \sqcap \exists \mathsf{associated\_With}.\underline{\mathsf{Cystic\_Fibrosis}}$

Figure 1: Reusing medical terminology for an ontology on medical research projects

Ideally, one would like to extract a (hopefully small) fragment of the medical ontology (a *module*) that describes in detail the concepts we are reusing in our ontology. Intuitively, importing the module $\mathcal{Q}_1$ into $\mathcal{P}$ instead of the full ontology $\mathcal{Q}$ should have no impact on the modeling of the ontology $\mathcal{P}$.

Continuing with the example, suppose that $\mathsf{Cystic\_Fibrosis}$ and $\mathsf{Genetic\_Disorder}$ are described in an ontology $\mathcal{Q}$ containing axioms M1-M5 in Figure 1. If we include in the module $\mathcal{Q}_1$ just the axioms that mention $\mathsf{Cystic\_Fibrosis}$ or $\mathsf{Genetic\_Disorder}$, namely M1, M4 and M5, we lose the following dependency:

$$\mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder} \qquad (1)$$

The concept inclusions $\mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$ follow from M1-M5, but not from M1, M4, M5, since the dependency $\mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Fibrosis}$ does not hold after removing M2 and M3. The dependency (1), however, is crucial for our ontology $\mathcal{P}$ as it (together with the axiom $\mathsf{EUProject} \sqsubseteq \mathsf{Project}$) implies the following axiom:

$$\mathsf{Cystic\_Fibrosis\_EUProject} \sqsubseteq \mathsf{Genetic\_Disease\_Project} \qquad (2)$$

This means, in particular, that all the projects annotated with $\mathsf{Cystic\_Fibrosis\_EUProject}$ must be included in the answer for a query on $\mathsf{Genetic\_Disease\_Project}$. Consequently, importing a part of $\mathcal{Q}$ containing only axioms that mention the terms used in $\mathcal{P}$ instead of $\mathcal{Q}$ results in an underspecified ontology. We stress that the ontology engineer might be unaware of dependency (2), even though this dependency concerns the concepts of his primary scope.

The example above suggests that the central requirement for a module $\mathcal{Q}_1 \subseteq \mathcal{Q}$ to be reused in our ontology $\mathcal{P}$ is that $\mathcal{P} \cup \mathcal{Q}_1$ should yield the *same* logical consequences in the vocabulary of $\mathcal{P}$ as $\mathcal{P} \cup \mathcal{Q}$ does. Note that, as seen in the example, this requirement does not force us to include in $\mathcal{Q}_1$ all the axioms in $\mathcal{Q}$ that mention the vocabulary to be reused, nor does it imply that the axioms in $\mathcal{Q}$ that do not mention this vocabulary should be omitted.

Based on the discussion above, we formalize our first notion of a *module* as follows:

Definition 1 [Module]. Let $\mathcal{Q}_1 \subseteq \mathcal{Q}$ be two ontologies and $\mathbf{S}$ a signature. We say that $\mathcal{Q}_1$ is an $\mathbf{S}$-*module in* $\mathcal{Q}$ w.r.t. a language L, if for every ontology $\mathcal{P}$ and every axiom α expressed in L with $\mathsf{Sig}(\mathcal{P} \cup \{\alpha\}) \cap \mathsf{Sig}(\mathcal{Q}) \subseteq \mathbf{S}$, we have $\mathcal{P} \cup \mathcal{Q} \models \alpha$ iff $\mathcal{P} \cup \mathcal{Q}_1 \models \alpha$.

In Definition 1 the signature $\mathbf{S}$ acts as the *interface* signature between $\mathcal{P}$ and $\mathcal{Q}$ in the sense that it contains the symbols that $\mathcal{P}$ and α may share with $\mathcal{Q}$. It is also important to realize that there are two free parameters in Definition 1, namely the ontology $\mathcal{P}$ and the axiom α. Both $\mathcal{P}$ and α are formulated in some ontology language L, which might not necessarily be a sub-language of OWL DL.

Fixing the language L in which $\mathcal{P}$ and α can be expressed is essential in Definition 1 since it may well be the case that $\mathcal{Q}_1$ is a module in $\mathcal{Q}$ w.r.t. a language L_1, but not w.r.t. L_2. Fixing L, however, is not always reasonable. If $\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$, it should always be possible to replace $\mathcal{Q}$ with $\mathcal{Q}_1$ independently of the particular language in which $\mathcal{P}$ and α are expressed. In fact, we may extend our ontology $\mathcal{P}$ with a set of Horn rules, or extend our query language to support arbitrary conjunctive queries. In any case, extending the ontology language for $\mathcal{P}$ and the query language for α should not prevent $\mathcal{Q}_1$ from being a module in $\mathcal{Q}$.

It is therefore convenient to formulate a more general notion of a module which abstracts from the particular language under consideration; that is, we say that $\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$ iff it is an $\mathbf{S}$-module in $\mathcal{Q}$, according to Definition 1 for *every* language with Tarski-style set-theoretic semantics L.

In our knowledge reuse scenario, small modules are preferred over large modules. Therefore, it makes sense to focus only on minimal modules. We say that $\mathcal{Q}_1$ is a *minimal* $\mathbf{S}$-*module in* $\mathcal{Q}$ if there is no $\mathcal{Q}_2 \subsetneq \mathcal{Q}_1$ that is also an $\mathbf{S}$-module in $\mathcal{Q}$. In our example from Figure 1, there are two minimal $\mathbf{S}$-modules $\mathcal{Q}_1 = \{\mathrm{M1}, \mathrm{M2}, \mathrm{M4}\}$ and $\mathcal{Q}_2 = \{\mathrm{M1}, \mathrm{M3}, \mathrm{M4}\}$: if we remove any axiom from them, the dependency (2) will no longer hold. Hence minimal modules are not necessarily unique. While in some cases it is reasonable to extract all minimal modules, in others it may suffice to extract just one. Thus, given $\mathcal{Q}$ and $\mathbf{S}$, the following tasks are of interest:

T1. compute *all* minimal $\mathbf{S}$-modules in $\mathcal{Q}$
T2. compute *some* minimal $\mathbf{S}$-module in $\mathcal{Q}$ (3)

In fact, it can be shown [4] that these tasks are inter-reducible; that is, an algorithm that solves T1 can be used to solve T2 and vice-versa.

PROPOSITION 1. *Tasks T1 and T2 from* (3) *are inter-reducible.*

Let us now consider the axioms M1–M4. These axioms occur in both minimal $\mathbf{S}$-modules $\mathcal{Q}_1$ and $\mathcal{Q}_2$; thus, they are , in a certain sense, essential for dependency (2). In certain situations, one can be interested in computing just the set $\mathcal{Q}_e$ of such essential axioms, instead of computing all minimal modules. This is the case, for example, if the modeler wants to compute a module that is "safe" under removal of axioms: if we remove M2 from $\mathcal{Q}$, then $\mathcal{Q}_1' = \mathcal{Q}_1 \setminus \mathrm{M2} = \{\mathrm{M1}, \mathrm{M4}\}$ is no longer an $\mathbf{S}$-module for the updated ontology $\mathcal{Q}' := \mathcal{Q} \setminus \{\mathrm{M2}\}$ since the dependency (2) is lost, but $\mathcal{Q}_e' := \mathcal{Q}_e \setminus \{\mathrm{M2}\}$ is still a module in $\mathcal{Q}$. This example suggests the following definition:

Definition 2 [Essential Axiom]. Given a signature $\mathbf{S}$ and an ontology $\mathcal{Q}$, we say that an axiom $\alpha \in \mathcal{Q}$ is $\mathbf{S}$-*essential in* $\mathcal{Q}$ w.r.t. L if α belongs to some minimal $\mathbf{S}$-module in $\mathcal{Q}$ w.r.t. L.

Hence, the following task may also be of interest:

T3. compute *the union* of all minimal $\mathbf{S}$-modules in $\mathcal{Q}$, which is the set of all $\mathbf{S}$-essential axioms in $\mathcal{Q}$ (4)

Obviously, task T3 is not harder then task T1: a procedure for computing all minimal modules can be used in a straightforward way to compute the union of these minimal modules.

In the last few years, numerous techniques for extracting fragments of ontologies for knowledge reuse purposes have been developed. Most of these techniques rely on syntactically traversing the axioms in the ontology and employ various heuristics for determining which axioms are relevant and which are not.

An example of such a procedure is the algorithm implemented in the PROMPT-FACTOR tool [10]. Given a signature $\mathbf{S}$ and an ontology $\mathcal{Q}$, the algorithm retrieves a fragment $\mathcal{Q}_1 \subseteq \mathcal{Q}$ as follows: first, the axioms in $\mathcal{Q}$ that mention any of the symbols in $\mathbf{S}$ are added to $\mathcal{Q}_1$; second, $\mathbf{S}$ is expanded with the symbols in $\mathsf{Sig}(\mathcal{Q}_1)$. These steps are repeated until a fixpoint is reached. In our example, the axioms M1-M5 would be retrieved.

Another example is the algorithm in [14], which was used for segmentation of the medical ontology GALEN [12]. Given a signature $\mathbf{S}$ and an ontology $\mathcal{Q}$, the algorithm adds to $\mathcal{Q}_1$ all definitions $A \equiv C$ for symbols in $\mathbf{S}$, expands $\mathbf{S}$ with symbols in $\mathsf{Sig}(\mathcal{Q}_1)$, and then repeats these steps again until a fixpoint is reached. The main idea of this algorithm is to prune irrelevant axioms by traversing the class hierarchy only "upwards" and across existential restrictions. Unfortunately, this algorithm does not detect other dependencies, in particular those expressed by GCIs. In our example, when initialized with $\mathsf{Cystic\_Fibrosis}$ and $\mathsf{Genetic\_Disorder}$, the algorithm retrieves only the axiom M1 and the dependency (1) is lost.

None of these algorithms is appropriate in general for extracting modules according to Definition 1. On the one hand, the PROMPT-FACTOR algorithm extracts many unnecessary axioms (such as M5 in our case) whereas, on the other hand, the segmentation algorithm from [14] misses essential axioms (like M2, M3 and M4).

In our example, the PROMPT-FACTOR algorithm would extract a module (though not a minimal one). In general, however, this is also not the case. For example, consider an ontology $\mathcal{Q} = \{\top \sqsubseteq \{a\},\ A \sqsubseteq B\}$ and $\alpha = (A \sqsubseteq \forall r.A)$. It is easy to see that $\mathcal{Q}$ admits only single element models and α is satisfied in every such a model; that is, $\mathcal{Q} \models \alpha$. The PROMPT-FACTOR algorithm extracts in this case $\mathcal{Q}_1 = \{A \sqsubseteq B\}$, which does not imply α.

The main problem with these algorithms is that they ignore the semantics of the ontologies. As a consequence, they may extract, on the one hand, irrelevant axioms and, on the other hand, they might miss essential axioms. These algorithms, however, was intended to extract modules in accordance to a formal collection of requirements. These procedures were intended to extract "relevant parts" of ontologies which are "likely to be related" to the given signature, but they do not guarantee the correctness of the results. Correctness, however, is the primary requirement for the procedures we present in this paper.

3.1 Modules and Conservative Extensions

The notion of a module is closely related to the notion of a conservative extension which has been used to characterize formal requirements in ontology integration tasks [7, 5, 3, 9]. In the literature we can find at least two different notions of conservative extensions in the context of ontologies [9]:

Definition 3 [Conservative Extensions]. Let $\mathcal{Q}_1 \subseteq \mathcal{Q}$ be two ontologies, $\mathbf{S}$ a signature and L a logic.

We say that $\mathcal{Q}$ is a *deductive* $\mathbf{S}$*-conservative extension* of $\mathcal{Q}_1$ w.r.t. L, if for every axiom α over L with $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$, we have $\mathcal{Q} \models \alpha$ iff $\mathcal{Q}_1 \models \alpha$.

We say that $\mathcal{Q}$ is a *model* $\mathbf{S}$*-conservative extension* of $\mathcal{Q}_1$ if, for every model $\mathcal{I}_1$ of $\mathcal{Q}_1$, there exists a model $\mathcal{I}$ of $\mathcal{Q}$ such that $\mathcal{I}|_{\mathbf{S}} = \mathcal{I}_1|_{\mathbf{S}}$.

Intuitively, an ontology $\mathcal{Q}$ is a deductive conservative extension of an ontology $\mathcal{Q}_1 \subseteq \mathcal{Q}$ for a signature $\mathbf{S}$ iff every logical consequence α of $\mathcal{Q}$ constructed using only symbols from $\mathbf{S}$ is already a consequence of $\mathcal{Q}_1$. In other words, the additional axioms in $\mathcal{Q}$ do not add new logical consequences over the vocabulary $\mathbf{S}$. Analogously to modules, the notion of a deductive conservative extension depends on the logic L in which $\mathcal{Q}$ and α are expressed.

In contrast, model conservative extensions are not defined in terms of logical entailment, but using the models directly. Intuitively, an ontology $\mathcal{Q}$ is a model conservative extension of $\mathcal{Q}_1 \subseteq \mathcal{Q}$ if every model of $\mathcal{Q}_1$ can be expanded to a model of $\mathcal{Q}$ by interpreting new symbols and leaving the interpretations of the old symbols unchanged.

The notion of semantic conservative extension is strictly stronger than the syntactic one [9] since it does not depend on expressivity of the logic. That is, if $\mathcal{Q}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$, it is also a deductive $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$, but not necessarily vice versa.

Example 1. Let $\mathcal{Q}$ be the ontology consisting of axioms $\mathrm{M1} - \mathrm{M5}$ in Figure 1. Let $\mathcal{Q}_1$ consist of the axioms $\mathrm{M1} - \mathrm{M4}$ and let $\mathbf{S} = \{\mathsf{Cystic\_Fibrosis}, \mathsf{Genetic\_Disorder}\}$. We show that $\mathcal{Q}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$ and, hence, also a deductive conservative extension of $\mathcal{Q}_1$.

Let $\mathcal{I}_1$ be an arbitrary model of $\mathcal{Q}_1$. We demonstrate that we can always construct a model $\mathcal{I}$ of $\mathcal{Q}$ which interprets the symbols from $\mathbf{S}$ in the same way as $\mathcal{I}_1$ does, i.e. $\mathcal{I}|_{\mathbf{S}} = \mathcal{I}_1|_{\mathbf{S}}$.

Indeed, let $\mathcal{I}$ be identical to $\mathcal{I}_1$ except for the interpretation of the atomic concepts $\mathsf{DEFBI\_Gene}$ and $\mathsf{Immuno\_Protein\_Gene}$, and the atomic role $\mathsf{associatedWith}$, all of which we interpret in $\mathcal{I}$ as the empty set. Note that these atomic concepts and this atomic role do not occur in $\mathcal{Q}_1$. Hence, $\mathcal{I}$ interprets the concepts in $\mathcal{Q}_1$ exactly like $\mathcal{I}_1$, and so $\mathcal{I}$ is a model of $\mathcal{Q}_1$. Furthermore, $\mathcal{I}$ is a model of M5 since the concepts on the left-hand-side and the right-hand-side of this axiom are both interpreted as the empty set. Thus, $\mathcal{Q}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$.

Although Definition 1 is close to the notion of deductive conservative extension, there are two important differences. First, in the definition of deductive conservative extension, the logical consequences are considered only w.r.t. the ontologies $\mathcal{Q}$ and $\mathcal{Q}_1$ of interest whereas, in our definition of module, all the possible ontologies $\mathcal{P}$ in which the module can be used are taken into account. Second, in the definition of deductive conservative extension, the signature of α is required to be a subset of $\mathbf{S}$ whereas, in our definition of module, only the common part of $\alpha \cup \mathcal{P}$ and $\mathcal{Q}$ is required to be a subset of $\mathbf{S}$. Despite these differences, the two notions of conservative extensions are related to our notion of module:

PROPOSITION 2. *Let $\mathcal{Q}_1 \subseteq \mathcal{Q}$ be two ontologies. Then:*

1. *If $\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$ w.r.t. L then $\mathcal{Q}$ is a deductive $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$ w.r.t. L;*

2. *If $\mathcal{Q}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$ then $\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$ for every ontology language L with Tarski-style set-theoretic semantics.*

Proposition 2 shows that our notion of module stays "in between" the two notions of conservative extensions. In particular, by applying Property 2 to Example 1, we can show that the axioms M1-M4 in Figure 1 constitute a module in the ontology $\mathcal{Q}$, consisting of M1-M5. The converse of Property 1 in Proposition 2, however, does not hold in general:

Example 2. Let $\mathcal{Q}_1 = \{\}$, $\mathcal{Q} = \{\top \sqsubseteq \exists R.A\}$ and $\mathbf{S} = \{A\}$. It is easy to see that $\mathcal{Q}$ is a deductive $\mathbf{S}$-conservative extension of

$\mathcal{Q}_1$ w.r.t. $\mathcal{ALC}$. Indeed, every $\mathcal{ALC}$-axiom $\alpha = (C_1 \sqsubseteq C_2)$ over $\mathbf{S} = \{A\}$, is equivalent in $\mathcal{ALC}$ to either $\top \sqsubseteq \top$, $\top \sqsubseteq \bot$, $\top \sqsubseteq A$ or $A \sqsubseteq \bot$, which are indistinguishable by $\mathcal{Q}_1$ and $\mathcal{Q}$—that is, the axiom is implied by $\mathcal{Q}_1$ iff it is implied by $\mathcal{Q}$.

However, $\mathcal{Q}_1$ is not an $\mathbf{S}$-module in $\mathcal{Q}$. To demonstrate this, consider an $\mathcal{ALC}$-ontology $\mathcal{P} = \{A \sqsubseteq \bot\}$, which is constructed over $\mathbf{S}$. It is easy to see that $\mathcal{P} \cup \mathcal{Q} \models \top \sqsubseteq \bot$, but $\mathcal{P} \cup \mathcal{Q}_1 \not\models \top \sqsubseteq \bot$.

Given the relationships between our definition of module and conservative extensions, it is worth examining the computational complexity of the associated problems. The problem of deciding whether $\mathcal{Q}$ is an $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$ has been studied in [9], where it has been shown that deductive conservative extensions are decidable and 2NEXPTIME-complete for $\mathcal{ALCIQ}$ (roughly OWL-Lite) and undecidable for OWL DL. For model conservative extensions, it has been shown in [9] that the problem is highly undecidable (non-recursively enumerable), even for $\mathcal{ALC}$.

The decidability result from [9] for deductive conservative extensions, however, does not transfer to our problem since an ontology $\mathcal{Q}$ may well be an $\mathbf{S}$-deductive conservative extension of $\mathcal{Q}_1$, but still $\mathcal{Q}_1$ might not be an $\mathbf{S}$-module in $\mathcal{Q}$. In fact, we show that our problem is already undecidable for $\mathcal{ALC}$ ontologies w.r.t. languages that admit nominals.

THEOREM 1 (UNDECIDABILITY FOR ESSENTIAL AXIOMS). *Given a signature $\mathbf{S}$, an $\mathcal{ALC}$-ontology $\mathcal{Q}$ and an axiom $\alpha \in \mathcal{Q}$, it is undecidable whether α is $\mathbf{S}$-essential in $\mathcal{Q}$ w.r.t.* $\mathrm{L} = \mathcal{ALCO}$.

The proof is a variation of the construction for undecidability of deciding deductive conservative extensions in $\mathcal{ALCQIO}$ given [9], based on reduction to domino tiling problems. The proof is rather involved and we refer to the reader to [4] for more details.

COROLLARY 1. *There exists no algorithm for performing any of the tasks T1-T3 from* (3), *and* (4) *for $\mathcal{ALC}$.*

PROOF. Theorem 1 implies directly that there is no algorithm for task T3 from (4), because otherwise, one can check if an axiom α is $\mathbf{S}$-essential in $\mathcal{Q}$ by simply computing the set of all essential axioms by this algorithm for T3 and then checking if α is contained in this set. The remaining tasks from (3) are unsolvable since they are reducible to T3 by Proposition 1. $\square$

COROLLARY 2. *Given a signature $\mathbf{S}$, an $\mathcal{ALC}$-ontology $\mathcal{Q}$ and an ontology $\mathcal{Q}_1 \subseteq \mathcal{Q}$, it is undecidable whether $\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$ w.r.t.* $\mathrm{L} = \mathcal{ALCO}$.

PROOF. The procedure for deciding if $\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$ can be used for solving task T1, which is not possible by Corollary 1. Indeed, by enumerating the subsets of $\mathcal{Q}$ and checking if they are modules, one can compute all subsets $\mathcal{M}$ of $\mathcal{Q}$ that are $\mathbf{S}$-modules in $\mathcal{Q}$. The set of all minimal modules in $\mathcal{Q}$ can be then computed from $\mathcal{M}$ by filtering out those sets in $\mathcal{M}$ that are proper subsets of some other sets in $\mathcal{M}$. $\square$

Corollary 2 has a strong impact on the problem of knowledge reuse and forces us to revisit the original problem we aim at solving. As the problem of extracting minimal modules cannot be computationally solved, for rather inexpressive fragments of OWL DL, in none of the forms T1-T3 we propose to relax some of the requirements in these tasks. We cannot give up the requirements that extracted fragments should be modules since, in this case, we have no guarantee for the correctness of the result. We can sacrifice, however, the minimality requirements for the computed modules and consider the following weakened version of the task T2:

T2w. compute *some* small enough $\mathbf{S}$-module in $\mathcal{Q}$ (5)

Although it is always possible to extract a module (one can simply return $\mathcal{Q}$ which is always an $\mathbf{S}$-module in $\mathcal{Q}$), it still makes sense to develop, compare, and practically apply procedures that compute reasonably small modules. In the rest of the paper we describe two procedures of this form, based on the notions of locality, which we first introduced in [3]. The modules we obtain might be larger than the minimal modules and therefore we need to show that, in practice, they are still reasonably small.

4. MODULES BASED ON LOCALITY

In this section, we formulate the notion of locality, first introduced in [3] which will constitute the basis of our algorithm for extracting modules.

4.1 Locality

As a consequence of Proposition 2, model conservative extensions can be used as a sufficient condition for the notion of module. It is not possible, however, to design a procedure that extracts modules based on it since the problem of deciding model conservative extensions is highly undecidable [9]. The idea underlying this notion, however, can be used to establish sufficient conditions for the notion of module which are decidable and can be used in practice.

Consider Example 1, where we show that the set $\mathcal{Q}$ of axioms M1-M5 in Figure 1 is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}_1 = \{\mathrm{M1}, \ldots, \mathrm{M4}\}$, for $\mathbf{S} = \{\mathsf{Cystic\_Fibrosis}, \mathsf{Genetic\_Disorder}\}$. In this example, the model conservative extension property was shown by finding a particular extension of any interpretation for $\mathsf{Sig}(\mathcal{Q}_1)$ to a model of $\mathcal{Q}$ in which all concept and atomic roles not in $\mathsf{Sig}(\mathcal{Q}_1)$ were interpreted as the empty set. One could consider the cases where conservative extensions (and hence modules) can be determined in this manner. This intuition can be conveniently formulated using the notion of locality:

Definition 4 [Locality [3]]. Let $\mathbf{S}$ be a signature. We say that an *axiom α is local w.r.t.* $\mathbf{S}$ if every trivial expansion of any $\mathbf{S}$-interpretation to $\mathbf{S} \cup \mathsf{Sig}(\alpha)$ is a model of α. We denote by $\mathsf{local}(\mathbf{S})$ the set of all axioms that are local w.r.t. $\mathbf{S}$. An ontology $\mathcal{O}$ is local w.r.t. $\mathbf{S}$ if $\mathcal{O} \subseteq \mathsf{local}(\mathbf{S})$.

Intuitively, an ontology $\mathcal{O}$ is *local* w.r.t. a signature $\mathbf{S}$ if we can take *any* interpretation for the symbols in $\mathbf{S}$ and extend it to a *model* of $\mathcal{O}$ that interprets the additional symbols as the empty set.

Example 3. Consider axiom M5 from Figure 1. This axiom is local w.r.t. $\mathbf{S} = \{\mathsf{Cystic\_Fibrosis}, \mathsf{Genetic\_Disorder}\}$. Indeed, as shown in Example 1, for every trivial expansion $\mathcal{I}$ of an $\mathbf{S}$-interpretation to $\mathbf{S} \cup \mathsf{Sig}(\alpha)$, the atomic concept $\mathsf{DEFBI\_Gene}$ is interpreted with the empty set, and so, $\mathcal{I}$ satisfies M5.

On the other hand, M5 is not local w.r.t. $\mathbf{S} = \{\mathsf{DEFBI\_Gene}\}$. Indeed, take any $\mathbf{S}$-interpretation $\mathcal{I}_1$ in which $\mathsf{DEFBI\_Gene}$ is interpreted as a non-empty set. Then, for every trivial expansion $\mathcal{I}$ of $\mathcal{I}_1$, the concept on the left-hand-side of M5 is always interpreted by a non-empty set, whereas the concept on the right-hand-side is always interpreted by the empty set. So $\mathcal{I}$ does not satisfy α.

Locality can be used to formulate a sufficient condition for an ontology to be a model conservative extension of another ontology:

PROPOSITION 3 (LOCALITY $\Rightarrow$ MODEL CONSERVATIVITY). *Let $\mathcal{O}_1$, $\mathcal{O}_2$ be two ontologies and $\mathbf{S}$ a signature such that $\mathcal{O}_2$ is local w.r.t.* $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$. *Then $\mathcal{O}_1 \cup \mathcal{O}_2$ is an $\mathbf{S}$-model conservative extension of $\mathcal{O}_1$.*

PROOF. Let $\mathcal{I}_1$ be a model of $\mathcal{O}_1$. We show that there exists a model $\mathcal{I}$ of $\mathcal{O}_1 \cup \mathcal{O}_2$ such that $\mathcal{I}|_{\mathbf{S}} = \mathcal{I}_1|_{\mathbf{S}}$.

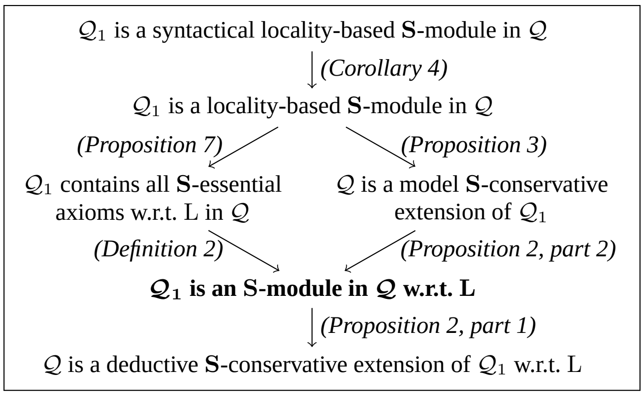

$\mathcal{Q}_1$ is a syntactical locality-based $\mathbf{S}$-module in $\mathcal{Q}$
(Corollary 4)
$\mathcal{Q}_1$ is a locality-based $\mathbf{S}$-module in $\mathcal{Q}$
(Proposition 7)
(Proposition 3)
$\mathcal{Q}_1$ contains all $\mathbf{S}$-essential axioms w.r.t. L in $\mathcal{Q}$
$\mathcal{Q}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$
(Definition 2)
(Proposition 2, part 2)
$\mathcal{Q}_1$ is an $\mathbf{S}$-module in $\mathcal{Q}$ w.r.t. L
(Proposition 2, part 1)
$\mathcal{Q}$ is a deductive $\mathbf{S}$-conservative extension of $\mathcal{Q}_1$ w.r.t. L

Figure 2: Summary for the main theoretical results of the paper

Let $\mathcal{I}$ be a trivial expansion of $\mathcal{I}_1|_{\mathbf{S}\cup\mathsf{Sig}(\mathcal{O}_1)}$ to $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1) \cup \mathsf{Sig}(\mathcal{O}_2)$, thus, in particular, $\mathcal{I}|_{\mathbf{S}\cup\mathsf{Sig}(\mathcal{O}_1)} = \mathcal{I}_1|_{\mathbf{S}\cup\mathsf{Sig}(\mathcal{O}_1)}$. We need to show that $\mathcal{I}$ is a model of $\mathcal{O}_1 \cup \mathcal{O}_2$. Since $\mathcal{O}_2$ is local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$, by Definition 4, $\mathcal{I}$ is a model of $\mathcal{O}_2$. Moreover, since $\mathcal{I}|_{\mathsf{Sig}(\mathcal{O}_1)} = \mathcal{I}_1|_{\mathsf{Sig}(\mathcal{O}_1)}$ and $\mathcal{I}_1 \models \mathcal{O}_1$, we have $\mathcal{I} \models \mathcal{O}_1$. Hence, $\mathcal{I} \models \mathcal{O}_1 \cup \mathcal{O}_2$ what was required to show. $\square$

Using Proposition 3 and Property 2 of Proposition 2 we obtain:

COROLLARY 3. *Let $\mathcal{O}_1, \mathcal{O}_2$ and $\mathbf{S}$ be as given in Proposition 3. Then $\mathcal{O}_1$ is an $\mathbf{S}$-module in $\mathcal{O}_1 \cup \mathcal{O}_2$.*

Next, we introduce our first restricted class of modules:

Definition 5 [Modules based on Locality Condition]. Given an ontology $\mathcal{Q}$ and a signature $\mathbf{S}$, we say that $\mathcal{Q}_1 \subseteq \mathcal{Q}$ *is a locality-based* $\mathbf{S}$*-module in* $\mathcal{Q}$ if $\mathcal{Q} \setminus \mathcal{Q}_1$ is local w.r.t $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$.

Example 4 [Example 3, continued]. We have seen in Example 3 that axiom M5 is local w.r.t. every $\mathbf{S}$ that does not contain the atomic concept DEFBI_Gene. In particular, for $\mathcal{Q}_1$ consisting of axioms M1-M4 from Figure 1, M5 is local w.r.t. $\mathsf{Sig}(\mathcal{Q}_1)$. Hence, according to Definition 5, $\mathcal{Q}_1$ is a locality-based $\mathbf{S}$-module in $\mathcal{Q} = \{\text{M1}, \ldots, \text{M5}\}$ for every $\mathbf{S} \subseteq \mathsf{Sig}(\mathcal{Q}_1)$.

For the reference and for the convenience of the reader, we illustrate in Figure 2 the relationships between the key theoretical results of this paper.

4.2 Computing Locality-Based Modules

As demonstrated in Example 3, for testing locality of an axiom α w.r.t. $\mathbf{S}$, it is sufficient to interpret every atomic concept and atomic role not in $\mathbf{S}$ with the empty set and then check if α is satisfied for all interpretations of the remaining symbols. This observation suggests that locality can be tested by first simplifying the ontology by eliminating atomic roles and concepts that are not in $\mathbf{S}$, and then checking if the resulting axioms are satisfied in every interpretation for the remaining symbols. This idea is formalized as follows:

PROPOSITION 4 (TESTING LOCALITY). *Let $\mathcal{O}$ be a $\mathcal{SHOIQ}$ ontology and $\mathbf{S}$ a signature. Let $\mathcal{O}_\mathbf{S}$ be obtained from $\mathcal{O}$ by applying the transformations below, where every A is an atomic concept, every r is an atomic role with $A, r \notin \mathbf{S}$, and every R is a role r or r^- with $r \notin \mathbf{S}$: (T1) replace all concepts of form A, $\exists R.C$ or $(\geqslant n\, R.C)$ with $\bot$; (T2) remove every transitivity axiom $\mathsf{Trans}(r)$; (T3) replace every assertion $a : A$ and $r(a, b)$ with the contradiction axiom $\top \sqsubseteq \bot$.*

Then $\mathcal{O}$ is local w.r.t. $\mathbf{S}$ iff every axiom in $\mathcal{O}_\mathbf{S}$ is a tautology.

PROOF. It is easy to check that the transformation above preserves the satisfaction of axioms under every trivial expansion $\mathcal{I}$ of every $\mathbf{S}$-interpretation to $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O})$. Hence, the resulting ontology $\mathcal{O}_\mathbf{S}$ is local w.r.t. $\mathbf{S}$ iff the original ontology $\mathcal{O}$ was local w.r.t. $\mathbf{S}$. Moreover, it is easy to see that there are no atomic concepts and atomic roles outside $\mathbf{S}$ left in $\mathcal{O}_\mathbf{S}$ after the transformation. Hence, every axiom α from $\mathcal{O}_\mathbf{S}$ is a tautology iff $\mathcal{Q}$ is local w.r.t. $\mathbf{S}$. $\square$

Note that according to Definition 4, assertions $a : A$ and $r(a, b)$ can never be local since, even when A and r are interpreted with the empty set, these axioms do not become tautologies. Hence, assertions must be included in every locality-based module, which is reflected by the step T3 of the transformation in Proposition 4.

An important conclusion of Proposition 4 is that one can use the standard capabilities of available DL-reasoners[2] for testing locality since these reasoners can test for DL-tautologies. Checking for tautologies in description logics is, theoretically, a difficult problem (e.g. for DL $\mathcal{SHOIQ}$ is NEXPTIME-complete). There are, however, several reasons to believe that the locality test would perform well in practice. First, and most importantly, the size of the axioms in an ontology is usually small compared to the size of the ontology. Second, DL reasoners are highly optimized for standard reasoning tasks and behave well for most realistic ontologies.

In case this is too costly, it is possible to formulate a tractable approximation to the locality conditions for $\mathcal{SHOIQ}$:

Definition 6 [Syntactic Locality for $\mathcal{SHOIQ}$]. Let $\mathbf{S}$ be a signature. The following grammar recursively defines two sets of concepts $\mathcal{C}^\bot_\mathbf{S}$ and $\mathcal{C}^\top_\mathbf{S}$ for a signature $\mathbf{S}$:

$$\begin{aligned}\mathcal{C}^\bot_\mathbf{S} &::= A^\bot \mid (\neg C^\top) \mid (C \sqcap C^\bot) \mid (\exists R^\bot.C)\\ &\quad\mid (\exists R.C^\bot) \mid (\geqslant n\, R^\bot.C) \mid (\geqslant n\, R.C^\bot)\,.\\ \mathcal{C}^\top_\mathbf{S} &::= (\neg C^\bot) \mid (C^\top_1 \sqcap C^\top_2)\,.\end{aligned}$$

where $A^\bot \notin \mathbf{S}$ is a atomic concept, R is a role, and C is a concept, $C^\bot \in \mathcal{C}^\bot_\mathbf{S}$, $C^\top_{(i)} \in \mathcal{C}^\top_\mathbf{S}$, $i = 1, 2$, and $R^\bot \notin \mathsf{Rol}(\mathbf{S})$ is a role.

An axiom α is *syntactically local w.r.t.* $\mathbf{S}$ if it is of one of the following forms: (1) $R^\bot \sqsubseteq R$, or (2) $\mathsf{Trans}(R^\bot)$, or (3) $C^\bot \sqsubseteq C$ or (4) $C \sqsubseteq C^\top$. We denote by $\mathsf{s\_local}(\mathbf{S})$ the set of all $\mathcal{SHOIQ}$-axioms that are syntactically local w.r.t. $\mathbf{S}$. A $\mathcal{SHOIQ}$-ontology $\mathcal{O}$ is *syntactically local w.r.t.* $\mathbf{S}$ if $\mathcal{O} \subseteq \mathsf{s\_local}(\mathbf{S})$.

Intuitively, every concept in $\mathcal{C}^\bot_\mathbf{S}$ becomes equivalent to $\bot$ if we replace every symbol $A^\bot$ or $R^\bot$ not in $\mathbf{S}$ with the bottom concept $\bot$ and the empty role respectively, which are both interpreted as the empty set under every interpretation. Similarly, the concepts from $\mathcal{C}^\top_\mathbf{S}$ are equivalent to $\top$ under this replacement. Syntactically local axioms become tautologies after these replacements.

For example, the axiom M2 from Figure 1 is local w.r.t. $\mathbf{S} = \{\mathsf{Fibrosis}, \mathsf{has\_Origin}\}$: if we replace the remaining symbols in this axiom with $\bot$, we obtain a tautology $\bot \equiv \bot$:

$$\overbrace{\mathsf{Genetic\_Fibrosis}}^{\bot} \equiv \underline{\mathsf{Fibrosis}} \sqcap \underbrace{\exists\underline{\mathsf{has\_Origin}}.\overbrace{\mathsf{Genetic\_Origin}}^{\bot}}_{\bot}$$

Syntactic locality is an approximation for (semantic) locality:

PROPOSITION 5. *Let $\mathbf{S}$ be a signature. Then* $\mathsf{s\_local}(\mathbf{S}) \subseteq \mathsf{local}(\mathbf{S})$.

PROOF. Let α be an axiom that is syntactically local w.r.t. $\mathbf{S}$ and let $\mathcal{I} = (\Delta, \cdot^\mathcal{I})$ be a trivial expansion of some $\mathbf{S}$-interpretation to

[2]See http://www.cs.man.ac.uk/~sattler/reasoners.html for a list of currently available reasoners.

Algorithm 1 extract_module($\mathcal{Q}, \mathbf{S}$)

```
Input:
    $\mathcal{Q}$: ontology
    $\mathbf{S}$: signature
Output:
    $\mathcal{Q}_1$: a locality-based $\mathbf{S}$-module in $\mathcal{Q}$

 1: $\mathcal{Q}_1 \leftarrow \emptyset \quad \mathcal{Q}_2 \leftarrow \mathcal{Q}$
 2: while not empty($\mathcal{Q}_2$) do
 3:     $\alpha \leftarrow$ select_axiom($\mathcal{Q}_2$)
 4:     if locality_test( $\alpha$, $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$ ) then
 5:         $\mathcal{Q}_2 \leftarrow \mathcal{Q}_2 \setminus \{\alpha\}$                ▷ $\alpha$ is processed
 6:     else
 7:         $\mathcal{Q}_1 \leftarrow \mathcal{Q}_1 \cup \{\alpha\}$                ▷ move $\alpha$ into $\mathcal{Q}_1$
 8:         $\mathcal{Q}_2 \leftarrow \mathcal{Q} \setminus \mathcal{Q}_1$   ▷ reset $\mathcal{Q}_2$ to the complement of $\mathcal{Q}_1$
 9:     end if
10: end while
11: return $\mathcal{Q}_1$
```

$\mathbf{S} \cup \mathsf{Sig}(\alpha)$. We have to demonstrate that $\mathcal{I}$ is a model of α. By induction over the definitions of $\mathcal{C}^{\perp}_{\mathbf{S}}$ and $\mathcal{C}^{\top}_{\mathbf{S}}$ from Definition 6, it is easy to show that: (i) every role $R \notin \mathsf{Rol}(\mathbf{S})$ and every every concept from $\mathcal{C}^{\perp}_{\mathbf{S}}$ is interpreted in $\mathcal{I}$ by the empty set, and (ii) every concept from $\mathcal{C}^{\top}_{\mathbf{S}}$ is interpreted in $\mathcal{I}$ by Δ. By checking the possible cases for a syntactically local axiom α in Definition 5, it is easy to see that in every of these cases $\mathcal{I}$ is a model of α. □

The converse of Proposition 5 does not hold in general since there are semantically local axioms that are not syntactically local. For example, the axiom $\alpha = (A \sqsubseteq A \sqcup B)$ is a tautology and thus is local w.r.t. every $\mathbf{S}$. This axiom, however, is not syntactically local w.r.t. $\mathbf{S} = \{A, B\}$ since it involves symbols in $\mathbf{S}$ only. Another example, which is not a tautology, is the GCI $\alpha = (\exists R.\neg A \sqsubseteq \exists R.\neg B)$, which is syntactically non-local and semantically local w.r.t. $\mathbf{S} = \{R\}$ since $\exists R.\top \sqsubseteq \exists R.\top$ is a tautology. Thus, the limitation of syntactic locality is its inability to "compare" different occurrences of elements from the given signature $\mathbf{S}$.

We distinguish the notion of modules based on these two locality conditions as *semantic locality-based modules* and *syntactic locality-based modules*.

COROLLARY 4. *If $\mathcal{Q}_1$ is a syntactic locality-based $\mathbf{S}$-module in $\mathcal{Q}$, then $\mathcal{Q}_1$ is a semantic locality-based $\mathbf{S}$-module in $\mathcal{Q}$.*

Recall that, according to Definition 5, in order to construct a locality-based $\mathbf{S}$-module in an ontology $\mathcal{Q}$, it suffices to partition the ontology $\mathcal{Q}$ as $\mathcal{Q} = \mathcal{Q}_1 \cup \mathcal{Q}_2$ such that $\mathcal{Q}_2$ is local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$. Algorithm 1 outlines a simple procedure which performs this task. Assuming there is an effective locality test locality_test($\alpha, \mathbf{S}$) (either using a reasoner or the syntactical approximation) that returns true if the axiom α is local w.r.t. $\mathbf{S}$, the algorithm first initializes the partition to the trivial one: $\mathcal{Q}_1 = \emptyset$ and $\mathcal{Q}_2 = \mathcal{Q}$, and then repeatedly moves to $\mathcal{Q}_1$ those axioms from $\mathcal{Q}_2$ that are not local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$ until no such axioms are left in $\mathcal{Q}_2$.

In Table 1 we provide a trace of Algorithm 1 for the input $(\mathcal{Q}, \mathbf{S})$, where $\mathcal{Q}$ consists of the axioms M1-M5 from Figure 1 and $\mathbf{S} = \{\mathsf{Cystic\_Fibrosis}, \mathsf{Genetic\_Disorder}\}$. Each row of the table corresponds to an iteration of the while loop in Algorithm 1. The last column of the table provides the result of the locality test in line 4.

Two remarks are in order. First, for the example in Table 1, the syntactic locality condition was sufficient in all tests: all axioms that were semantically non-local were also syntactically non-local. Second, the result of the algorithm does not depend on the choice of

$\sharp$	$\mathcal{Q}_1$	$\mathcal{Q}_2$	New elements in $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$	α	loc.?
1	$\emptyset$	M1 − M5	Cystic_Fibrosis, Genetic_Disorder	M1	No
2	M1	M2 − M5	Fibrosis, located_In, Pancreas, has_Origin, Genetic_Origin	M2	No
3	M1, M2	M3 − M5	Genetic_Fibrosis	M3	No
4	M1 − M3	M4, M5	–	M4	No
5	M1 − M4	M5	–	M5	Yes
6	M1 − M4	–	–	–	

Table 1: A trace of Algorithm 1 for $\mathcal{Q} = \{\mathrm{M1}, \ldots, \mathrm{M5}\}$ and $\mathbf{S} = \{\mathsf{Cystic\_Fibrosis},\ \mathsf{Genetic\_Disorder}\}$

α in line 3. For example, in Table 1 at step 2 we might have selected axiom M3 instead of M2. The final result does not change.

PROPOSITION 6 (CORRECTNESS OF ALGORITHM 1). *For every input $\mathcal{Q}$ and $\mathbf{S}$, Algorithm 1 computes the smallest (syntactic) locality-based $\mathbf{S}$-module in $\mathcal{Q}$.*

PROOF. We have to show that (1) Algorithm 1 terminates for every input $\mathcal{T}$ and $\mathbf{S}$, and (2) the output extract_module($\mathbf{S}, \mathcal{Q}$) is a locality-based $\mathbf{S}$-module in $\mathcal{Q}$.

(1) Termination of the algorithm follows from the fact that in every iteration of the while loop either the size of $\mathcal{Q}_1$ decreases, or the size of $\mathcal{Q}_1$ remains the same but the size of $\mathcal{Q}_2$ decreases. Note that this means that Algorithm 1 terminates in quadratic time in the number of axioms in $\mathcal{Q}$, assuming constant time locality test.

(2) It is easy to observe that every axiom α that is neither in $\mathcal{Q}_1$ nor in $\mathcal{Q}_2$ is local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$, since the only way such an α can appear is at the line 3 of the algorithm, and α remains in $\mathcal{Q} \setminus (\mathcal{Q}_1 \cup \mathcal{Q}_2)$ only if $\mathbf{S} \cup \mathsf{Sig}(\mathcal{Q}_1)$ does not change. □

4.3 Properties of Locality-based Modules

In this section, we outline some interesting properties of locality-based modules which make it possible to use them for applications other than knowledge reuse.

Let $\mathcal{Q}^{loc}_{\mathbf{S}}$ be the minimal locality-based $\mathbf{S}$-module in $\mathcal{Q}$, which is unique by Proposition 6 and is the output of Algorithm 1 for $\mathcal{Q}$ and $\mathbf{S}$. The first property is a direct consequence of Proposition 6 (see [4] for details):

PROPOSITION 7. *$\mathcal{Q}^{loc}_{\mathbf{S}}$ contains all $\mathbf{S}$-essential axioms in $\mathcal{Q}$ w.r.t. every logic L with Tarski-style set-theoretic semantics.*

As shown in Table 1, the minimal locality-based $\mathbf{S}$-module extracted from $\mathcal{Q}$ contains all $\mathbf{S}$-essential axioms M1–M4. In our case, the module contains only essential axioms; in general, however, locality-based modules might contain non-essential axioms; otherwise, they would provide a solution for our task T3 in (4).

PROPOSITION 8. *Let $\mathcal{Q}$ be ontology, A and B atomic concepts and $\mathbf{S}_{(i)}$ a signature. Then:*

1. *$\mathbf{S}_1 \subseteq \mathbf{S}_2$ implies $\mathcal{Q}^{loc}_{\mathbf{S}_1} \subseteq \mathcal{Q}^{loc}_{\mathbf{S}_2}$ (monotonicity);*
2. *$\mathcal{Q} \models (A \sqsubseteq B)$ iff $\mathcal{Q}^{loc}_{\{A\}} \models (A \sqsubseteq B)$.*

Proposition 8 gives two interesting properties of locality-based modules (see [4] for a proof). The first one states that the such modules may only grow if the input signature is extended. The second one implies that the module for a single atomic concept A provides complete information about all the super-classes of A.

Ontology	Language	♯ Atomic	Prompt-Factor [10] (A1)		Modularization from [6] (A2)		Locality-based mod. (A3)	
		Concepts	Max. Size (%)	Avg. Size (%)	Max. Size (%)	Avg. Size (%)	Max. Size(%)	Avg. Size(%)
NCI	$\mathcal{EL}$	27772	24342 (87.6)	21045 (75.8)	15254 (55)	8565 (30.8)	226 (0.8)	22 (0.08)
SNOMED	$\mathcal{EL}$	255318	255318 (100)	255318 (100)	255318 (100)	255318 (100)	136 (0.5)	12.8 (0.05)
GO	$\mathcal{EL}$	22357	226 (1)	22 (0.1)	226 (1)	22 (0.1)	92 (0.4)	13 (0.05)
SUMO	$\mathcal{EL}$	869	869 (100)	869 (100)	869 (100)	869 (100)	18 (2)	8 (0.09)
GALEN-Small	$\mathcal{SHF}$	2749	2748 (100)	2748 (100)	2748 (100)	2748 (100)	297 (10)	47.7 (1.7)
GALEN-Full	$\mathcal{SHIF}$	24089	24089 (100)	24089 (100)	24089 (100)	24089 (100)	7379 (29.8)	865.5 (3.5)
SWEET	$\mathcal{SHOIF}$	1816	1750 (96.4)	1610 (88.7)	1512 (83.3)	935 (51.5)	34 (1.9)	1.7 (0.1)
DOLCE-Lite	$\mathcal{SHOIN}$	499	498 (100)	497.9 (100)	498 (100)	497.9 (100)	186 (37.3)	123.4 (24.6)

Table 2: Comparison of Different Modularization Algorithms

This property can be used for optimizing classification: in order to *classify an ontology* $\mathcal{Q}$, i.e. to compute all *subsumption relation* $A \sqsubseteq B$ between pairs A, B of atomic concepts in $\mathcal{Q}$, it is sufficient to (1) extract all modules $\mathcal{Q}^{loc}_{\{A\}}$ of $\mathcal{Q}$ for each atomic concept A (2) classify each of these modules *independently* (possibly *in parallel*), and (3) merge the results of the individual classifications. By Property 2, if the subsumption $A \sqsubseteq B$ is implied by the ontology $\mathcal{Q}$ then it is implied by the module $\mathcal{Q}^{loc}_{\{A\}}$ and, hence, it will be obtained in step (2).

5. RELATED WORK

The problem of extracting modular fragments of ontologies has recently been addressed in [15], [10] and [14].

In [15], the authors have proposed an algorithm for partitioning the concepts in an ontology. The intended application is to facilitate the visualization of and navigation through the ontology. The algorithm uses a set of heuristics for measuring the degree of dependency between the concepts in the ontology and outputs a graphical representation of these dependencies. The algorithm is intended as a visualization technique, and does not establish a correspondence between the nodes of the graph and sets of axioms in the ontology.

The algorithms in [10] and [14], which we have briefly outlined in Section 3, use structural traversal to extract modules of ontologies for a given signature. None of these approaches provides a characterization of the logical properties of the extracted modules, nor do they establish a notion of correctness of the modularization.

In [6], the authors propose a definition of a module and an algorithm for extracting modules based on that definition. The notion of a module in an ontology $\mathcal{Q}$ for a signature $\mathbf{S}$ is also based on conservative extensions: if $\mathcal{Q}_1 \subseteq \mathcal{Q}$ is an $\mathbf{S}$-module in $\mathcal{Q}$ as in [6], then it can be shown that $\mathcal{Q}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{Q}$. The definition in [6], however, makes use of additional requirements which lead, in many cases, to the extraction of modules which are larger than one may wish. The reason is that, for every atomic concept $A \in \mathbf{S}$, the module $\mathcal{Q}_1$ for A in $\mathcal{Q}$ must be a module for all its sub-classes and super-classes.

It is worth pointing out that, given $\mathcal{Q}$ and $\mathbf{S}$, the fragment obtained using the algorithm in [6] is an $\mathbf{S}$-module according to Definition 1. This is not the case, however, for the fragment extracted using [14], as we have illustrated in Section 3.

6. IMPLEMENTATION AND EVALUATION

Given an input ontology and an input signature, locality-based modules are not the only possible modules we can obtain. It remains to be shown that the locality-based modules obtained in realistic ontologies are *small enough* to be useful in practice.

For evaluation and comparison, we have implemented the following algorithms using Manchester's OWL API:[3]

- The PROMPT-FACTOR algorithm, as described in [10] (A1).
- The algorithm for extracting modules described in [6] (A2).
- Our algorithm for extracting modules (Algorithm 1), based on syntactic locality (A3).

As a test suite, we have collected a set of well-known ontologies available on the Web, which can be divided into two groups:

Simple. In this group, we have included the National Cancer Institute (NCI), Ontology[4] the SUMO Upper Ontology,[5] the Gene Ontology,[6] and the SNOMED Ontology[7]. These ontologies use a simple ontology language and are of a simple structure; in particular, they do not contain GCIs, but only definitions.

Complex. This group contains the well-known GALEN[8] ontology, the DOLCE upper ontology[9] and NASA's Semantic Web for Earth and Environmental Terminology (SWEET)[10]. These ontologies are complex since they use many constructors from OWL DL and/or include a significant number of GCIs. In the case of GALEN, we have also considered a fragment GALEN-Small that has commonly been used as a benchmark for OWL reasoners. This fragment is almost 10 times smaller than the original GALEN-Full ontology, yet similar in structure.

For each of these ontologies, and for each atomic concept in their signature, we have extracted the corresponding modules using algorithms A1-A3 and measured their size. We use modules for single atomic concepts to get an idea of the typical size of locality-based modules compared to the size of the whole ontology. Also, modules for atomic concepts are especially interesting for modular classification of ontologies, as discussed in Section 4.3.

The results we have obtained are summarized in Table 2. The table provides the size of the largest module and the average size

[3]http://sourceforge.net/projects/owlapi
[4]http://www.mindswap.org/2003/CancerOntology/nciOncology.owl
[5]http://ontology.teknowledge.com/
[6]http://www.geneontology.org
[7]http://www.snomed.org
[8]http://www.openclinical.org/prj_galen.html
[9]http://www.loa-cnr.it/DOLCE.html
[10]http://sweet.jpl.nasa.gov/ontology/

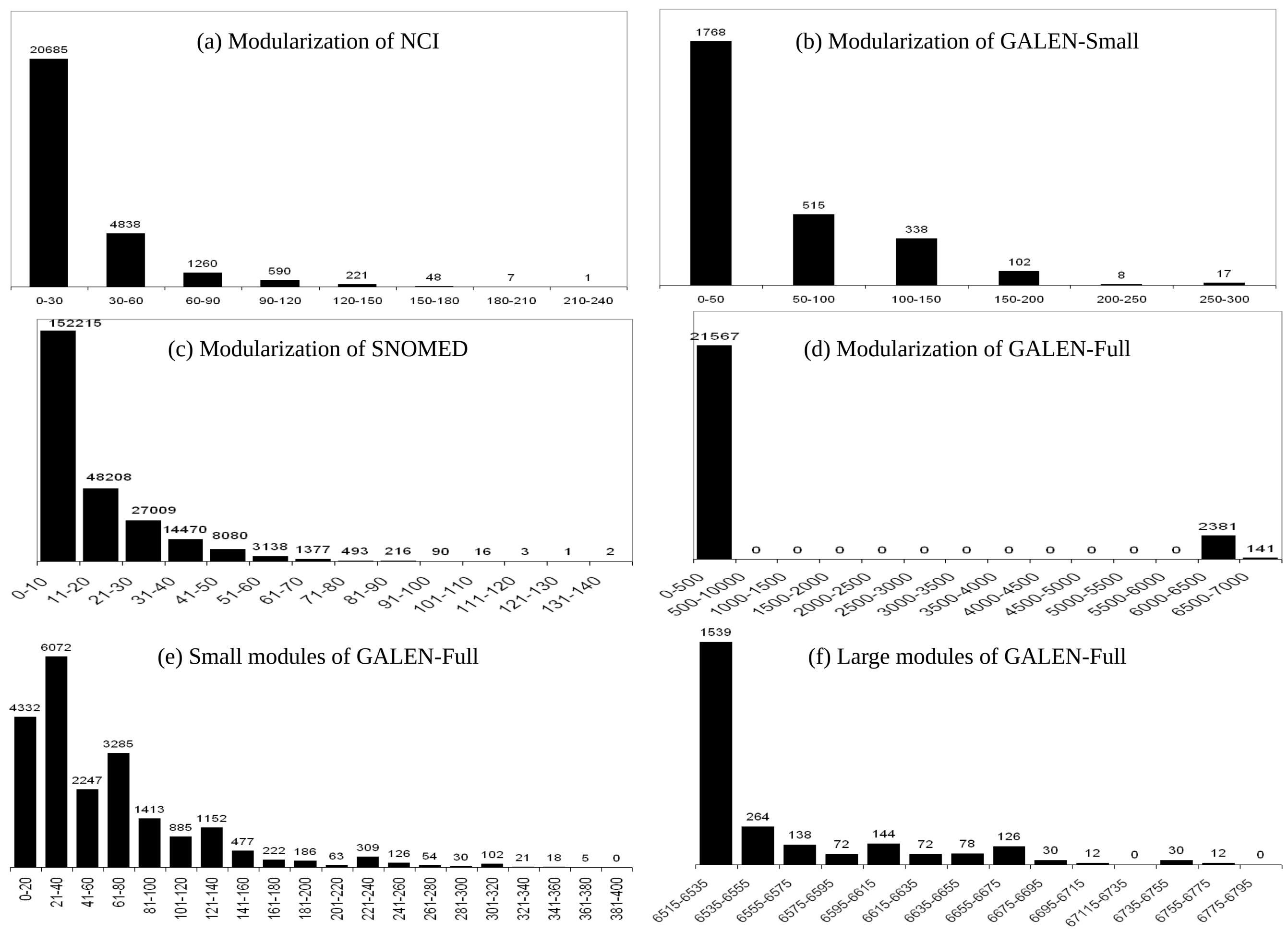


Figure 3: Distribution for the sizes of syntactic locality-based modules for atomic concepts: the X-Axis gives the number of concepts in the modules and the Y-Axis the number of modules for each size range.

of the modules obtained using each of these algorithms. In the table, we can clearly see that locality-based modules are significantly smaller than the ones obtained using the other methods; in particular, in the case of SUMO, DOLCE, GALEN and SNOMED, the algorithms A1 and A2 retrieve the whole ontology as the module for each atomic concept. In contrast, the modules we obtain using our algorithm are significantly smaller than the size of the input ontology. Our modules turned out to be not only smaller, but are strict subsets of the respective modules computed using A1 and A2.

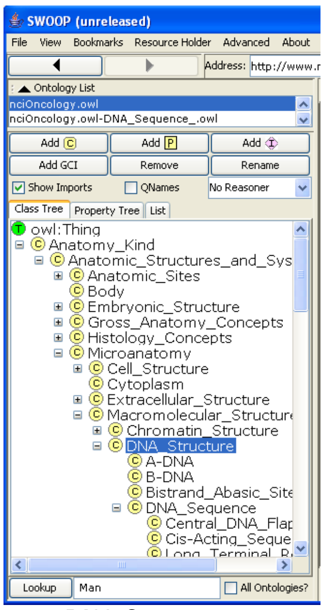
For NCI, SNOMED, GO and SUMO, we have obtained very small locality-based modules. This can be explained by the fact that these ontologies, even if large, are simple in structure and logical expressivity. For example, in SNOMED, the largest locality-based module obtained is approximately 1/10000 of the size of the ontology, and the average size of the modules is 1/10 of the size of the largest module. In fact, most of the modules we have obtained for these ontologies contain less than 40 atomic concepts.

For GALEN, SWEET and DOLCE, we have obtained larger locality-based modules. Indeed, the largest module in GALEN-Small is 1/10 of the size of the ontology, as opposed to 1/10000 in the case of SNOMED. For DOLCE, the modules are even bigger –1/3 of the size of the ontology– which indicates that the dependencies between the different concepts in the ontology are very strong and complicated. The SWEET ontology is an exception: even though the ontology uses most of the constructors available in OWL, the ontology is heavily underspecified, which yields small modules.

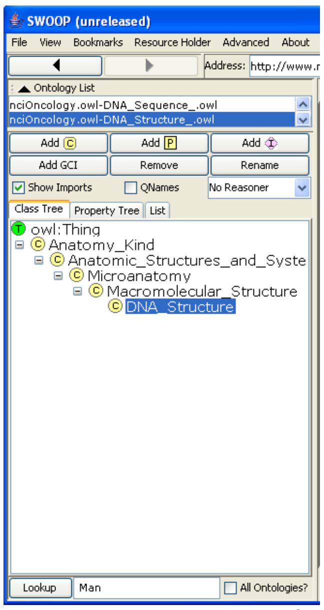
In Figure 3, we have presented a more detailed analysis of the modules for NCI, SNOMED, GALEN-Small and GALEN-Full. Here, the X-axis represents the size ranges of the obtained modules and the Y-axis the number of modules whose size is within the given range. The plots thus give an idea of the distribution for the sizes of the different modules.

For SNOMED, NCI and GALEN-Small, we can observe that the size of the modules follows a smooth distribution. In contrast, for GALEN-Full, we have obtained a large number of small modules and a significant number of very big ones, but no medium-sized modules in-between. This abrupt distribution indicates the presence of a big cycle of dependencies in the ontology, which involves all the concepts with large modules. The presence of this cycle can be spotted more clearly in Figure 3f; the figure shows that there is a large number of modules of size in between 6515 and 6535 concepts. This cycle does not occur in the simplified version of GALEN and thus we have the smooth distribution for that case. In contrast, in Figure 3e we can see that the distribution for the "small" modules in GALEN-Full is smooth and much more similar to the one for the simplified version of GALEN.

In order to explore the use of our results for ontology design and



(a) DNA_Structure in NCI



(b) Locality-based Module for DNA_Structure in NCI

Figure 4: The Module Extraction Functionality in Swoop

analysis, we have integrated our algorithm for extracting modules in the ontology editor SWOOP [8]. The user interface of SWOOP allows for the selection of an input signature and the retrieval of the corresponding module.

As an illustration, consider the locality-based module, shown in Figure 4b, for the atomic concept DNA_Structure in the NCI ontology, as obtained in SWOOP. Recall that Proposition 8 provides the *scope* of a locality-based module: given the atomic concept $A \in \mathsf{Sig}(\mathcal{O})$, the module $\mathcal{O}^{loc}_{\{A\}}$ contains all necessary axioms for, at least, all the (entailed) super-concepts of A in $\mathcal{O}$. Thus, one can intuitively understand the locality-based module for a atomic concept A as the "upper ontology" above A. Indeed, the figure shows that the locality-based module only contains the classes in the "path" from Man to the top level concept Organism_Kind. In this case, the atomic concepts in this particular path of the concept hierarchy are the *only* symbols that are finally brought into the module; this suggests that the knowledge contained in NCI about the particular concept DNA_Structure is very shallow in the sense that NCI only "knows" that a DNA_Structure is a Macromolecular_Structure, which itself is an anatomic structure.

7. CONCLUSION

In this paper, we have proposed a definition of a module for a given vocabulary within an ontology that we want to reuse. Based on this definition, we have formulated three reasoning problems concerning the extraction of modules and shown that all of them are algorithmically unsolvable, even for very inexpressive sub-languages of OWL DL. We have proposed the notion of locality as an interesting approximation that still guarantees that modules will completely capture the meaning of the given vocabulary. Our empirical results show that the modules we extract by exploiting locality are small enough to be useful in applications.

8. ACKNOWLEDGMENTS

This work is supported by the EU Project TONES (Thinking ONtologiES) ref: IST-007603 and by the EPSRC Project REOL (Reasoning in Expressive Ontology Languages) ref:EP/C537211/1. The authors would like to thank Boris Motik for his suggestions and assistance concerning this work.

9. REFERENCES

[1] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. IJCAI-2005*, pages 364–370.

[2] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[3] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. A logical framework for modularity of ontologies. In *Proc. IJCAI-2007*, pages 298–304.

[4] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Extracting modules from ontologies: Theory and practice. Technical report, University of Manchester, 2007. Available at http://www.cs.man.ac.uk/~bcg/Publications.html.

[5] B. Cuenca Grau, I. Horrocks, O. Kutz, and U. Sattler. Will my Ontologies Fit Together? In *Proc. DL-2006*, 2006.

[6] B. Cuenca Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Modularity and Web Ontologies. In *Proc. KR-2006*, pages 198–209.

[7] S. Ghilardi, C. Lutz, and F. Wolter. Did I Damage my Ontology? A Case for Conservative Extensions in Description Logics. In *Proc. KR-2006*, pages 187–197.

[8] A. Kalyanpur, B. Parsia, E.Sirin, B. Cuenca Grau, and J. Hendler. SWOOP: A web editing browser. *Elsevier's Journal Of Web Semantics*, 4(2):144–153.

[9] C. Lutz, D. Walther, and F. Wolter. Conservative extensions in expressive description logics. In *Proc. of IJCAI-2007*, pages 453–459.

[10] N. Noy and M. Musen. The PROMPT suite: Interactive tools for ontology mapping and merging. *Int. Journal of Human-Computer Studies*, 6(59), 2003.

[11] P. Patel-Schneider, P. Hayes, and I. Horrocks. Web ontology language OWL Abstract Syntax and Semantics. *W3C Recommendation*, 2004.

[12] A. Rector and J. Rogers. Ontological issues in using a description logic to represent medical concepts: Experience from GALEN. In *Proc. of IMIA WG6 Workshop*, 1999.

[13] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artif. Intell.*, 48(1):1–26, 1991.

[14] J. Seidenberg and A. Rector. Web ontology segmentation: Analysis, classification and use. In *Proc. WWW-2006*, 2006.

[15] H. Stuckenschmidt and M. Klein. Structure-based partitioning of large class hierarchies. In *Proc. ISWC-2004*, pages 289–303.