Module Extraction and Incremental Classification: A Pragmatic Approach for $\mathcal{EL}^+$ Ontologies*

Boontawee Suntisrivaraporn

Theoretical Computer Science, TU Dresden, Germany
`meng@tcs.inf.tu-dresden.de`

Abstract. The description logic $\mathcal{EL}^+$ has recently proved practically useful in the life science domain with presence of several large-scale biomedical ontologies such as SNOMED CT. To deal with ontologies of this scale, standard reasoning of classification is essential but not sufficient. The ability to extract relevant fragments from a large ontology and to incrementally classify it has become more crucial to support ontology design, maintenance and re-use. In this paper, we propose a pragmatic approach to module extraction and incremental classification for $\mathcal{EL}^+$ ontologies and report on empirical evaluations of our algorithms which have been implemented as an extension of the CEL reasoner.

1 Introduction

In the past few years, the $\mathcal{EL}$ family of description logics (DLs) has received an increasing interest and been intensively studied (see, e.g., [1,2,3,8]). The attractiveness of the $\mathcal{EL}$ family is twofold: on the one hand, it is computationally tractable, i.e., subsumption is decidable in polytime; on the other hand, it is sufficiently expressive to formulate many life science ontologies. Examples include the Gene Ontology, the thesaurus of the US National Cancer Institute (NCI), the Systematized Nomenclature of Medicine, Clinical Terms (SNOMED CT), and large part (more than 95%) of the Galen Medical Knowledge Base (GALEN). We lay emphasis on SNOMED CT which comprises about four hundred thousand axioms and is now a standardized clinical terminology adopted by health care sectors in several countries [13].

Being a standard ontology, SNOMED has been designed to comprehensively cover a whole range of concepts in the medical and clinical domains. For this reason, it is often the case that only a small part is actually needed in a specific application. The ability to automate extraction of meaningful sub-ontologies that cover all relevant information is becoming important to support re-use of typically comprehensive standardized ontologies. Several techniques for syntactic module extraction have been proposed [9,11,6], since semantic extraction is highly complex [6]. Though (deductive) conservative extension could be used as a sufficient condition for extracting a module, it is unfortunately too expensive

* Supported by DFG-Project under grant BA 1122/11-1 and EU-Project TONES.

(ExpTime-complete already in $\mathcal{EL}$ with GCIs [8]). In Section 3 of the present paper, we define a new kind of module, called *reachability-based modules*, which is motivated by a once-employed optimization technique in the CEL system and which can be extracted in linear time. Also, we propose an algorithm for extracting modules of this kind and show some interesting properties.

Despite being classifiable by modern DL reasoners, design and maintenance of large-scale ontologies like Snomed ct requires additional reasoning support. This is due to the fact that an ontology under development evolves continuously, and the developer often has to undergo the long process of *full classification* after addition of a few new axioms. Though classification of Snomed requires less than half an hour (see [2] or Table 1 in the present paper), the ontology developer is not likely willing to wait that long for a single change. In the worst case, she may end up not using automated reasoning support which could have helped identify potential modeling errors at an early stage. In Section 4, we propose a *goal-directed* variant of the $\mathcal{EL}^+$ classification algorithm developed in [3] which can be used for testing subsumption queries prior to full classification. Section 5 presents an extension of the algorithm in [3] to cater for two ontologies: the permanent ontology $\mathcal{O}_p$ which has been carefully modeled, and axioms of which are not supposed to be modified; and, the temporary ontology $\mathcal{O}_t$ that contains new axioms currently being authored. The extended algorithm reuses information from the previous classification of $\mathcal{O}_p$ and thus dispense with the need of the full classification of $\mathcal{O}_p \cup \mathcal{O}_t$. We call reasoning in this setting *restricted incremental classification.*

All algorithms proposed in this paper have been implemented in the CEL reasoner [2] and various experiments on realistic ontologies have been performed. The experiments and their promising results are discussed in Section 6.

For interested readers, proofs omitted from the present paper can be found in the associated technical report [12].

2 Preliminaries

The present paper focuses on the sub-Boolean DL $\mathcal{EL}^+$ [3], which is the underlying logical formalism of the CEL reasoner [2]. Similar to other DLs, an $\mathcal{EL}^+$ signature is the disjoint union $\mathsf{S} = \mathsf{CN} \cup \mathsf{RN}$ of the sets of concept names and role names. $\mathcal{EL}^+$ *concept descriptions (or complex concepts)* can be defined inductively as follows: each concept name $A \in \mathsf{CN}$ and the top concept $\top$ are $\mathcal{EL}^+$ concept descriptions; and, if C, D are $\mathcal{EL}^+$ concept descriptions and $r \in \mathsf{RN}$ is a role name, then concept conjunction $C \sqcap D$ and existential restriction $\exists r.C$ are $\mathcal{EL}^+$ concept descriptions. An $\mathcal{EL}^+$ *ontology* $\mathcal{O}$ is a finite set of *general concept inclusion (GCI)* axioms $C \sqsubseteq D$ and *role inclusion (RI)* axioms $r_1 \circ \cdots \circ r_n \sqsubseteq s$ with C, D $\mathcal{EL}^+$ concept descriptions and r_i, s role names. Concept equivalences and (primitive) concept definitions are expressible using GCIs, whereas RIs can be used to express various role axioms, such as reflexivity ($\epsilon \sqsubseteq r$), transitivity ($r \circ r \sqsubseteq r$), right-identity ($r \circ s \sqsubseteq r$), and role hierarchy ($r \sqsubseteq s$) axioms. Figure 1 illustrates an example in the medical domain. For convenience, we write $\mathsf{Sig}(\mathcal{O})$ (resp., $\mathsf{Sig}(\alpha)$, $\mathsf{Sig}(C)$)

$$
\begin{array}{lrcl}
\alpha_1 & \mathsf{Pericardium} & \sqsubseteq & \mathsf{Tissue} \sqcap \exists \mathsf{contained\text{-}in}.\mathsf{Heart} \\
\alpha_2 & \mathsf{Endocardium} & \sqsubseteq & \mathsf{Tissue} \sqcap \exists \mathsf{part\text{-}of}.\mathsf{HeartValve} \\
\alpha_3 & \mathsf{Pericarditis} & \sqsubseteq & \mathsf{Inflammation} \sqcap \exists \mathsf{has\text{-}location}.\mathsf{Pericardium} \\
\alpha_4 & \mathsf{Endocarditis} & \sqsubseteq & \mathsf{Inflammation} \sqcap \exists \mathsf{has\text{-}location}.\mathsf{Endocardium} \\
\alpha_5 & \mathsf{Inflammation} & \sqsubseteq & \mathsf{Disease} \sqcap \exists \mathsf{acts\text{-}on}.\mathsf{Tissue} \\
\alpha_6 & \mathsf{Disease} \sqcap \exists \mathsf{has\text{-}location}.\mathsf{Heart} & \sqsubseteq & \mathsf{HeartDisease} \\
\alpha_7 & \mathsf{HeartDisease} & \sqsubseteq & \exists \mathsf{has\text{-}state}.\mathsf{NeedsTreatment} \\
\alpha_8 & \mathsf{part\text{-}of} \circ \mathsf{part\text{-}of} & \sqsubseteq & \mathsf{part\text{-}of} \\
\alpha_9 & \mathsf{has\text{-}location} \circ \mathsf{contained\text{-}in} & \sqsubseteq & \mathsf{has\text{-}location}
\end{array}
$$

Fig. 1. An example $\mathcal{EL}^+$ ontology $\mathcal{O}_{\mathsf{ex}}$

to denote the signature of the ontology $\mathcal{O}$ (resp., the axiom α, the concept C), i.e., concept and role names occurring in it. Also, let $\mathsf{CN}^\top(\mathcal{O})$ denote the set of $\top$ and concept names occurring in $\mathcal{O}$.

The main inference problem for concepts is *subsumption query*: given an ontology $\mathcal{O}$ and two concept descriptions C, D, check if C is subsumed by (i.e., more specific than) D w.r.t. $\mathcal{O}$, written $C \sqsubseteq_{\mathcal{O}} D$. From our example ontology, it is not difficult to draw that $\mathsf{Pericarditis} \sqsubseteq_{\mathcal{O}_{\mathsf{ex}}} \exists \mathsf{has\text{-}state}.\mathsf{NeedsTreatment}$. The identification of subsumption relationships between *all* pairs of concept names occurring in $\mathcal{O}$ is known as *ontology classification*.

The semantics of $\mathcal{EL}^+$ ontologies, as well as of subsumption, is defined by means of interpretations in the standard way, and we refer the reader to [12,1].

3 Modules Based on Connected Reachability

In this section, we introduce a new kind of module based on *connected reachability*, and propose an algorithm for extracting the modules of this kind. We also show that, in the DL $\mathcal{EL}^+$, our modules indeed correspond to modules based on syntactic locality first introduced in [6]. We start by giving the general definition of module:

Definition 1 (Modules for an axiom and a signature). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, and $\mathcal{O}'$ a (possibly empty) set of axioms from $\mathcal{O}$. We say that $\mathcal{O}'$ is a* module in $\mathcal{O}$ for an axiom α *(for short,* α-module in $\mathcal{O}$ *) if: $\mathcal{O}' \models \alpha$ iff $\mathcal{O} \models \alpha$.*

We say that $\mathcal{O}'$ is a module for a signature $\mathbf{S}$ *if, for every axiom α with* $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$*, we have that $\mathcal{O}'$ is an α-module in $\mathcal{O}$.*

Intuitively, a module of an ontology $\mathcal{O}$ is a subset $\mathcal{O}' \subseteq \mathcal{O}$ that preserves an axiom of interest or the axioms over a signature of interest. Observe that this is a very generic definition, in the sense that the whole ontology is itself a module. In the following, we are interested in certain sufficient conditions that not only help extract a module according to Definition 1 but also guarantee relevancy of the extracted axioms. Note that if $\mathcal{O} \models \alpha$, a justification (minimal axiom set that has the consequence) is a minimal α-module in $\mathcal{O}$. A justification covers one axiom, not the axioms over a signature, thus it is normally expensive to obtain and involve standard inference reasoning, such as subsumption. For this reason, various syntactic approaches to extracting ontology fragments have been proposed in the literature [9,11,6]. In [6], Cuenca Grau et al. introduced a kind of module based on so-called syntactic locality for $\mathcal{SHOIQ}$. Though $\mathcal{EL}^+$ is not a sublanguage of $\mathcal{SHOIQ}$ due to RIs, the definition from [6] can be straightforwardly adjusted to suit $\mathcal{EL}^+$ as shown below:

Definition 2 (Locality-based modules). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, and* $\mathbf{S}$ *a signature. The following grammar recursively defines* $\mathbf{Con}^\perp(\mathbf{S})$*:*

$$\mathbf{Con}^\perp(\mathbf{S}) ::= A^\perp \mid (C^\perp \sqcap C) \mid (C \sqcap C^\perp) \mid (\exists r.C^\perp) \mid (\exists r^\perp.C)$$

with r is a role name, C a concept description, $A^\perp, r^\perp \notin \mathbf{S}$, and $C^\perp \in \mathbf{Con}^\perp(\mathbf{S})$.

An $\mathcal{EL}^+$ axiom α is syntactically local *w.r.t.* $\mathbf{S}$ *if it is one of the following forms:* (1) *RI $R^\perp \sqsubseteq s$ where $R^\perp$ is either a role name $r^\perp \notin \mathbf{S}$ or a role composition $r_1 \circ \cdots \circ r_n$ with $r_i \notin \mathbf{S}$ for some $i \leq n$, or* (2) *GCI $C^\perp \sqsubseteq C$ where $C^\perp \in \mathbf{Con}^\perp(\mathbf{S})$. We write* $\mathsf{local}(\mathbf{S})$ *to denote the collection of all $\mathcal{EL}^+$ axioms that are syntactically local w.r.t.* $\mathbf{S}$*.*

If $\mathcal{O}$ can be partitioned into $\mathcal{O}'$ and $\mathcal{O}''$ s.t. every axiom in $\mathcal{O}''$ is syntactically local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}')$, then $\mathcal{O}'$ is a locality-based module *for* $\mathbf{S}$ *in* $\mathcal{O}$*.*

Now we consider the optimization techniques of "reachability" that are used to heuristically determine obvious subsumption and non-subsumption relationships. The reachability heuristic for non-subsumption can easily be exploited in module extraction for $\mathcal{EL}^+$ ontologies. To obtain a more satisfactory module size, however, we introduce a more appropriate (i.e., stronger) reachability notion and develop an algorithm for extracting modules based on this notion.

Definition 3 (Strong/weak reachability). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, and $A, B \in \mathsf{CN}^\top(\mathcal{O})$. The strong (weak) reachability graph $\mathcal{G}_s(\mathcal{O})$ ($\mathcal{G}_w(\mathcal{O})$) for $\mathcal{O}$ is a tuple (V_s, E_s) ((V_w, E_w)) with $V_s = \mathsf{CN}^\top(\mathcal{O})$ ($V_w = \mathsf{CN}^\top(\mathcal{O})$) and E_s (E_w) the smallest set containing an edge (A, B) if $B = \top$ or $A \sqsubseteq D \in \mathcal{O}$ s.t. B is a conjunct in D (if $B = \top$ or $C \sqsubseteq D \in \mathcal{O}$ s.t. $A \in \mathsf{Sig}(C)$ and $B \in \mathsf{Sig}(D)$).*

We say that B is strongly reachable *(*weakly reachable*) from A in $\mathcal{O}$ if there is a path from A to B in $\mathcal{G}_s(\mathcal{O})$ ($\mathcal{G}_w(\mathcal{O})$).*

Observe that B is strongly reachable from A in $\mathcal{O}$ implies $A \sqsubseteq_\mathcal{O} B$, while $A \sqsubseteq_\mathcal{O} B$ implies that B is weakly reachable from A in $\mathcal{O}$.

The weak reachability graph $\mathcal{G}_w(\mathcal{O})$ for $\mathcal{O}$ can be extended in a straightforward way to cover all the symbols in $\mathcal{O}$, i.e., also role names. Precisely, we define the extension as $\mathcal{G}'_w(\mathcal{O}) := (\mathsf{Sig}(\mathcal{O}) \cup \{\top\}, E'_w)$ with $(x, y) \in E'_w$ iff $y = \top$ or there is an axiom $\alpha_L \sqsubseteq \alpha_R \in \mathcal{O}$ s.t. $x \in \mathsf{Sig}(\alpha_L)$ and $y \in \mathsf{Sig}(\alpha_R)$. A module for $\mathbf{S} = \{A\}$ in an ontology $\mathcal{O}$ based on extended weak reachability can be extracted as follows: construct $\mathcal{G}'_w(\mathcal{O})$, extract all the paths from A in $\mathcal{G}_w(\mathcal{O})$, and finally, accumulate axioms responsible for the edges in those paths. However, this kind of module is relatively large, and many axioms are often irrelevant.

For example, any GCIs with $\mathsf{Disease}$ appearing on the left-hand side, such as $\mathsf{Disease} \sqcap \exists\mathsf{has\text{-}location}.\mathsf{Brain} \sqsubseteq \mathsf{BrainDisease}$, would be extracted as part of the module for $\mathbf{S} = \{\mathsf{Pericarditis}\}$. This axiom is irrelevant since $\mathsf{Pericarditis}$ does not refer to Brain and thus $\mathsf{BrainDisease}$. Such a module would end up comprising the definitions of all disease concepts. To rule out this kind of axioms, we make the notion of reachability graph stronger as follows: All symbols appearing on the left-hand side (e.g., $\mathsf{Disease}$, $\mathsf{has\text{-}location}$ and Brain) are viewed as a connected node in the graph, which has an edge to each symbol (e.g., $\mathsf{BrainDisease}$) on the right-hand side of the axiom. The connected node is reachable from x iff all symbols participating in it are reachable from x. In our example, since Brain is not reachable from $\mathsf{Pericarditis}$, neither is $\mathsf{BrainDisease}$. Therefore, the axiom is not extracted as part of the refined module.

Definition 4 (Connected reachability and modules). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, $\mathbf{S} \subseteq \mathsf{Sig}(\mathcal{O})$ a signature, and $x, y \in \mathsf{Sig}(\mathcal{O})$ concept or role names. We say that x is* connectedly reachable *from $\mathbf{S}$ w.r.t. $\mathcal{O}$ (for short,* $\mathbf{S}$-reachable*) iff $x \in \mathbf{S}$ or there is an axiom (either GCI or RI) $\alpha_L \sqsubseteq \alpha_R \in \mathcal{O}$ s.t. $x \in \mathsf{Sig}(\alpha_R)$ and, for all $y \in \mathsf{Sig}(\alpha_L)$, y is reachable from $\mathbf{S}$.*

We say that an axiom $\beta_L \sqsubseteq \beta_R$ is connected reachable *from $\mathbf{S}$ w.r.t. $\mathcal{O}$ (for short,* $\mathbf{S}$-reachable*) if, for all $x \in \mathsf{Sig}(\beta_L)$, x is $\mathbf{S}$-reachable.* The reachability-based module *for $\mathbf{S}$ in $\mathcal{O}$, denoted by $\mathcal{O}^{\mathsf{reach}}_{\mathbf{S}}$, is the set of all $\mathbf{S}$-reachable axioms.*

Intuitively, x is connectedly reachable from $\{y\}$ w.r.t. $\mathcal{O}$ means that y syntactically refers to x, either directly or indirectly via axioms in $\mathcal{O}$. If x, y are concept names, then the reachability suggests a potential subsumption relationship $y \sqsubseteq_{\mathcal{O}} x$. Note, in particular, that axioms of the forms $\top \sqsubseteq D$ and $\epsilon \sqsubseteq r$ in $\mathcal{O}$ are connectedly reachable from any signature because $\mathsf{Sig}(\top) = \mathsf{Sig}(\epsilon) = \emptyset$, and therefore occur in every reachability-based module. In our example, $\mathcal{O}^{\mathsf{reach}}_{\{\mathsf{Pericarditis}\}}$ contains axioms $\alpha_1, \alpha_3, \alpha_5$–$\alpha_7$ and α_9. We now show some properties of connected reachability and reachability-based modules that are essential for establishing the subsequent lemma and theorem:

Proposition 1 (Properties of reachability and $\mathcal{O}^{\mathsf{reach}}_{\mathbf{S}}$). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, $\mathbf{S}, \mathbf{S}_1, \mathbf{S}_2 \subseteq \mathsf{Sig}(\mathcal{O})$ signatures, x, y, z symbols in $\mathsf{Sig}(\mathcal{O})$, and A, B concept names in $\mathsf{CN}(\mathcal{O})$. Then, the following properties hold:*

1. *If $\mathbf{S}_1 \subseteq \mathbf{S}_2$, then $\mathcal{O}^{\mathsf{reach}}_{\mathbf{S}_1} \subseteq \mathcal{O}^{\mathsf{reach}}_{\mathbf{S}_2}$.*
2. *If x is $\{y\}$-reachable and y is $\{z\}$-reachable, then x is $\{z\}$-reachable.*
3. *If x is connected reachable from $\{y\}$ w.r.t. $\mathcal{O}$, then $\mathcal{O}^{\mathsf{reach}}_{\{x\}} \subseteq \mathcal{O}^{\mathsf{reach}}_{\{y\}}$*
4. *$x \in \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}^{\mathsf{reach}}_{\mathbf{S}})$ if, and only if, x is $\mathbf{S}$-reachable w.r.t. $\mathcal{O}$.*
5. *If B is* not *connected reachable from $\{A\}$ w.r.t. $\mathcal{O}$, then $A \not\sqsubseteq_{\mathcal{O}} B$.*

The converse of Point 5 is not true in general, for instance, $\mathsf{Pericarditis}$ involves Tissue, but the corresponding subsumption does not follow from the ontology. This suggests that we could use connected reachability as a heuristic for answering negative subsumption, in a similar but finer way as in weak reachability.

We outline our algorithm for extracting the reachability-based module given a signature $\mathbf{S}$ and an ontology $\mathcal{O}$ in Algorithm 1. Similar to the technique developed

Algorithm 1. extract-module

```
Input: $\mathcal{O}$: $\mathcal{EL}^+$ ontology; $\mathbf{S}$: signature
Output: $\mathcal{O}_\mathbf{S}$: reachability-based module for $\mathbf{S}$ in $\mathcal{O}$
1: $\mathcal{O}_\mathbf{S} \leftarrow \emptyset$
2: queue $\leftarrow$ active-axioms($\mathbf{S}$)
3: while not empty(queue) do
4:     $(\alpha_L \sqsubseteq \alpha_R) \leftarrow$ fetch(queue)
5:     if Sig$(\alpha_L) \subseteq \mathbf{S} \cup$ Sig$(\mathcal{O}_\mathbf{S})$ then
6:         $\mathcal{O}_\mathbf{S} \leftarrow \mathcal{O}_\mathbf{S} \cup \{\alpha_L \sqsubseteq \alpha_R\}$
7:         queue $\leftarrow$ queue $\cup$ (active-axioms(Sig$(\alpha_R)) \setminus \mathcal{O}_\mathbf{S}$)
8: return $\mathcal{O}_\mathbf{S}$
```

in [3], we view the input ontology $\mathcal{O}$ as a mapping active-axioms : $\mathsf{Sig}(\mathcal{O}) \rightarrow \mathcal{O}$ with active-axioms(x) comprising all and only axioms $\alpha_L \sqsubseteq \alpha_R \in \mathcal{O}$ such that x occurs in α_L. The main differences, compared to the $\widehat{\mathcal{O}}$ mapping in [3] (also used in Section 4), are that active-axioms does not assume the input ontology to be in normal form, and that it is defined for both concept and role names. The intuition is that every axiom $\alpha \in$ active-axioms(x) is "active" for x, in the sense that y could be connectedly reachable via α from x for some $y \in \mathsf{Sig}(\mathcal{O})$. For convenience, we define active-axioms$(\mathbf{S}) := \bigcup_{x \in \mathbf{S}}$ active-axioms(x) for a signature $\mathbf{S} \subseteq \mathsf{Sig}(\mathcal{O})$.

It is easy to see that each axiom Algorithm 1 extracts to $\mathcal{O}_\mathbf{S}$ is $\mathbf{S}$-reachable. The fact that all $\mathbf{S}$-reachable axioms are extracted to $\mathcal{O}_\mathbf{S}$ can be proved by induction on connected reachability.

Proposition 2 (Algorithm 1 produces $\mathcal{O}_\mathbf{S}^{\mathsf{reach}}$). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology and $\mathbf{S} \subseteq \mathsf{Sig}(\mathcal{O})$ a signature. Then, Algorithm 1 returns the reachability-based module for $\mathbf{S}$ in $\mathcal{O}$.*

In fact, connected reachability can be reduced to propositional Horn clause implication. The idea is to translate each $\mathcal{EL}^+$ axiom $\alpha_L \sqsubseteq \alpha_R$ into the Horn clause $l_1 \wedge \cdots \wedge l_m \rightarrow r_1 \wedge \cdots \wedge r_n$ where $l_i \in \mathsf{Sig}(\alpha_L)$ and $r_i \in \mathsf{Sig}(\alpha_R)$. Given a signature $\mathbf{S}$ and a symbol x, x is $\mathbf{S}$-reachable iff x is implied by $\bigwedge_{y \in \mathbf{S}} y$ w.r.t. the Horn clauses. The Dowling-Gallier algorithm [4] can check this in linear time.

In the following, we show a tight relationship between our reachability-based modules and the (minimal) locality-based modules.

Theorem 1 ($\mathcal{O}_\mathbf{S}^{\mathsf{reach}}$ is the minimal locality-based module). *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, and $\mathbf{S} \subseteq \mathsf{Sig}(\mathcal{O})$ a signature. Then, $\mathcal{O}_\mathbf{S}^{\mathsf{reach}}$ is the minimal locality-based module for $\mathbf{S}$ in $\mathcal{O}$.*

So, Algorithm 1 can be used to extract a locality-based module in an $\mathcal{EL}^+$ ontology. The main difference, in contrast to the algorithm used in [6,5], is that our algorithm considers only "active" axioms for α_R when a new axiom $\alpha_L \sqsubseteq \alpha_R$ is extracted. Also, testing whether an $\mathcal{EL}^+$ axiom $\alpha = (\alpha_L \sqsubseteq \alpha_R)$ is non-local w.r.t. a signature $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_\mathbf{S})$ boils down to testing $\mathbf{S}$-reachability of α, which is a simpler operation of testing set inclusion $\mathsf{Sig}(\alpha_L) \subseteq^? \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_\mathbf{S})$. This is due to the fact that any concept description and role composition α_L, with $x \in \mathsf{Sig}(\alpha_L)$ interpreted as the empty set, is itself interpreted as the empty set. This observation could be used to optimize module extraction for ontologies in expressive description logics.

It has been shown for $\mathcal{SHOIQ}$ that locality-based modules for $\mathbf{S} = \{A\}$ in $\mathcal{O}$ preserves the subsumption $A \sqsubseteq B$ for any $B \in \mathsf{CN}(\mathcal{O})$ [6]. This property could have been transferred to our setting as a corollary of Theorem 1 if $\mathcal{EL}^+$ were a sublanguage of $\mathcal{SHOIQ}$. Despite this not being the case, it is not hard to show that reachability-based modules in $\mathcal{EL}^+$ also enjoy the property:

Lemma 1 **($\mathcal{O}_A^{\mathsf{reach}}$ preserves $A \sqsubseteq_{\mathcal{O}} B$).** *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, $A \in \mathsf{CN}(\mathcal{O})$, and $\mathcal{O}_{\{A\}}^{\mathsf{reach}}$ the reachability-based module for $\mathbf{S} = \{A\}$ in $\mathcal{O}$. Then, for any $\alpha = A \sqsubseteq B$ with $B \in \mathsf{CN}(\mathcal{O})$, $\mathcal{O} \models \alpha$ iff $\mathcal{O}_{\{A\}}^{\mathsf{reach}} \models \alpha$.*

4 Goal-Directed Subsumption Algorithm

In general, the techniques developed for module extraction have a number of potential applications, including optimization of standard reasoning, incremental classification, explanation, and ontology re-use. An obvious way to exploit module extraction to speed up standard reasoning, such as subsumption $\phi \sqsubseteq_{\mathcal{O}}^? \psi$, is to first extract the module $\mathcal{O}_{\{\phi\}}^{\mathsf{reach}}$ for $\mathbf{S} = \{\phi\}$ in $\mathcal{O}$, and then query the subsumption $\phi \sqsubseteq_{\mathcal{O}_{\{\phi\}}^{\mathsf{reach}}}^? \psi$, i.e., w.r.t. the module instead of the original ontology. Based on the assumption that modules are relatively much smaller than the ontology, this optimization should be highly effective. In this section, however, we argue that module extraction actually does not help speed up standard reasoning in $\mathcal{EL}^+$. This stems from the deterministic and goal-directed nature of the reasoning algorithm for deciding subsumption in $\mathcal{EL}^+$, which is in contrast to non-deterministic tableau-based algorithms for expressive logics, such as $\mathcal{SHOIQ}$.

In fact, with small modifications to the $\mathcal{EL}^+$ classification algorithm (first introduced in [1] for $\mathcal{EL}^{++}$ and later refined for implementation in [3]), we obtain a subsumption testing algorithm. The modified algorithm does not actually have to perform steps irrelevant to the subsumption in question – *the goal*. We call this variant the *goal-directed subsumption algorithm*.

Algorithm 2 outlines the modified core procedure goal-directed-process to replace process of Figure 3 in [3]. The procedure process-new-edge, as well as essential data structures, i.e., $\widehat{\mathcal{O}}$, queue, R, S, remains intact. In particular, we view the (normalized) input ontology $\mathcal{O}$ as a mapping $\widehat{\mathcal{O}}$ from concepts (appearing on the left-hand side of some GCI) to sets of queue entries. Here, $\mathbf{B}$ denotes the set of all concept names appearing in the conjunction $B_1 \sqcap \cdots \sqcap B_n$.

The main difference is the initialization of S, thus of queue. Since we are interested in the particular subsumption $\phi \sqsubseteq \psi$, we "activate" only ϕ by initializing $S(\phi)$ with $\{\phi, \top\}$ and queue(ϕ) with $\widehat{\mathcal{O}}(\phi) \cup \widehat{\mathcal{O}}(\top)$. We activate a concept name B *only* when it becomes the second component of a tuple added to some $R(r)$ and has not been activated previously (see lines 8-9 in goal-directed-process of Algorithm 2). Thereby, $S(B)$ and queue(B) are initialized accordingly. Queues

Algorithm 2. Goal-directed subsumption algorithm

```
Procedure subsumes($\phi \sqsubseteq \psi$)
Input: ($\phi \sqsubseteq \psi$): target subsumption
Output: 'positive' or 'negative' answer to the subsumption
 1: activate($\phi$)
 2: while not empty(queue($A$)) for some $A \in \mathsf{CN}(\mathcal{O})$ do
 3:     $X \leftarrow$ fetch(queue($A$))
 4:     if goal-directed-process($A, X, \phi \sqsubseteq \psi$) then
 5:         return 'positive'
 6: return 'negative'

Procedure goal-directed-process($A, X, \phi \sqsubseteq \psi$)
Input: $A$: concept name; $X$: queue entry; ($\phi \sqsubseteq \psi$): target subsumption
Output: 'positive' or 'unknown' answer to the subsumption
 1: if $X = \mathbf{B} \to B$, $\mathbf{B} \subseteq S(A)$ and $B \notin S(A)$ then
 2:     $S(A) := S(A) \cup \{B\}$
 3:     queue($A$) := queue($A$) $\cup\ \widehat{\mathcal{O}}(B)$
 4:     for all concept names $A'$ and role names $r$ with $(A', A) \in R(r)$ do
 5:         queue($A'$) := queue($A'$) $\cup\ \widehat{\mathcal{O}}(\exists r.B)$
 6:     if $A = \phi$ and $B = \psi$ then
 7:         return 'positive'
 8: if $X = \exists r.B$ and $(A, B) \notin R(r)$ then
 9:     activate($B$)
10:     process-new-edge($A, r, B$)
11: return 'unknown'
```

are processed in the same fashion as before except that ϕ and ψ are now being monitored (Line 6), so that immediately after ψ is added to $S(\phi)$, the algorithm terminates with the positive answer (Line 7). Otherwise, goal-directed-process terminates normally, and the next queue entry will be fetched (Line 3 in subsumes? of Algorithm 2) and processed (Line 4). Unless 'positive' is returned, queues processing is continued until they are all empty. In this case, the algorithm returns 'negative.'

It is important to note that the goal-directed algorithm activates only concept names relevant to the target subsumption $\phi \sqsubseteq \psi$, i.e., those reachable via $R(\cdot)$ from ϕ. The subsumer sets of concept names that do not become activated are not populated. Moreover, axioms that are involved in rule applications during the computation of subsumes?($\phi \sqsubseteq \psi$) are those from the reachability-based module $\mathcal{O}^{\mathsf{reach}}_{\{\phi\}}$ in $\mathcal{O}$. The following proposition states this correlation:

Proposition 3 (subsumes?($\phi \sqsubseteq \psi$) only requires axioms in $\mathcal{O}^{\mathsf{reach}}_{\phi}$). *Let $\mathcal{O}$ be an ontology in $\mathcal{EL}^+$ normal form, and $\mathcal{O}^{\mathsf{reach}}_{\{\phi\}}$ the reachability-based module for $\mathbf{S} = \{\phi\}$ in $\mathcal{O}$. Then,* subsumes?($\phi \sqsubseteq \psi$) *only requires axioms in $\mathcal{O}^{\mathsf{reach}}_{\{\phi\}}$.*

Intuitively, the proposition suggests that our goal-directed subsumption algorithm inherently takes into account the notion of connected reachability, i.e., it applies rules only to relevant axioms in the reachability-based module. In fact, the preprocessing overhead of extracting the relevant module $\mathcal{O}^{\mathsf{reach}}_{\{\phi\}}$ for the subsumption query $\phi \sqsubseteq^?_{\mathcal{O}} \psi$ makes the overall computation time for an individual subsumption query longer. This has been empirically confirmed in our experiments (see the last paragraph of Section 6).

Despite what has been said, module extraction is still useful for, e.g., ontology re-use, explanation, and full-fledged incremental reasoning [5].

5 Duo-Ontology Classification

Unlike tableau-based algorithms, the polynomial-time algorithm in [1,3] inherently classifies the input ontology by making all subsumptions between *concept names* explicit. This algorithm can be used to query subsumption between concept names occurring in the ontology, but complex subsumptions, such as

$$\mathsf{Inflammation} \sqcap \exists \mathsf{has\text{-}location.Heart} \sqsubseteq^?_{\mathcal{O}_{\mathsf{ex}}} \mathsf{HeartDisease} \sqcap \exists \mathsf{has\text{-}state.NeedsTreatment}$$

cannot be answered directly. First, the ontology $\mathcal{O}_{\mathsf{ex}}$ from Figure 1 has to be augmented to $\mathcal{O}'_{\mathsf{ex}} := \mathcal{O}_{\mathsf{ex}} \cup \{A \sqsubseteq \mathsf{Inflammation} \sqcap \exists \mathsf{has\text{-}location.Heart}, \mathsf{HeartDisease} \sqcap \exists \mathsf{has\text{-}state.NeedsTreatment} \sqsubseteq B\}$ with A, B new concept names, and then the subsumption test $A \sqsubseteq^?_{\mathcal{O}'_{\mathsf{ex}}} B$ can be carried out to decide the original complex subsumption. Since A, B are new names not occurring in $\mathcal{O}_{\mathsf{ex}}$, our complex subsumption holds iff $A \sqsubseteq_{\mathcal{O}'_{\mathsf{ex}}} B$. This approach is effective but inefficient unless only one such complex subsumption is queried for each ontology. Constructing and normalizing the augmented ontology every time each subsumption is tested is not likely to be acceptable in practice, especially when the background ontology is large. For instance, normalization of SNOMED CT takes more than one minute.

In this section, we propose an extension to the refined algorithm (henceforth referred to as *the original algorithm*) developed in [3] to cater for a *duo-ontology* $\mathcal{O} = (\mathcal{O}_p \cup \mathcal{O}_t)$ with $\mathcal{O}_p$ a *permanent* $\mathcal{EL}^+$ ontology and $\mathcal{O}_t$ a set of *temporary* GCIs. Intuitively, $\mathcal{O}_p$ is the input ontology of which axioms have been read in and processed before, while $\mathcal{O}_t$ contains temporary GCIs that are asserted later. The main purpose is to reuse the information made available by the preprocess and classification of $\mathcal{O}_p$. Once $\mathcal{O}_p$ has been classified, the classification of $\mathcal{O}_p \cup \mathcal{O}_t$ should not start from scratch but rather use the existing classification information together with the new GCIs from $\mathcal{O}_t$ to do incremental classification.

In our extension, we use two sets of the core data structures $\widehat{\mathcal{O}}(\cdot), R(\cdot), S(\cdot)$, but retain a single set of queues $\mathsf{queue}(\cdot)$. The mappings $\widehat{\mathcal{O}}_p, R_p, S_p$ are initialized and populated exactly as in the original algorithm, i.e., $\widehat{\mathcal{O}}_p$ encodes axioms in $\mathcal{O}_p$, and R_p, S_p store subsumption relationships inferred from $\mathcal{O}_p$. Similarly, the mapping $\widehat{\mathcal{O}}_t$ encodes axioms in $\mathcal{O}_t$, but R_t, S_t represent additional inferred subsumptions drawn from $\mathcal{O}_p \cup \mathcal{O}_t$ that are not already present in R_p, S_p, respectively. The extended algorithm is based on the tenet that description logics are monotonic, i.e., $\mathcal{O}_p \models \alpha$ implies $\mathcal{O}_p \cup \mathcal{O}_t \models \alpha$. There may be an additional consequence β such that $\mathcal{O}_p \not\models \beta$ but $\mathcal{O}_p \cup \mathcal{O}_t \models \beta$. Our algorithm stores such a consequence β in a separate set of data structures, namely R_p, S_p. Analogously to the original algorithm, queue

Algorithm 3. Processing queue entries in duo-ontology classification

```
Procedure process-duo($A, X$)
Input: $A$: concept name; $X$: queue entry;
1: if $X = \mathbf{B} \rightarrow B$, $\mathbf{B} \subseteq S_p(A) \cup S_t(A)$ and $B \notin S_p(A) \cup S_t(A)$ then
2:     $S_t(A) := S_t(A) \cup \{B\}$
3:     $\mathsf{queue}(A) := \mathsf{queue}(A) \cup \widehat{\mathcal{O}}_p(B) \cup \widehat{\mathcal{O}}_t(B)$
4:     for all $A'$ and $r$ with $(A', A) \in R_p(r) \cup R_t(r)$ do
5:         $\mathsf{queue}(A') := \mathsf{queue}(A') \cup \widehat{\mathcal{O}}_p(\exists r.B) \cup \widehat{\mathcal{O}}_t(\exists r.B)$
6: if $X = \exists r.B$ and $(A, B) \notin R_p(r) \cup R_t(r)$ then
7:     process-new-edge($A, r, B$)

Procedure process-new-edge-duo($A, r, B$)
Input: $A, B$: concept names; $r$: role name;
1: for all role names $s$ with $r \sqsubseteq^*_{\mathcal{O}_p} s$ do
2:     $R_t(s) := R_t(s) \cup \{(A, B)\}$
3:     $\mathsf{queue}(A) := \mathsf{queue}(A) \cup \bigcup_{\{B' | B' \in S_p(B) \cup S_t(B)\}} (\widehat{\mathcal{O}}_p(\exists s.B') \cup \widehat{\mathcal{O}}_t(\exists s.B'))$
4:     for all concept name $A'$ and role names $u, v$ with $u \circ s \sqsubseteq v \in \mathcal{O}_p$ and
       $(A', A) \in R_p(u) \cup R_t(u)$ and $(A', B) \notin R_p(v) \cup R_t(v)$ do
5:         process-new-edge-duo($A', v, B$)
6:     for all concept name $B'$ and role names $u, v$ with $s \circ u \sqsubseteq v \in \mathcal{O}_p$ and
       $(B, B') \in R_p(u) \cup R_t(u)$ and $(A, B') \notin R_p(v) \cup R_t(v)$ do
7:         process-new-edge-duo($A, v, B'$)
```

entries are repeatedly fetched and processed until all queues are empty. Instead of the procedures process and process-new-edge, we use the extended versions for duo-ontology classification as outlined in Algorithm 3.

The behavior of Algorithm 3 is identical to that of the original one [3] if $\mathcal{O}_p$ has not been classified before. In particular, $\widehat{\mathcal{O}}_p(\cdot) \cup \widehat{\mathcal{O}}_t(\cdot)$ here is equivalent to $\widehat{\mathcal{O}}(\cdot)$ in [3] given that $\mathcal{O} = (\mathcal{O}_p \cup \mathcal{O}_t)$. Since no classification has taken place, $S_p(A) = R_p(r) = \emptyset$ for each concept name A and role name r. Initialization and processing of queues are done in the same manner with the only difference that inferred consequences are now put in R_t and S_t.

If $\mathcal{O}_p$ has been classified (thus, S_p, R_p have been populated), then a proper initialization has to be done w.r.t. the previously inferred consequences (i.e., S_p, R_p) and the new GCIs (i.e., $\widehat{\mathcal{O}}_t$). To this end, we initialize the data structures by setting:

– for each role name $r \in \mathsf{RN}(\mathcal{O})$, $R_t(r) := \emptyset$;
– for each *old* concept name $A \in \mathsf{CN}(\mathcal{O}_p)$, $S_t(A) := \emptyset$ and $\mathsf{queue}(A) := \bigcup_{X \in S_p(A)} \widehat{\mathcal{O}}_t(X) \ \cup \ \bigcup_{\{(A,B) \in R_p(r), X \in S_p(B)\}} \widehat{\mathcal{O}}_t(\exists r.X)$;
– for each *new* concept name $A \in \mathsf{CN}(\mathcal{O}_t) \setminus \mathsf{CN}(\mathcal{O}_p)$, $S_t(A) := \{A, \top\}$ $\mathsf{queue}(A) := \widehat{\mathcal{O}}_t(A) \cup \widehat{\mathcal{O}}_t(\top)$.

After initialization, queue processing is carried out by Algorithm 3 until all queues are empty. Observe the structural analogy between these procedures and the original ones in [3]. Observe also the key difference: information is always retrieved from both sets of data structures, e.g., $S_p(A) \cup S_t(A)$ in Line 1, while modifications are only made to the temporary set of data structures, e.g., $S_t(A) := S_t(A) \cup \{B\}$ in Line 2. The correctness of Algorithm 3 can be shown following the correctness proofs structures of the original algorithm (see the submitted journal version of [3]) w.r.t. additional subsumption consequences obtained during incremental classification.

Lemma 2 (Correctness of Algorithm 3). *Let $\mathcal{O} = (\mathcal{O}_p \cup \mathcal{O}_t)$ be a duo-ontology, and S_p, R_p be the results after the original algorithm terminates on $\mathcal{O}_p$. Then, the extended algorithm (Algorithm 3), applied to $\mathcal{O}_t$, incrementally classifies $\mathcal{O}_t$ against $\mathcal{O}_p$ (i.e., classifies $\mathcal{O}$) in time polynomial in the size of $\mathcal{O}$. That is, $B \in S_p(A) \cup S_t(A)$ iff $A \sqsubseteq_{\mathcal{O}} B$ for all $A, B \in \mathsf{CN}(\mathcal{O})$.*

In our example, we may view $\mathcal{O}_{\mathsf{ex}}$ as the permanent ontology $\mathcal{O}_p$ and the two new GCIs as the temporary ontology $\mathcal{O}_t$. We can then run the extended algorithm on $\mathcal{O}_p \cup \mathcal{O}_t$ and reuse existing information in S_p and R_p, if any. After termination, our complex subsumption boils down to the set membership test $B \in^? S_p(A) \cup S_t(A) = S_t(A)$. To decide subsequent subsumption queries, only $\mathcal{O}_t, R_t, S_t$, and queue need to be initialized, leaving the background ontology $\mathcal{O}_p$ and possibly its classification information R_t, S_t intact.

Interestingly, this algorithm can be used effectively in certain scenarios of incremental classification. Consider $\mathcal{O}_p$ as a well-developed, permanent ontology, and $\mathcal{O}_t$ as a small set of temporary axioms currently being authored. Obviously, if the permanent ontology is large, it would be impractical to reclassify from scratch every time some new axioms are to be added. Algorithm 3 incrementally classifies $\mathcal{O}_t$ against $\mathcal{O}_p$ and its classification information. If the inferred consequences are satisfactory, the temporary axioms can be committed to the permanent ontology by merging the two sets of data structures. Otherwise, axioms in $\mathcal{O}_t$ and their inferred consequences could be easily retracted, since these are segregated from $\mathcal{O}_p$ and its consequences. To be precise, we simply dump the values of $\mathcal{O}_t(\cdot), R_t(\cdot)$ and $S_t(\cdot)$, when the temporary axioms are retracted.

6 Experiments and Empirical Results

This section describes the experiments and results of the three algorithms we proposed in this paper: module extraction, goal-directed subsumption query, and duo-ontology classification, which have been implemented and integrated as new features into the CEL reasoner [2] version 1.0b. All the experiments have been carried out on a standard PC: 2.40 GHz Pentium-4 processor and 1 GB of physical memory. In order to show interesting characteristics of reachability-based modules and scalability of subsumption and incremental classification in $\mathcal{EL}^+$, we have selected a number of large-scale medical ontologies. Our test suite comprises Snomed ct, Nci, and the $\mathcal{EL}^+$ fragments[1] of FullGalen and

[1] FullGalen is precisely based on $\mathcal{SHIF}$ dispensed with negation, disjunction, and value restriction. The DL $\mathcal{EL}^+$ can indeed express most of its axioms, namely 95.75%, and we obtained this fragment for experimental purposes by dropping role inverse and functionality axioms.

Table 1. $\mathcal{EL}^+$ ontology test suite

Ontologies	♯Concepts/roles	♯Concept/role axioms	Class. time (sec)	Positive subs. (%)
$\mathcal{O}^{\text{NotGalen}}$	2 748 / 413	3 937 / 442	7.36	0.6013
$\mathcal{O}^{\text{FullGalen}}$	23 136 / 950	35 531 / 1 016	512.72	0.1648
$\mathcal{O}^{\text{Nci}}$	27 652 / 70	46 800 / 140	7.01	0.0441
$\mathcal{O}^{\text{Snomed}}$	379 691 / 62	379 691 / 13	1 671.23	0.0074

NotGalen, denoted respectively by $\mathcal{O}^{\text{Snomed}}$, $\mathcal{O}^{\text{Nci}}$,$\mathcal{O}^{\text{FullGalen}}$, and $\mathcal{O}^{\text{NotGalen}}$.[2] The FullGalen ontology shall not be confused with the original version of Galen, the latter of which is almost 10 times smaller and commonly used in DL benchmarking. The sizes of our test suite ontologies are shown in the second and third columns of Table 1. The last but one column shows the time CEL needs to classify each ontology, while the last presents in percentage the ratio of positive subsumption relationships between concept names. Observe that all ontologies have a very low ratio of positive subsumption (less than 1%); in particular, less than a ten-thousandth of all potential subsumptions *actually* hold in $\mathcal{O}^{\text{Snomed}}$.

Modularization: For each ontology $\mathcal{O}$ in the test suite and each concept name $A \in \mathsf{CN}(\mathcal{O})$, we extracted the reachability-based module $\mathcal{O}_A^{\mathsf{reach}}$. Statistical data concerning the sizes of modules and times required to extract them are presented in Table 2. Observe that it took a tiny amount of time to extract a single module based on connected reachability, with the maximum time less than four seconds. However, extracting large the number of modules (i.e., one for each concept name) required considerably more time and even longer than classification. This was nevertheless the first implementation that was not highly optimized. Several optimization techniques could be employed in module extraction, especially recursive extraction as suggested by Point 3 of Proposition 1 and the counting techniques from [4]. To empirically support Theorem 1, we have compared our modularization algorithm to that from [5,6]. As expected, the results of both algorithms coincide w.r.t. $\mathcal{O}^{\text{NotGalen}}$ and $\mathcal{O}^{\text{Nci}}$, while we were unable to obtain locality-based modularization results w.r.t. the other two ontologies.[3]

Interestingly, module extraction reveals important structural dependencies that reflect complexity of the ontology. Though very large, concepts in $\mathcal{O}^{\text{Nci}}$ and $\mathcal{O}^{\text{Snomed}}$ are loosely connected w.r.t. reachability which makes it relatively easy to classify. In contrast, $\mathcal{O}^{\text{FullGalen}}$ contains more complex dependencies[4], thus is hard to classify.

Duo-ontology classification: As mentioned before, there are at least two applications of Algorithm 3, viz., complex subsumption query and (restricted)

[2] Obtainable at `http://lat.inf.tu-dresden.de/~meng/toyont.html`.

[3] Due to memory exhaustion with 0.8 GB of Java heap space.

[4] Based on the statistical data analysis, there are two clearly distinct groups of concepts in $\mathcal{O}^{\text{FullGalen}}$: the first with module sizes between 0 and 523 (med. 39; avg. 59.29) and the second between 14 791 and 15 545 (med. 14 792; avg. 14 829). Surprisingly, there is no module of size between those of these two groups.

Table 2. Module extraction (time in second; size in number of axioms)

Ontologies	Extraction time				Module size (%)		
	median	average	maximum	total	median	average	maximum
$\mathcal{O}^{\text{NotGalen}}$	< 0.01	~ 0.00	0.01	2.38	35 (1.27)	68.64 (2.50)	495 (18.00)
$\mathcal{O}^{\text{FullGalen}}$	0.01	0.04	0.85	960	178 (0.77)	7092 (30.65)	15 545 (67.18)
$\mathcal{O}^{\text{Nci}}$	< 0.01	~ 0.00	0.17	3.43	12 (0.026)	28.97 (0.062)	436 (0.929)
$\mathcal{O}^{\text{Snomed}}$	< 0.01	~ 0.01	3.83	3 744	18 (0.005)	30.31 (0.008)	262 (0.069)

Table 3. Incremental classification (in second)

♯Temp. axioms ($\lvert\mathcal{O}_t\rvert$)	$\mathcal{O}^{\text{NotGalen}}$		$\mathcal{O}^{\text{FullGalen}}$		$\mathcal{O}^{\text{Nci}}$		$\mathcal{O}^{\text{Snomed}}$	
	C. time	IC. time	C. time	IC. time	C. time	IC. time	C. time	IC. time
0.2%	6.53	1.75	486.19	56.94	5.10	2.00	1 666.43	55.86
0.4%	6.50	1.88	484.89	59.37	4.81	2.15	1 663.51	57.97
0.6%	6.48	2.45	482.13	62.34	4.78	2.37	1 661.49	68.58
0.8%	6.43	2.88	466.97	80.52	4.70	2.54	1 652.84	83.27
1.0%	6.38	4.46	450.61	109.81	4.59	3.19	1 640.11	93.89

incremental classification. For complex subsumption query, we have adopted the "activation" idea from Algorithm 2 to quickly answer the query. To perform meaningful experiments, it is inevitable to involve a domain expert to obtain sensible test data. Though we have done so w.r.t. $\mathcal{O}^{\text{Snomed}}$, the numbers of complex subsumption queries and additional axioms are very small compared to the ontology size.[5] For this reason, we have developed our test strategy as follows: for each ontology $\mathcal{O}$ and various numbers n, we have (i) partitioned $\mathcal{O}$ into $\mathcal{O}_p$ and $\mathcal{O}_t$ such that $\mathcal{O}_t$ contains n% of GCIs from $\mathcal{O}$; (ii) classified $\mathcal{O}_p$ normally; finally, (iii) incrementally classified $\mathcal{O}_t$ against $\mathcal{O}_p$. The average computation times for several runs of (ii) and (iii) are shown in the left and right columns of each ontology in Table 3, respectively. It requires only 4% (resp., 15%, 35%, and 38%) of the total classification time for $\mathcal{O}^{\text{Snomed}}$ (resp., for $\mathcal{O}^{\text{FullGalen}}$, $\mathcal{O}^{\text{Nci}}$, and $\mathcal{O}^{\text{NotGalen}}$) to incrementally classify up to 1% of all axioms, i.e., about four-thousand axioms in the case of $\mathcal{O}^{\text{Snomed}}$.

Subsumption: To evaluate our goal-directed algorithm, we have run subsumption tests between *random* pairs of concept names without any heuristics.[6] Average/maximum querying times (in second) are 0.09/1.51 for $\mathcal{O}^{\text{NotGalen}}$, 124.01/254.31 for $\mathcal{O}^{\text{FullGalen}}$, 0.0034/0.44 for $\mathcal{O}^{\text{Nci}}$, and 0.0183/3.32 for $\mathcal{O}^{\text{Snomed}}$.

[5] On average, a typical complex subsumption query against $\mathcal{O}^{\text{Snomed}}$ took 0.00153 *milli*seconds, while incremental classification of one axiom needed 48.74 seconds.

[6] Since there are about *144 billion pairs* of concept names in the case of $\mathcal{O}^{\text{Snomed}}$ and some subsumption queries against $\mathcal{O}^{\text{FullGalen}}$ took a few minutes, performing subsumption queries between *all* pairs would not be feasible. Therefore, one thousand random pairs of subsumption were tested against $\mathcal{O}^{\text{FullGalen}}$, and one million random pairs against each of the other ontologies.

Notice that subsumption requires a negligible amount of time and not much more than extracting a module in the case of $\mathcal{O}^{\text{NCI}}$ and $\mathcal{O}^{\text{SNOMED}}$. Observe also that subsumption querying times are roughly proportional to module sizes, which reflects the nature of the goal-directed algorithm as stated by Proposition 3.

7 Related Work

Recently, various techniques for extracting fragments of ontologies have been proposed in the literature. An example is the algorithm proposed in [11] which was developed specifically for Galen. The algorithm traverses in definitional order and into existential restrictions but does not take into account other dependencies, e.g., role hierarchy and GCIs. If applied to our example ontology $\mathcal{O}_{\mathsf{ex}}$, the algorithm extracts only α_1, α_3 and α_5 as its segmentation output for $\mathsf{Pericarditis}$. This is obviously not a module because we lose the subsumption $\mathsf{Pericarditis} \sqsubseteq_{\mathcal{O}_{\mathsf{ex}}} \mathsf{HeartDisease}$. Another example is the Prompt-Factor tool [9] which implements an algorithm that, given an ontology $\mathcal{O}$ and a signature $\mathbf{S}$, retrieves a subset $\mathcal{O}_1 \subseteq \mathcal{O}$ by retrieving to $\mathcal{O}_1$ axioms that contain symbols in $\mathbf{S}$ and extending $\mathbf{S}$ with $\mathsf{Sig}(\mathcal{O}_1)$ until a fixpoint is reached. This is similar to our modules based on *weak* reachability, but it does not distinguish symbols occurring on lhs and rhs of axioms. In our example, the tool will return the whole ontology as output for $\mathbf{S} = \{\mathsf{Pericarditis}\}$, even though several axioms are irrelevant. As we have shown, modules based on syntactic locality [6] are equivalent to our reachability-based modules relative to $\mathcal{EL}^+$ ontologies. Since reachability is much simpler to check, our algorithm has proved more efficient.

Incremental classification and reasoning have received much attention in the recent years. In [7,10], the so-called model-caching techniques have been investigated for application scenarios that only ABox is modified. A technique for incremental schema reasoning has recently been proposed in [5]: it utilizes modules to localize ramifications of changes and performs additional reasoning only on affected modules. The framework supports full-fledged incremental reasoning in the sense that arbitrary axioms can be retracted or modified, and as such it is worthwhile to investigate how its techniques can be integrated into our duo-ontology classification algorithm. All above-mentioned works focus on expressive languages. Here, however, we developed a very specific approach to (restricted) incremental classification in $\mathcal{EL}^+$. Since the technique exploits the facts that the original $\mathcal{EL}^+$ algorithm maintains completed subsumer sets, it is not obvious how this may benefit tableau-based algorithms for expressive DLs.

8 Conclusion

In this paper, we have introduced a new kind of module (based on connected reachability) and proposed an algorithm to extract them from $\mathcal{EL}^+$ ontologies. We have shown that these are equivalent to locality-based modules w.r.t. $\mathcal{EL}^+$ ontologies and empirically demonstrated that modules can be extracted in reasonable time and are reasonably small. Also, we have proposed a goal-directed variant of the algorithm in [3] for testing subsumption prior to classification and have extended this algorithm to cater for a duo-ontology which can be utilized to answer complex subsumption queries and to do (restricted) incremental classification. Our empirical results have evidently confirmed that the proposed algorithms are practically feasible in large-scale ontology applications.

Despite not being directly useful to speed up standard reasoning in $\mathcal{EL}^+$, modularization obviously benefits ontology re-use and explanation. As future work, we shall study the effectiveness of using modules to optimize axiom pinpointing, which is the cornerstone of explanation support.

Acknowledgement. The author would like to acknowledge Franz Baader and Carsten Lutz for their valuable suggestions and Christian H.-Wiener for his willingness in comparing the two modularization approaches.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proc. of IJCAI 2005, Morgan Kaufmann, San Francisco (2005)
2. Baader, F., Lutz, C., Suntisrivaraporn, B.: CEL—a polynomial-time reasoner for life science ontologies. In: Proc. of IJCAR 2006, Springer, Heidelberg (2006)
3. Baader, F., Lutz, C., Suntisrivaraporn, B.: Efficient reasoning in $\mathcal{EL}^+$. In: Prof. of DL (2006), J. of Logic, Language and Information (to appear)
4. Dowling, W.F., Gallier, J.: Linear-time algorithms for testing the satisfiability of propositional horn formulae. J. of Logic Programming 1(3), 267–284 (1984)
5. Cuenca Grau, B., Halaschek-Wiener, C., Kazakov, Y.: History matters: Incremental ontology reasoning using modules. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ISWC 2007. LNCS, vol. 4825, Springer, Heidelberg (2007)
6. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Just the right amount: Extracting modules from ontologies. In: Proc. of WWW 2007, ACM Press, New York (2007)
7. Haarslev, V., Möller, R.: Incremental query answering for implementing document retrieval services. In: Proc. of DL 2003 (2003)
8. Lutz, C., Wolter, F.: Conservative extensions in the lightweight description logic $\mathcal{EL}$. In: Pfenning, F. (ed.) CADE 2007. LNCS (LNAI), vol. 4603, Springer, Heidelberg (2007)
9. Noy, N., Musen, M.: The PROMPT suite: Interactive tools for ontology mapping and merging. International Journal of Human-Computer Studies (2003)
10. Parsia, B., Halaschek-Wiener, C., Sirin, E.: Towards incremental reasoning through updates in OWL-DL. In: Proc. of Reasoning on the Web Workshop (2006)
11. Seidenberg, J., Rector, A.: Web ontology segmentation: Analysis, classification and use. In: Proc. of WWW 2006, ACM Press, New York (2006)
12. Suntisrivaraporn, B.: Module extraction and incremental classification: A pragmatic approach for $\mathcal{EL}^+$ ontologies. LTCS-Report. TU Dresden, Germany (2007), see `http://lat.inf.tu-dresden.de/research/reports.html`
13. The systematized nomenclature of medicine, clinical terms (SNOMED CT). The International Health Terminology Standards Development Organisation (2007), `http://www.ihtsdo.org/our-standards/`