Noname manuscript No.
(will be inserted by the editor)

Incremental Classification of Description Logics Ontologies

Bernardo Cuenca Grau · Christian Halaschek-Wiener · Yevgeny Kazakov · Boontawee Suntisrivaraporn

Received: date / Accepted: date

Abstract The development of ontologies involves continuous but relatively small modifications. However, existing ontology reasoners do not take advantage of the similarities between different versions of an ontology. In this paper, we propose a collection of techniques for incremental reasoning — that is, reasoning that reuses information obtained from previous versions of an ontology. We have applied our results to incremental classification of OWL ontologies and found significant improvement over regular classification time on a set of real-world ontologies.

1 Introduction

Ontologies — formal conceptualizations of a domain of interest — have become increasingly important in computer science. The most popular ontology modeling languages are the Web Ontology Language (OWL) [33, 21] and its revision OWL 2 [29, 11], which are World Wide Web Consortium (W3C) standards. OWL ontologies are already being used in domains as diverse as bio-medicine, geography, astronomy, and defense. Prominent examples of bio-medical ontologies are SNOMED, the NCI Thesaurus, and GALEN.

The ontology languages OWL DL — the most expressive decidable variant of OWL — and OWL 2 are strongly related to *description logics* (DLs) [5] — a family of logic-based knowledge representation formalims with well-understood computational properties. The developers of ontologies have recognized the benefits of using DLs for ontology modeling; in particular, DLs provide unambiguous semantics to the

This is an extended version of the papers published by the authors at ISWC-07 [7] and ESWC-08 [39]

Bernardo Cuenca Grau and Yevgeny Kazakov
Computing Laboratory, University of Oxford, UK.

Christian Halaschek-Wiener
Clados Management LLC, San Mateo, CA, USA.

Boontawee Suntisrivaraporn
ICT, Sirindhorn International Institute of Technology, Thammasat University, Thailand.

modeling constructs available in OWL DL and OWL 2. These semantics make it possible to formalize and design algorithms for a number of *reasoning services*, which are critical to the development of large ontologies. For example, ontology classification involves organizing the concepts in an ontology into a subsumption hierarchy and allows for the detection of potential modeling errors, which typically manifest themselves as unintended subsumption relationships. There is currently a significant number of reasoners that support classification of ontologies written in OWL DL, OWL 2, or in some of their fragments. Prominent examples are CEL [3], FaCT++ [41], HermiT [30], KAON2 [22], Pellet [37], and RACER [17].

For developing and maintaining an ontology, it is important to detect possible errors as soon as possible. To this end, the ontology should be classified quite often and thus real time response from the reasoner becomes an important issue. If the response of the reasoner is too slow, ontology engineers may end up not using the reasoner as often as they would wish. As a consequence, a considerable amount of research effort has been devoted to make ontology classification feasible in practice. The main outcome of this line of research has been twofold. First, the development of a number of reasoning algorithms and optimization techniques for classifying ontologies; second, the identification of fragments of OWL DL and OWL 2 for which classification can be performed in polynomial time, such as the $\mathcal{EL}$ family of description logics [1]. The logics of the $\mathcal{EL}$ family are especially interesting since they can express many real-world ontologies, including SNOMED. Despite these achievements, classification of large and complex ontologies may still require a considerable amount of time, hence making the use of reasoners for ontology development sometimes impractical.

An important limitation of current reasoners is that they do not take advantage of the similarities between an ontology and its previous version. That is, when reasoning over the latest version of an ontology, current reasoners repeat the whole reasoning process from scratch. This is often unnecessary given that the development of ontologies typically involves continuous but relatively small modifications and therefore an ontology and its previous version usually share most of their axioms.

In this paper, we propose a collection of techniques for incremental ontology reasoning; that is, reasoning that reuses the results obtained from previous computations. The first technique we propose is based on the notion of a *module* [10, 8, 26] and can be applied to the incremental classification of OWL 2 ontologies. This technique does not depend on a particular reasoner or reasoning method and can be implemented using any existing reasoner. Our second proposed technique is specific to the reasoning algorithm used in the CEL system for the DL $\mathcal{EL}^+$ [4], and its implementation requires modifying the internals of the reasoner. Our empirical results using both techniques show substantial performance improvements over regular classification time.

2 Preliminaries

In Section 2.1, we introduce the description logics $\mathcal{SROIQ}$ [25] and $\mathcal{EL}^+$ [2], which provide the logical underpinning for OWL 2 [29, 11] and the EL profile of OWL 2

[28], respectively. In Section 2.2 we briefly present the classification algorithm for $\mathcal{EL}^+$, which is at the core of the CEL reasoner.

2.1 Description Logics

We assume fixed disjoint countably infinite sets $\mathbf{R}$ of *atomic roles* $(R, S, \ldots)$, $\mathbf{C}$ of *atomic concepts* $(A, B, \ldots)$ and $\mathbf{I}$ of *individuals* $(a, b, c, \ldots)$. The set of $\mathcal{SROIQ}$-*roles* is the set $\mathbf{Rol} = \{R,\ R^- \mid R \in \mathbf{R}\}$. We define a function $\mathsf{Inv}(\cdot) : \mathbf{Rol} \to \mathbf{Rol}$ as follows: $\mathsf{Inv}(R) = R^-$ and $\mathsf{Inv}(R^-) = R$.

A *strict partial order* $\prec$ on a set X is an irreflexive and transitive relation on X. A strict partial order $\prec$ on the set $\mathbf{Rol}$ of $\mathcal{SROIQ}$-roles is *regular* if $\prec$ additionally satisfies the following condition: $R_1 \prec R_2$ if and only if $R_1^- \prec R_2$ for all roles in $\mathbf{Rol}$. Let $\prec$ be a regular order on roles. A *role inclusion axiom* (RIA for short) is an expression of the form $w \sqsubseteq R$ where w is a finite string of roles called a *role chain* and R is an atomic role. A *role hierarchy* $\mathcal{R}_h$ is a finite set of RIAs. A RIA $w \sqsubseteq R$ is $\prec$-*regular* if R is an atomic role and any of the following conditions holds:

- $w = RR$, or
- $w = R^-$, or
- $w = S_1 \ldots S_n$ and $S_i \prec R$ for all $1 \leq i \leq n$, or
- $w = RS_1 \ldots S_n$ and $S_i \prec R$ for all $1 \leq i \leq n$, or
- $w = S_1 \ldots S_n R$ and $S_i \prec R$ for all $1 \leq i \leq n$.

The hierarchy $\mathcal{R}_h$ is *regular* if there exists a regular order $\prec$ such that each RIA in $\mathcal{R}_h$ is $\prec$-regular. Given a role hierarchy $\mathcal{R}_h$, we define the relation $\sqsubseteq^*$ to be the transitive-reflexive closure of $\sqsubseteq$ over $\{R \sqsubseteq S, \mathsf{Inv}(R) \sqsubseteq \mathsf{Inv}(S) \mid R \sqsubseteq S \in \mathcal{R}_h\}$. A role R is a *sub-role* (respectively *super-role*) of a role S if $R \sqsubseteq^* S$ (respectively $S \sqsubseteq^* R$). The set of roles that are *simple* in $\mathcal{R}_h$ is inductively defined as follows:

- an atomic role is simple if it does not occur on the right hand side of a RIA in $\mathcal{R}_h$,
- an inverse role R^- is simple if R is, and
- if R occurs on the right hand side of a RIA in $\mathcal{R}_h$, then R is simple if, for each $w \sqsubseteq R \in \mathcal{R}_h$, we have that $w = S$ for a simple role S.

If $\mathcal{R}_h$ is clear from the context, we often use "simple" instead of "simple in $\mathcal{R}_h$". An RBox $\mathcal{R}$ is composed of a regular role hierarchy and a finite set of role disjointness assertions of the form $\mathsf{Dis}(S_1, S_2)$, where S_1 and S_2 are simple roles.

The set $\mathbf{Con}$ of $\mathcal{SROIQ}$-*concepts* is defined by the following grammar:

$$\mathbf{Con} ::= \top \mid \{a\} \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid \exists R.C \mid \exists S.\mathsf{Self} \mid \geqslant n\, S.C$$

where $a \in \mathbf{I}$, $A \in \mathbf{C}$, $C_{(i)} \in \mathbf{Con}$, $R, S \in \mathbf{Rol}$, with S a simple role, and n a positive integer.

A *general concept inclusion axiom* (GCI) is an expression of the form $C_1 \sqsubseteq C_2$ with $C_i \in \mathbf{Con}$. A TBox $\mathcal{T}$ is a finite set of GCIs. An *individual assertion* is an expression of the form $C(a)$ or $R(a, b)$ with $C \in \mathbf{Con}$, $a, b \in \mathbf{I}$, and $R \in \mathbf{Rol}$. An ABox $\mathcal{A}$ is a finite set of individual assertions. A $\mathcal{SROIQ}$ ontology is a tuple $\mathcal{O} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$, where $\mathcal{T}$ is a TBox, $\mathcal{R}$ is an RBox and $\mathcal{A}$ is an ABox.

	Ontology $\mathcal{O}^1$:
D1	$\mathsf{Cystic\_Fibrosis} \equiv \mathsf{Fibrosis} \sqcap \exists \mathsf{located\_In.Pancreas}$
D2	$\mathsf{Genetic\_Fibrosis} \equiv \mathsf{Fibrosis} \sqcap \exists \mathsf{has\_Origin.Genetic\_Origin}$
D3	$\mathsf{Pancreatic\_Fibrosis} \equiv \mathsf{Fibrosis} \sqcap \mathsf{Pancreatic\_Disorder}$
C1	$\mathsf{Genetic\_Fibrosis} \sqsubseteq \mathsf{Genetic\_Disorder}$
C2	$\mathsf{Pancreatic\_Disorder} \sqsubseteq \mathsf{Disorder} \sqcap \exists \mathsf{located\_In.Pancreas}$

Table 1 A Bio-medical DL Ontology $\mathcal{O}^1$

Interpretation of Roles and Concepts		
$(R^-)^{\mathcal{I}}$	$=$	$\{(y,x) \mid (x,y) \in R^{\mathcal{I}}\}$
$\top^{\mathcal{I}}$	$=$	$\triangle^{\mathcal{I}}$
$\{a\}^{\mathcal{I}}$	$=$	$\{a^{\mathcal{I}}\}$
$(\neg C)^{\mathcal{I}}$	$=$	$\triangle^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$(C \sqcap D)^{\mathcal{I}}$	$=$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$(\exists R.C)^{\mathcal{I}}$	$=$	$\{x \mid \exists y : (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
$(\exists S.\mathsf{Self})^{\mathcal{I}}$	$=$	$\{x \mid (x,x) \in S^{\mathcal{I}}\}$
$(\geqslant n\, S.C)^{\mathcal{I}}$	$=$	$\{x \mid \sharp\{y \mid (x,y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geq n\}$
Interpretation of Axioms		
$\mathcal{I} \models C \sqsubseteq D$	iff	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$\mathcal{I} \models w \sqsubseteq R$	iff	$w^{\mathcal{I}} \subseteq R^{\mathcal{I}}$
$\mathcal{I} \models \mathsf{Dis}(S_1, S_2)$	iff	$S_1^{\mathcal{I}} \cap S_2^{\mathcal{I}} = \emptyset$
$\mathcal{I} \models C(a)$	iff	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
$\mathcal{I} \models R(a,b)$	iff	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Note: $\sharp N$ is the number of elements in N.

Table 2 Model-Theoretic Semantics of $\mathcal{SROIQ}$ and $\mathcal{EL}^+$

Note that other constructors and axioms of $\mathcal{SROIQ}$ as originally defined in [25], such as the bottom concept ($\bot$), concept disjunction ($C \sqcup D$), universal restriction ($\forall R.C$), at-most cardinality restrictions ($\leqslant n\, S.C$), concept definitions ($A \equiv C$), negative role assertions ($\neg R(a,b)$), role (ir)reflexivity, role (a)symmetry, and role transitivity can be expressed using the given ones.

The ontology $\mathcal{O}^1$ in Table 1 is an example of a $\mathcal{SROIQ}$ ontology with an empty RBox and ABox and whose TBox contains the axioms D1-D3, C1, and C2 in the table.

An *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 each $R \in \mathbf{R}$ a binary relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, to each $A \in \mathbf{C}$ a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and to each $a \in \mathbf{I}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ of the interpretation domain. Given a role chain $w = R_1 \dots R_n$, we set $w^{\mathcal{I}} = R_1^{\mathcal{I}} \circ \dots \circ R_n^{\mathcal{I}}$, where $\circ$ denotes composition of binary relations. The interpretation function $\cdot^{\mathcal{I}}$ is extended to complex roles and concepts as shown in the upper part of Table 2.

The *satisfaction relation* $\mathcal{I} \models \alpha$ between an interpretation $\mathcal{I}$ and a $\mathcal{SROIQ}$-axiom α (in words, $\mathcal{I}$ *satisfies* α) is defined in the lower part of Table 2. An interpretation $\mathcal{I}$ is a *model* of $\mathcal{O} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ if $\mathcal{I}$ satisfies all the axioms in $\mathcal{T}$, $\mathcal{R}$ and $\mathcal{A}$. An ontology $\mathcal{O}$ is consistent if it has a model. An ontology $\mathcal{O}$ *implies* an axiom α (written $\mathcal{O} \models \alpha$) if $\mathcal{I} \models \alpha$ for every model $\mathcal{I}$ of $\mathcal{O}$. In this case, we say that α

Algorithm 1 Classification of $\mathcal{EL}^+$ Ontologies

Procedure $\mathsf{classify}(\mathcal{O}^1)$
Input: $\mathcal{O}^1$: $\mathcal{EL}^+$ ontology;
Output: $\sqsubseteq_1$: binary relation

1: $\mathcal{O}^1 := \mathsf{Norm}(\mathcal{O}^1)$
2: $\sqsubseteq_1 := \emptyset$
3: **repeat**
4: Apply rules from Table 4
5: **until** no more rules apply
6: **return** $\sqsubseteq_1$

is a logical consequence of $\mathcal{O}$. A concept A is subsumed by B in $\mathcal{O}$ if and only if $\mathcal{O} \models A \sqsubseteq B$. The identification of the subsumption relationships between all pairs of atomic concepts occurring in $\mathcal{O}$ plus $\top$ and $\bot$ is called *ontology classification*.

The description logic $\mathcal{EL}^+$ is defined as follows. An $\mathcal{EL}^+$ RBox $\mathcal{R}$ is a finite set of RIAs of the form $w \sqsubseteq R$ where w is a role chain and R is an atomic role. The set of $\mathcal{EL}^+$ concepts is given by the following grammar:

$$\mathbf{Con} ::= \top \mid A \mid C_1 \sqcap C_2 \mid \exists R.C$$

An $\mathcal{EL}^+$ TBox is a finite set of GCIs of the form $C \sqsubseteq D$, where C, D are $\mathcal{EL}^+$-concepts. An $\mathcal{EL}^+$ ABox is a finite set of assertions of the form $C(a)$ or $R(a, b)$. An $\mathcal{EL}^+$ ontology $\mathcal{O}$ is a tuple $\mathcal{O} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$. For example, the ontology $\mathcal{O}^1$ in Table 1 is also an $\mathcal{EL}^+$ ontology. To represent an $\mathcal{EL}^+$ axiom α (either a GCI or a RIA), we will sometimes use the notation $\alpha_L \sqsubseteq \alpha_R$, where α_L and α_R denote the left hand side and the right hand side of the implication respectively. Note that every $\mathcal{EL}^+$ ontology $\mathcal{O} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ is also a $\mathcal{SROIQ}$ ontology provided that $\mathcal{R}$ is regular.

A *signature* is any subset $\mathbf{S}$ of $\mathbf{R} \uplus \mathbf{C} \uplus \mathbf{I}$. The *signature of an axiom* α is the set $\mathsf{Sig}(\alpha)$ of atomic roles, atomic concepts, and individuals that occur in α. The *signature of an ontology* $\mathcal{O}$ is the set $\mathsf{Sig}(\mathcal{O})$ of symbols that occur in $\mathcal{O}$. For convenience, we will sometimes use $\mathsf{CN}(\mathcal{O})$ and $\mathsf{RN}(\mathcal{O})$ to denote the set of atomic concepts and atomic roles respectively in $\mathsf{Sig}(\mathcal{O})$; we also use the notation $\mathsf{CN}^\top(\mathcal{O})$ for $\mathsf{CN}(\mathcal{O}) \cup \{\top\}$, $\mathsf{CN}^\bot(\mathcal{O})$ for $\mathsf{CN}(\mathcal{O}) \cup \{\bot\}$, and $\mathsf{CN}^{\top\bot}(\mathcal{O})$ for $\mathsf{CN}(\mathcal{O}) \cup \{\top, \bot\}$.

For two interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, $\mathcal{J} = (\Delta^\mathcal{J}, \cdot^\mathcal{J})$, and a signature $\mathbf{S}$ we write $\mathcal{I}|_\mathbf{S} = \mathcal{J}|_\mathbf{S}$ if (i) $\Delta^\mathcal{I} = \Delta^\mathcal{J}$, and (ii) $R^\mathcal{I} = R^\mathcal{J}$, $A^\mathcal{I} = A^\mathcal{J}$, and $a^\mathcal{I} = a^\mathcal{J}$ for every atomic role $R \in \mathbf{S}$, atomic concept $A \in \mathbf{S}$, and individual $a \in \mathbf{S}$.

2.2 Reasoning in $\mathcal{EL}^+$

In this section, we briefly present a polynomial-time classification algorithm for $\mathcal{EL}^+$. We refer the interested reader to [1] for a more detailed description.

Algorithm 1 accepts an $\mathcal{EL}^+$ ontology $\mathcal{O}^1$ and produces subsumptions of the form $A \sqsubseteq_1 B$ and $A \sqsubseteq_1 \exists R.B$ where $A, B \in \mathsf{CN}^\top(\mathcal{O}^1)$ and $R \in \mathsf{RN}(\mathcal{O}^1)$ such that:

$$\mathcal{O}^1 \models A \sqsubseteq B \;\Leftrightarrow\; A \sqsubseteq_1 B \tag{1}$$

Axiom	**Ontology** $\mathsf{Norm}(\mathcal{O}^1)$	**Comes from**
Ax1	Cystic_Fibrosis $\sqsubseteq$ Fibrosis	D1
Ax2	Cystic_Fibrosis $\sqsubseteq \exists$located_In.Pancreas	D1
Ax3	Fibrosis $\sqcap$ Aux1 $\sqsubseteq$ Cystic_Fibrosis	D1
Ax4	$\exists$located_In.Pancreas $\sqsubseteq$ Aux1	D1
Ax5	Genetic_Fibrosis $\sqsubseteq$ Fibrosis	D2
Ax6	Genetic_Fibrosis $\sqsubseteq \exists$has_Origin.Genetic_Origin	D2
Ax7	Fibrosis $\sqcap$ Aux2 $\sqsubseteq$ Genetic_Fibrosis	D2
Ax8	$\exists$has_Origin.Genetic_Origin $\sqsubseteq$ Aux2	D2
Ax9	Pancreatic_Fibrosis $\sqsubseteq$ Fibrosis	D3
Ax10	Pancreatic_Fibrosis $\sqsubseteq$ Pancreatic_Disorder	D3
Ax11	Fibrosis $\sqcap$ Pancreatic_Disorder $\sqsubseteq$ Pancreatic_Fibrosis	D3
Ax12	Genetic_Fibrosis $\sqsubseteq$ Genetic_Disorder	C1
Ax13	Pancreatic_Disorder $\sqsubseteq$ Disorder	C2
Ax14	Pancreatic_Disorder $\sqsubseteq \exists$located_In.Pancreas	C2

Table 3 Normalization of the $\mathcal{EL}^+$ ontology $\mathcal{O}^1$ from Table 1

I1	$A \in \mathsf{CN}(\mathcal{O}^1) \vdash A \sqsubseteq_1 A$
I2	$A \in \mathsf{CN}(\mathcal{O}^1) \vdash A \sqsubseteq_1 \top$
CR1	$A \sqsubseteq_1 B, B \sqsubseteq C \in \mathcal{O}^1 \vdash A \sqsubseteq_1 C$
CR2	$A \sqsubseteq_1 B_1, A \sqsubseteq_1 B_2, B_1 \sqcap B_2 \sqsubseteq C \in \mathcal{O}^1 \vdash A \sqsubseteq_1 C$
CR3	$A \sqsubseteq_1 B, B \sqsubseteq \exists R.C \in \mathcal{O}^1 \vdash A \sqsubseteq_1 \exists R.C$
CR4	$A \sqsubseteq_1 \exists R.B, B \sqsubseteq_1 C, \exists R.C \sqsubseteq D \in \mathcal{O}^1 \vdash A \sqsubseteq_1 D$
CR5	$A \sqsubseteq_1 \exists R.B, R \sqsubseteq S \in \mathcal{O}^1 \vdash A \sqsubseteq_1 \exists S.B$
CR6	$A \sqsubseteq_1 \exists R.B, B \sqsubseteq_1 \exists S.C, RS \sqsubseteq T \in \mathcal{O}^1 \vdash A \sqsubseteq_1 \exists T.C$

Table 4 Completion rules for $\mathcal{EL}^+$

The classification algorithm works on $\mathcal{EL}^+$ ontologies expressed in a suitable normal form. We say that $\mathcal{O}^1$ is in normal form if all GCIs and RIAs in $\mathcal{O}^1$ are of one of the following forms, where $A, A_1, A_2, B \in \mathsf{CN}^\top(\mathcal{O}^1)$ and $R, S, T \in \mathsf{RN}(\mathcal{O}^1)$:

$$A \sqsubseteq B; \quad A_1 \sqcap A_2 \sqsubseteq B; \quad A \sqsubseteq \exists R.B; \quad \exists R.A \sqsubseteq B; \quad R \sqsubseteq S; \quad RS \sqsubseteq T$$

The normalization (see Line 1 in Algorithm 1) can be carried out in linear time and yields an ontology whose size is linear in the size of the original ontology [1]. Furthermore, the normalized ontology $\mathsf{Norm}(\mathcal{O}^1)$ is a conservative extension of the original ontology $\mathcal{O}^1$: every model of $\mathsf{Norm}(\mathcal{O}^1)$ is a model of $\mathcal{O}^1$ and every model of $\mathcal{O}^1$ can be extended to a model of $\mathsf{Norm}(\mathcal{O}^1)$ by appropriately choosing the interpretation of the new atomic concepts and roles.

Table 3 shows the result of normalizing the $\mathcal{EL}^+$ ontology from Table 1. The last column of Table 3 indicates the axiom in $\mathcal{O}^1$ from which the normalized axiom has been obtained. Note that the normalization process may involve the introduction of new atomic concepts and roles (e.g., the concepts Aux1 and Aux2 in Table 3).

After normalization, Algorithm 1 initializes the output relation $\sqsubseteq_1$ to the empty set and exhaustively applies the rules from Table 4. The application of these rules on the normalized ontology from Table 3 is given in Table 5, where we indicate the obtained subsumption relations between atomic concepts with a checkmark (✓).

		New subsumption in $\sqsubseteq_1$	Rule Applied
	S0	$A \sqsubseteq_1 A$ for each $A \in \mathsf{CN}(\mathcal{O})$	I1$[A]$
	S1	$A \sqsubseteq_1 \top$ for each $A \in \mathsf{CN}(\mathcal{O})$	I2$[A]$
✓	S2	Cystic_Fibrosis $\sqsubseteq_1$ Fibrosis	CR1[S0,Ax1]
	S3	Cystic_Fibrosis $\sqsubseteq_1$ ∃located_In.Pancreas	CR3[S0,Ax2]
✓	S4	Genetic_Fibrosis $\sqsubseteq_1$ Fibrosis	CR1[S0,Ax5]
	S5	Genetic_Fibrosis $\sqsubseteq_1$ ∃has_Origin.Genetic_Origin	CR3[S0,Ax6]
✓	S6	Pancreatic_Fibrosis $\sqsubseteq_1$ Fibrosis	CR1[S0,Ax9]
✓	S7	Pancreatic_Fibrosis $\sqsubseteq_1$ Pancreatic_Disorder	CR1[S0,Ax10]
✓	S8	Genetic_Fibrosis $\sqsubseteq_1$ Genetic_Disorder	CR1[S0,Ax12]
✓	S9	Pancreatic_Disorder $\sqsubseteq_1$ Disorder	CR1[S0,Ax13]
	S10	Pancreatic_Disorder $\sqsubseteq_1$ ∃located_In.Pancreas	CR3[S0,Ax14]
✓	S11	Pancreatic_Fibrosis $\sqsubseteq_1$ Disorder	CR1[S7,Ax13]
	S12	Pancreatic_Disorder $\sqsubseteq_1$ Aux1	CR4[S10,S0,Ax4]
	S13	Cystic_Fibrosis $\sqsubseteq_1$ Aux1	CR4[S3,S0,Ax4]
	S14	Pancreatic_Fibrosis $\sqsubseteq_1$ ∃located_In.Pancreas	CR5[S7,S0,Ax14]
	S15	Pancreatic_Fibrosis $\sqsubseteq_1$ Aux1	CR4[S14,S0,Ax4]
✓	S15	Pancreatic_Fibrosis $\sqsubseteq_1$ Cystic_Fibrosis	CR2[S6,S15,Ax3]
	S16	Genetic_Fibrosis $\sqsubseteq_1$ Aux2	CR4[S5,S0,Ax8]

Table 5 The saturation of the $\mathcal{EL}^+$ ontology $\mathsf{Norm}(\mathcal{O}^1)$ from Table 3 under the rules from Table 4

	Original Ontology $\mathcal{O}^1$:	**Modified Ontology $\mathcal{O}^2$:**
D1	Cystic_Fibrosis ≡ Fibrosis ⊓ ∃located_In.Pancreas	Cystic_Fibrosis ≡ Fibrosis ⊓ ∃located_In.Pancreas ⊓ **∃has_Origin.Genetic_Origin**
D2	Genetic_Fibrosis ≡ Fibrosis ⊓ ∃has_Origin.Genetic_Origin	Genetic_Fibrosis ≡ Fibrosis ⊓ ∃has_Origin.Genetic_Origin
D3	Pancreatic_Fibrosis ≡ Fibrosis ⊓ Pancreatic_Disorder	Pancreatic_Fibrosis ≡ Fibrosis ⊓ Pancreatic_Disorder
C1	Genetic_Fibrosis ⊑ Genetic_Disorder	Genetic_Fibrosis ⊑ Genetic_Disorder
C2	Pancreatic_Disorder ⊑ Disorder ⊓ ∃located_In.Pancreas	Pancreatic_Disorder ⊑ Disorder ⊓ ∃located_In.Pancreas
$\Delta\mathcal{O} = \mathbf{diff}(\mathcal{O}^1, \mathcal{O}^2) = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$		
$\Delta^-\mathcal{O} =$	Cystic_Fibrosis ≡ Fibrosis ⊓ ∃located_In.Pancreas	
$\Delta^+\mathcal{O} =$	Cystic_Fibrosis ≡ Fibrosis ⊓ ∃located_In.Pancreas ⊓ **∃has_Origin.Genetic_Origin**	

Table 6 Evolution of a Bio-Medical Ontology

We conclude this section by pointing out that the classification algorithm implemented in CEL is an optimized version of the one described here. We refer the interested reader to [2,4] for a detailed description of this optimized algorithm.

3 The Challenge for Incremental Reasoning in Ontologies

Consider the medical ontology $\mathcal{O}^1$ given in Table 1, which consists of three concept definitions D1 – D3 and two concept inclusion axioms C1, C2. Suppose that an ontology engineer developing this ontology notices that the definition D1 for the concept Cystic_Fibrosis is incomplete and reformulates it by adding the new conjunct

α	Axiom:	$\mathcal{O}^1 \models^? \alpha$,	follows from:	$\mathcal{O}^2 \models^? \alpha$,	follows from:
α_1	Pancreatic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	Yes	D3, C2, **D1**	No	—
α_2	Cystic_Fibrosis $\sqsubseteq$ Genetic_Disorder	No	—	Yes	**D1**, D2, C1
α_3	Pancreatic_Fibrosis $\sqsubseteq$ Disorder	Yes	D3, C2	Yes	D3, C2
α_4	Genetic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	No	—	No	—

Table 7 Subsumption Relations Before and After the Change

$\exists$has_Origin.Genetic_Origin. As a result, a new version $\mathcal{O}^2$ of the ontology is obtained, as shown in Table 6. In order to ensure that no errors have been introduced by this change, the ontology engineer uses a reasoner to classify the new ontology $\mathcal{O}^2$.

Table 7 shows some subsumption relationships between atomic concepts in $\mathcal{O}^1$ and $\mathcal{O}^2$, which should be computed for classification. We can see that some of these subsumption relations have changed as a result of the modification to the ontology: subsumption α_1 follows from axioms D3, C2 and D1 in $\mathcal{O}^1$, but does not follow from $\mathcal{O}^2$ anymore since D1 has been modified; in contrast, the subsumption α_2, which did not follow from $\mathcal{O}^1$, is now a consequence of the modified D1, D2 and C1 in $\mathcal{O}^2$. Other subsumptions such as α_3 and α_4 did not change: α_3 is a consequence of axioms D3 and C2 which have not been modified; α_4 follows neither from $\mathcal{O}^1$ nor from $\mathcal{O}^2$.

It is reasonable to expect that small changes in ontologies will not affect many subsumption relations. That is, the number of subsumptions that change their entailment status w.r.t. the ontology (e.g., α_1 or α_2 in Table 7) is often small compared to the number of subsumptions that do not (e.g., α_3 or α_4). If so, many (possibly expensive) re-computations can be avoided by reusing the subsumption relations computed for the previous version of the ontology. In order to realize this idea, one has to identify which subsumptions could be affected by a change and which cannot.

Suppose we know that a subsumption α holds in $\mathcal{O}^1$. Then we can guarantee that α still holds in $\mathcal{O}^2$ provided the axioms from which α follows in $\mathcal{O}^1$ have not been modified. For example, in Table 7, the subsumption α_3 is a consequence of axioms D3 and C2, both of which have not been modified in $\mathcal{O}^2$. Hence, we can conclude that α_3 holds in $\mathcal{O}^2$ without performing reasoning over $\mathcal{O}^2$. In contrast, this test is not applicable to α_1, since α_1 is a consequence of axioms D3, C2 and D1 in $\mathcal{O}^1$, and D1 has been modified in $\mathcal{O}^2$. In this case, the status of α_1 in $\mathcal{O}^2$ has to be computed by other means (e.g., using a reasoner). Thus, the status of every subsumption relation α that holds in $\mathcal{O}^1$ requires re-computation for $\mathcal{O}^2$ only if in every *justification* for α (i.e., every minimal subset of $\mathcal{O}^1$ which implies α) some axiom has been modified. This approach is reminiscent of the way Truth Maintenance Systems (TMS) maintain logical dependencies between axioms [16, 12]. The notion of a justification for an axiom has also been used for pinpointing the axioms responsible for errors in ontologies, such as unsatisfiable concepts and unintended subsumptions [35, 32].

The situation is principally different in the case of subsumptions α that do *not* hold in $\mathcal{O}^1$. In this case, if to follow the previous approach, one has to keep track of "evidences" for *non-entailments* of subsumptions in ontologies and verify if at least one such "evidences" for α in $\mathcal{O}^1$ can be reused in $\mathcal{O}^2$. Here, the "evidence" might be, for example, a (part of a) counter-model for α in $\mathcal{O}^1$ that is constructed by tableau-

based procedures. Such techniques based on *model caching* have been recently proposed in the context of incremental reasoning [19]. These techniques apply, however, only to additions and deletions of ABox assertions since changes in general axioms often require considerable modifications of the models. Moreover, such techniques require close interaction with the model construction routine of the tableau reasoner, which precludes their use in arbitrary "off-the-shelf" reasoners without considerable modifications. In particular, these techniques cannot be directly used in reasoners like KAON2, or CEL, which are not tableaux-based.

We stress that the main challenge for incremental ontology reasoning is to maintain non-subsumptions since, in typical ontologies, more than 99% of subsumption relations between atomic concepts do not hold. In other words, the case of axiom α_4 in Table 7 is likely to be the most common one after a change in an ontology. This will be confirmed by our experimental results in Section 6.

4 Incremental Ontology Classification Using Modules

In this section, we present a technique for incremental reasoning in ontologies using a notion of a *module* for an axiom. Our technique can be used to keep track of "evidences" for both subsumptions and non-subsumptions modulo arbitrary changes in ontologies and works in combination with any DL reasoner that provides for standard reasoning services. The DL reasoner is used as a "black-box" to answer subsumption queries, in the sense that the subsumption algorithm implemented by the reasoner does not need to be modified. As a consequence, our technique is very flexible and easy to implement.

In the remainder of this paper, we adopt the following notational conventions for naming ontologies: we use numbers in subscripts ($\mathcal{O}_1$, $\mathcal{O}_2$, ...) to denote subsets of a given ontology (typically modules) and superscripts ($\mathcal{O}^1$, $\mathcal{O}^2$, ...) to denote different ontologies (in particular, different ontology versions).

4.1 Modules for Axioms as Evidences for Entailments and Non-entailments

Definition 1 (Module for an Axiom) Let $\mathcal{O}$ be an ontology and $\mathcal{O}_1 \subseteq \mathcal{O}$ a (possibly empty) subset of axioms in $\mathcal{O}$. We say that $\mathcal{O}_1$ is a *module for an axiom* α in $\mathcal{O}$ (or an α*-module in* $\mathcal{O}$) if the following condition holds: $\mathcal{O}_1 \models \alpha$ iff $\mathcal{O} \models \alpha$.

Intuitively, a module for an axiom α in $\mathcal{O}$ is a subset $\mathcal{O}_1$ of $\mathcal{O}$ which contains the axioms that are "relevant" to the entailment of α, in the sense that $\mathcal{O}$ implies α if and only if $\mathcal{O}_1$ implies α. The module, however, can also contain "non-relevant" axioms: if $\mathcal{O}_1$ is a module for α in $\mathcal{O}$, then each superset of $\mathcal{O}_1$ in $\mathcal{O}$ is also a module for α. In particular, the whole ontology $\mathcal{O}$ is always a module in $\mathcal{O}$ for every α. Note that the "only if" direction of Definition 1 always holds; that is, $\mathcal{O}_1 \models \alpha$ implies $\mathcal{O} \models \alpha$, since the entailment relation $\models$ is monotonic for any standard DL.

If $\mathcal{O}$ implies α, then every module $\mathcal{O}_1$ for α should contain at least one *justification* — that is, a minimal set of axioms which imply α [36, 23]. Conversely, every justification for α is also a *minimal module* for α — that is, a module containing no

other module as a proper subset. In case $\mathcal{O}$ does not imply α (i.e., there are no justifications for α in $\mathcal{O}$), $\mathcal{O}_1$ can be any subset of $\mathcal{O}$. Hence, knowing all the justifications for α in $\mathcal{O}$ is sufficient for identifying all modules for α in $\mathcal{O}$.

The following proposition, which is a simple consequence of Definition 1, provides the main property underlying incremental ontology reasoning using modules:

Proposition 1 *Let $\mathcal{O}^1$, $\mathcal{O}^2$ be ontologies, α an axiom, and $\mathcal{O}^1_\alpha$, $\mathcal{O}^2_\alpha$ respectively modules for α in $\mathcal{O}^1$ and $\mathcal{O}^2$. Then, the following properties hold:*

1. *If $\mathcal{O}^1 \models \alpha$ and $\mathcal{O}^1_\alpha \subseteq \mathcal{O}^2$, then $\mathcal{O}^2 \models \alpha$.*
2. *If $\mathcal{O}^1 \not\models \alpha$ and $\mathcal{O}^2_\alpha \subseteq \mathcal{O}^1$, then $\mathcal{O}^2 \not\models \alpha$.*

Proof 1. Assume that $\mathcal{O}^1 \models \alpha$ and $\mathcal{O}^1_\alpha \subseteq \mathcal{O}^2$. Since $\mathcal{O}^1 \models \alpha$ and $\mathcal{O}^1_\alpha$ is a module for α in $\mathcal{O}^1$, by Definition 1 we have $\mathcal{O}^1_\alpha \models \alpha$. Since $\mathcal{O}^1_\alpha \subseteq \mathcal{O}^2$ and $\mathcal{O}^1_\alpha \models \alpha$, by monotonicity we obtain $\mathcal{O}^2 \models \alpha$, as required.

2. By Point 1 of this proposition (when swapping indexes 1 and 2), we have $\mathcal{O}^2 \models \alpha$ and $\mathcal{O}^2_\alpha \subseteq \mathcal{O}^1$ implies $\mathcal{O}^1 \models \alpha$. Hence, it is not possible to have $\mathcal{O}^1 \not\models \alpha$, $\mathcal{O}^2_\alpha \subseteq \mathcal{O}^1$, and $\mathcal{O}^2 \models \alpha$ at the same time, which implies Point 2. □

Proposition 1 provides two sufficient conditions for reusing in $\mathcal{O}^2$ the entailment status of the axiom α in $\mathcal{O}^1$. Point 1 of the proposition ensures that, in the case when the entailment $\mathcal{O}^1 \models \alpha$ holds, it is always correct to assume that $\mathcal{O}^2 \models \alpha$ provided all the axioms in some module $\mathcal{O}^1_\alpha$ for α in $\mathcal{O}^1$ still occur in $\mathcal{O}^2$. In this case, $\mathcal{O}^1_\alpha$ is an evidence module for the entailment α in $\mathcal{O}^2$. This solution is not principally different from the one outlined in Section 3. Note that $\mathcal{O}^1_\alpha$ is also a module for α in $\mathcal{O}^2$ according to Definition 1.

A more interesting situation is when the entailment $\mathcal{O}^1 \models \alpha$ does not hold. According to Point 2 of Proposition 1, it is always correct to assume that this entailment does not hold in $\mathcal{O}^2$ as well, provided that all the axioms in some module $\mathcal{O}^2_\alpha$ for α in $\mathcal{O}^2$ also occur in $\mathcal{O}^1$. In this case, $\mathcal{O}^2_\alpha$ is an evidence module for the non-entailment of α in $\mathcal{O}^2$. Note that in this case $\mathcal{O}^2_\alpha$, as well as any subset of $\mathcal{O}^1$, is a module for α in $\mathcal{O}^1$ since $\mathcal{O}^1 \not\models \alpha$.

Using Proposition 1, one can outline a basic incremental algorithm for checking the entailment of an axiom α with respect to a changing ontology. In order to check if the entailment status of α in $\mathcal{O}^1$ can be reused for $\mathcal{O}^2$, it is sufficient to compute, depending on whether $\mathcal{O}^1 \models \alpha$ or $\mathcal{O}^1 \not\models \alpha$, a module $\mathcal{O}^1_\alpha$ for α in $\mathcal{O}^1$, or a module $\mathcal{O}^2_\alpha$ for α in $\mathcal{O}^2$ respectively. If the change does not involve any of the axioms in the module, then the status of the entailment of α also does not change. The converse of this is not necessarily true: the status of α could remain unchanged even if the corresponding module has been modified. For example, it is easy to see that every module for the axiom $\alpha = (\mathsf{Cystic\_Fibrosis} \sqsubseteq \mathsf{Fibrosis})$ in ontologies $\mathcal{O}^1$ and $\mathcal{O}^2$ in Table 6 contains the respective axiom D1, since α follows from both $\mathcal{O}^1$ and $\mathcal{O}^2$, but does not follow if the respective axiom is removed. Thus, the status of the entailment of α remained the same during the change $\mathcal{O}^1 \Rightarrow \mathcal{O}^2$ despite the fact that the axiom D1 has been modified. If an axiom in the evidence module has been modified during the change $\mathcal{O}^1 \Rightarrow \mathcal{O}^2$, the sufficient condition does not apply, and thus the status of α w.r.t. $\mathcal{O}^2$ should be verified using the reasoner. The use of modules, however, is also valuable in this situation: instead of checking if α follows from $\mathcal{O}^2$, one could

equivalently check if α follows from any module $\mathcal{O}^2_\alpha$ for α in $\mathcal{O}^2$ which could be an easy task if the module is small.

To summarize, the use of modules provides two compelling advantages for incremental reasoning in ontologies: first, the computation of a given query may be avoided and the answer can simply be reused from the "cache" of a previous computation; second, even if the query needs to be recomputed, the use of modules allows for filtering out irrelevant axioms and reducing the search space.

4.2 Locality-Based Modules—Small Modules which are Fast to Compute

In the previous section, we have outlined the use modules for incremental ontology reasoning, but we did not discuss how these modules can be computed. In this section, we discuss algorithmic aspects and trade-offs for the computations of modules.

As pointed out in the previous section, a module for an axiom in an ontology is not necessarily unique. Clearly, the choice of evidence modules $\mathcal{O}^1_\alpha$ and $\mathcal{O}^2_\alpha$ has a direct impact on the quality of the incremental entailment test for α. If one chooses the whole ontology $\mathcal{O}^1$ or $\mathcal{O}^2$ as evidence module (recall that the whole ontology is always a module in itself), then it is more likely that at least one axiom in the module has been modified and therefore the sufficient condition from Proposition 1 is not applicable. Thus, choosing an evidence module that is as small as possible is advantageous for incremental ontology reasoning. However, smaller modules are generally harder to compute. In the extreme case, computing any *minimal* module for α in $\mathcal{O}$ is at least as hard as to check whether $\mathcal{O}$ implies α — the original problem we have started with. Indeed, it is easy to see from Definition 1 that the minimal module is non-empty if and only if α is implied by $\mathcal{O}$ provided α is not a tautology. Thus, there is a trade-off between the complexity of computing a module on the one hand, and its usefulness for incremental reasoning on the other hand — the smaller the module, the more useful it is, but the harder it is to compute. In the rest of this section, we define several kinds of modules that are algorithmically easy to compute and at the same time are reasonably small for typical ontologies.

We start by pointing out a connection between the notion of a module for an axiom and a well known notion of model-conservative extension (see, e.g. [26]).

Definition 2 (Model Conservative Extension) Let $\mathcal{O}$ be an ontology and $\mathbf{S}$ a signature. We say that $\mathcal{O}$ is a *model* $\mathbf{S}$*-conservative extension* of $\mathcal{O}_1 \subseteq \mathcal{O}$, if for every model $\mathcal{I}$ of $\mathcal{O}_1$, there exists a model $\mathcal{J}$ of $\mathcal{O}$ such that $\mathcal{I}|_{\mathbf{S}} = \mathcal{J}|_{\mathbf{S}}$.

Proposition 2 *If $\mathcal{O}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{O}_1 \subseteq \mathcal{O}$, then $\mathcal{O}_1$ is a module in $\mathcal{O}$ for every axiom α such that* $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$.

Proof Assume that $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}_1 \subseteq \mathcal{O}$ and $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$. We have to prove that $\mathcal{O}_1 \models \alpha$ iff $\mathcal{O} \models \alpha$.

As pointed out after Definition 1, the "only if" direction of the implication is trivial. For proving the "if" direction, assume that $\mathcal{O}_1 \not\models \alpha$. We prove that $\mathcal{O} \not\models \alpha$. Since $\mathcal{O}_1 \not\models \alpha$, there exists a model $\mathcal{I}$ of $\mathcal{O}_1$ such that $\mathcal{I} \not\models \alpha$. Since $\mathcal{O}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{O}_1$ and $\mathcal{I}$ is a model of $\mathcal{O}_1$, by Definition 2, there exists

a model $\mathcal{J}$ of $\mathcal{O}$ such that $\mathcal{I}|_{\mathbf{S}} = \mathcal{J}|_{\mathbf{S}}$. In particular, $\mathcal{J} \not\models \alpha$, since $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$ and $\mathcal{I} \not\models \alpha$. Hence $\mathcal{O} \not\models \alpha$, as required. □

Proposition 2 implies that, in order to extract a module for α in $\mathcal{O}$, it is sufficient to find a subset $\mathcal{O}_1$ of $\mathcal{O}$ such that $\mathcal{O}$ is a model conservative extension of $\mathcal{O}_1$ for $\mathbf{S} = \mathsf{Sig}(\alpha)$. Unfortunately, this observation cannot be directly used for finding modules since the problem of checking whether an ontology $\mathcal{O}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{O}_1$ is undecidable already for $\mathcal{EL}$ [27]; for $\mathcal{ALC}$ it is even not semi-decidable [26]. In [9, 8, 10] the following *sufficient condition* for model conservativity was shown to work well in typical ontologies.

Definition 3 (Semantic Locality [9]) Let $\mathbf{S}$ be a signature. We say that an interpretation $\mathcal{I}$ is *local for* $\mathbf{S}$ if for every atomic concept $A \notin \mathbf{S}$ and every atomic role $R \notin \mathbf{S}$, we have $A^{\mathcal{I}} = R^{\mathcal{I}} = \emptyset$. A $\mathcal{SROIQ}$ axiom α is *semantically local for a signature* $\mathbf{S}$ if $\mathcal{I} \models \alpha$ for every $\mathcal{I}$ that is local for $\mathbf{S}$. A $\mathcal{SROIQ}$ ontology $\mathcal{O}$ is *local for* $\mathbf{S}$ if every axiom in $\mathcal{O}$ is local for $\mathbf{S}$.

Intuitively, an axiom α (or ontology $\mathcal{O}$) is semantically local for $\mathbf{S}$ if each interpretation of the symbols in $\mathbf{S}$ can be extended to a model of α (respectively $\mathcal{O}$) by interpreting the remaining symbols as the empty set. Note that if α is semantically local for $\mathbf{S}$, then it is also semantically local for any subset of $\mathbf{S}$. For example, the axiom D2 from Table 6 is semantically local w.r.t. $\mathbf{S} = \{\mathsf{Fibrosis}, \mathsf{has\_Origin}\}$: if we interpret the remaining symbols in this axiom as the empty set, we obtain a model of the axiom, regardless of the interpretation of the symbols in $\mathbf{S}$.

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

If an ontology $\mathcal{O}$ can be partitioned as $\mathcal{O} = \mathcal{O}_1 \cup \mathcal{O}_s$ such that $\mathcal{O}_s$ is semantically local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$, then it is easy to see that $\mathcal{O}$ is a model $\mathbf{S}$-conservative extension of $\mathcal{O}_1$ since every model of $\mathcal{O}_1$ can be extended to a model of $\mathcal{O}_s$ (and hence of $\mathcal{O}$) by interpreting the symbols that do not occur in $\mathcal{O}_1$ and $\mathbf{S}$ as the empty set. By combining this observation with Proposition 2, we obtain the following notion.

Definition 4 (Semantic Locality-based Module [8]) Let $\mathcal{O}$ be a $\mathcal{SROIQ}$-ontology and $\mathbf{S}$ a signature. An ontology $\mathcal{O}_1 \subseteq \mathcal{O}$ is a *semantic locality-based* $\mathbf{S}$*-module* in $\mathcal{O}$ if $\mathcal{O} \setminus \mathcal{O}_1$ is semantically local w.r.t. $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$. In addition, if for every locality based $\mathbf{S}$-module $\mathcal{O}_2$ in $\mathcal{O}$ we have $\mathcal{O}_1 \subseteq \mathcal{O}_2$, then $\mathcal{O}_1$ is the *smallest* semantic locality-based $\mathbf{S}$-module in $\mathcal{O}$.

It follows from Definition 4 using Proposition 2 that any semantic locality-based module for $\mathbf{S} = \mathsf{Sig}(\alpha)$ in $\mathcal{O}$ is a module for α in $\mathcal{O}$ according to Definition 1. Given an ontology $\mathcal{O}$ and a signature $\mathbf{S}$, the smallest locality-based $\mathbf{S}$-module in $\mathcal{O}$ can be extracted using Algorithm 2 (see [8, 10] for more details and correctness proofs). The algorithm works by first initializing $\mathcal{O}_1$ to the empty set and then repeatedly moving all axioms α that are not semantically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$ to $\mathcal{O}_1$. The ontology $\mathcal{O}_2$ is used to keep track of those axioms from $\mathcal{O} \setminus \mathcal{O}_1$ that have not been checked for

Algorithm 2 Module Extraction Algorithm for $\mathcal{SROIQ}$

Procedure extract_module($\mathcal{O}, \mathbf{S}$)
Input:
$\mathcal{O}$: ontology;
$\mathbf{S}$: signature;
Output:
$\mathcal{O}_1$: a module for $\mathbf{S}$ in $\mathcal{O}$

```
1: O1 := ∅   O2 := O
2: while not empty(O2) do
3:     α := select_axiom(O2)
4:     if local( α, S ∪ Sig(O1) ) then
5:         O2 := O2 \ {α}
6:     else
7:         O1 := O1 ∪ {α}
8:         O2 := O \ O1
9:     end if
10: end while
11: return O1
```

	$\mathcal{O}_1$	$\mathcal{O}_2$	New $A \in \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$	α	loc?
1	–	D1, D2, D3, C1, C2	Pancreatic_Fibrosis	D3	No
2	D3	D1, D2, C1, C2	Fibrosis, Pancreatic_Disorder	D1	Yes
3	D3	D2, C1, C2	–	D2	Yes
4	D3	C1, C2	–	C1	Yes
5	D3	C2	–	C2	No
6	D3, C2	D1, D2, C1,	Disorder, located_In, Pancreas	D1	Yes
7	D3, C2	D2, C1	–	D2	Yes
8	D3, C2	C1	–	C1	Yes
9	D3, C2	–	–	–	

Table 8 A sample trace for the Algorithm 2 for $\mathcal{O} = \mathcal{O}^1$ from Table 6 and $\mathbf{S} = \{\mathsf{Pancreatic\_Fibrosis}\}$

locality. In Table 8 we provide an example trace of Algorithm 2 for the input ontology $\mathcal{O}_1$ in Table 6 and signature $\mathbf{S} = \{\mathsf{Pancreatic\_Fibrosis}\}$.

Algorithm 2 internally uses a subroutine local$(\alpha, \mathbf{S})$ for checking whether an axiom α is semantically local for $\mathbf{S}$. We argue that testing semantic locality should be easier than checking the entailment w.r.t. the ontology. First, from the computational complexity point of view, checking locality for $\mathcal{SROIQ}$-axioms is PSPACE-complete (this can be reduced to checking if an $\mathcal{ALCIQ}$-axiom is a tautology [8, 10]), whereas, as recently shown, checking entailment in $\mathcal{SROIQ}$ is N2EXPTIME-complete [24]. Second, the size of a single axiom in typical ontologies is much smaller than the size of the ontology. Therefore, performing several locality tests should be easier than checking entailment w.r.t. the whole ontology. In the cases when checking for semantic locality may seem too costly, one can instead use the following sufficient syntactic condition for local$(\alpha, \mathbf{S})$.

Definition 5 (Syntactic Locality for $\mathcal{SROIQ}$) Let $\mathbf{S}$ be a signature. The following grammar recursively defines two sets of concepts $\mathbf{Con}^{\emptyset}(\mathbf{S})$ and $\mathbf{Con}^{\Delta}(\mathbf{S})$ for $\mathbf{S}$:

$$\begin{aligned}\mathbf{Con}^{\emptyset}(\mathbf{S}) ::= {} & A^{\emptyset} \mid (\neg C^{\Delta}) \mid (C^{\emptyset} \sqcap C) \mid (C \sqcap C^{\emptyset}) \\ & \mid (\exists R^{\emptyset}.C) \mid (\exists R.C^{\emptyset}) \mid (\geqslant n\, S^{\emptyset}.C) \mid (\geqslant n\, S.C^{\emptyset}) \mid \exists S^{\emptyset}.\mathsf{Self}\,. \\ \mathbf{Con}^{\Delta}(\mathbf{S}) ::= {} & \top \mid (\neg C^{\emptyset}) \mid (C_1^{\Delta} \sqcap C_2^{\Delta})\,.\end{aligned}$$

In the grammar, we have that $A^{\emptyset} \notin \mathbf{S}$ is an atomic concept, $R^{\emptyset}$ (resp. $S^{\emptyset}$) is either an atomic role (resp. a simple atomic role) not in $\mathbf{S}$ or the inverse of an atomic role (resp. of a simple atomic role) not in $\mathbf{S}$, C is any concept, R is any role, S is any simple role, and $C^{\emptyset} \in \mathbf{Con}^{\emptyset}(\mathbf{S})$, $C_1^{\Delta}, C_2^{\Delta} \in \mathbf{Con}^{\Delta}(\mathbf{S})$. We also denote by $w^{\emptyset}$ a role chain $w = R_1 \ldots R_n$ such that for some i with $1 \leq i \leq n$, we have that R_i is (possibly inverse of) an atomic role not in $\mathbf{S}$. An axiom α is *syntactically local w.r.t.* $\mathbf{S}$ if it is of the form: (1) $w^{\emptyset} \sqsubseteq R$, or (2) $\mathsf{Dis}(S^{\emptyset}, S)$, or (3) $\mathsf{Dis}(S, S^{\emptyset})$, or (4) $C^{\emptyset} \sqsubseteq C$, or (5) $C \sqsubseteq C^{\Delta}$. An ontology $\mathcal{O}$ is syntactically local w.r.t. $\mathbf{S}$ if all the axioms in $\mathcal{O}$ are syntactically local w.r.t. $\mathbf{S}$.

Definition 5 is a straightforward extension to $\mathcal{SROIQ}$ of the definition introduced in [9] for $\mathcal{SHOIQ}$. Intuitively, the concepts in $\mathbf{Con}^{\emptyset}(\mathbf{S})$ (respectively in $\mathbf{Con}^{\Delta}(\mathbf{S})$) are those for which it can be inductively shown that they are interpreted as the empty set $\emptyset$ (respectively as $\Delta^{\mathcal{I}}$) in each interpretation $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ which interprets the atomic concepts and roles that are not in $\mathbf{S}$ as the empty set. It is immediate from the definition that testing for syntactic locality can be done in polynomial time. In [8] it has been shown that the version of Algorithm 2 where $\mathsf{local}(\alpha, \mathbf{S})$ checks for syntactic locality, returns the (unique) smallest *syntactic locality-based module* for $\mathbf{S}$ in $\mathcal{O}$ (the smallest subset $\mathcal{O}_1$ of $\mathcal{O}$ such that $\mathcal{O} \setminus \mathcal{O}_1$ is syntactically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$), regardless of the way in which the axioms in O_2 are selected.

Note that even though both versions of Algorithm 2 return the smallest (semantic or syntactic) locality-based modules for $\mathbf{S} = \mathsf{Sig}(\alpha)$, the result might still not be a minimal module for α due to the complexity reasons given in the beginning of this section. In Section 6, we demonstrate empirically that (an optimized version of) Algorithm 2 works fast and computes small enough modules to be useful for incremental reasoning.

4.3 The Incremental Ontology Classification Procedure

As shown in the previous section, in order to extract a module for an axiom α in $\mathcal{O}$, it is sufficient to compute the smallest (syntactic / semantic) locality-based module for $\mathbf{S} = \mathsf{Sig}(\alpha)$ in $\mathcal{O}$. However, when α is a subsumption between atomic concepts $A, B \in \mathsf{CN}(\mathcal{O})$, it suffices to extract a module only for $\mathbf{S} = \{A\}$, as shown in [8].

Proposition 3 *Let $\mathcal{O}$ be a $\mathcal{SROIQ}$ ontology, $A, B \in \mathsf{CN}(\mathcal{O})$, and $\mathcal{O}_A$ the output of Algorithm 2 for input $\mathcal{O}$ and $\mathbf{S} = \{A\}$. Then $\mathcal{O}_A$ is a module in $\mathcal{O}$ for $\alpha = (A \sqsubseteq B)$.*

A practical implication of Proposition 3 is that in order to maintain incrementally the status of all subsumption relations between atomic concepts, it is sufficient to keep track of only the locality-based modules for singleton sets of atomic concepts.

α	Axiom $A \sqsubseteq B$:	$\mathcal{O}^1_A$	$\mathcal{O}^2_A$
α_1	Pancreatic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	D3,C2,**D1**	D3,C2
α_2	Cystic_Fibrosis $\sqsubseteq$ Genetic_Disorder	**D1**	**D1**,D2,C1
α_3	Pancreatic_Fibrosis $\sqsubseteq$ Disorder	D3,C2,**D1**	D3,C2
α_4	Genetic_Fibrosis $\sqsubseteq$ Cystic_Fibrosis	C1	C1

Table 9 Locality-based Modules for Subsumption in Ontologies from Table 6

A	$\mathcal{O}^1_A$	$\mathcal{O}^2_A$
Cystic_Fibrosis	**D1**	**D1**, D2, C1
Fibrosis	$\emptyset$	$\emptyset$
Pancreas	$\emptyset$	$\emptyset$
Genetic_Fibrosis	C1	C1
Genetic_Origin	$\emptyset$	$\emptyset$
Pancreatic_Fibrosis	D3, C2, **D1**	D3, C2
Pancreatic_Disorder	C2	C2
Genetic_Disorder	C1	C1
Disorder	C2	C2
$\top$	$\emptyset$	$\emptyset$

Table 10 Modules For Atomic Concepts in Ontologies from Table 6

In the following, by a locality-based module for an axiom $\alpha = (A \sqsubseteq B)$, we mean the syntactic or semantic locality-based module for $\mathbf{S} = \{A\}$.

Consider the ontologies $\mathcal{O}^1$ and $\mathcal{O}^2$ in Table 6 and the axioms α_1–α_4 in Table 7. Each of these axioms is of the form $\alpha = (A \sqsubseteq B)$, with $A, B \in \mathsf{CN}(\mathcal{O})$. Table 9 provides the locality-based modules for α_1–α_4 in $\mathcal{O}^1$ and $\mathcal{O}^2$ computed using Algorithm 2 (the algorithms based on semantic and syntactic locality produce the same results). Note that some modules are supersets of the actual minimal modules from Table 7; we have underlined the additional axioms. The modules for axioms α_1–α_3 have been changed, whereas the module for the axiom α_4 has remained unchanged. Hence, the sufficient test for preservation of (non)subsumptions using modules resulted in only one "false positive" for subsumption α_3, where the subsumption relation did not change, but the modules have been modified.

Table 10 provides the full picture of the modules and their changes for our example ontology from Table 6. The only modules that have been modified are those for $A = \mathsf{Cystic\_Fibrosis}$ and $A = \mathsf{Pancreatic\_Fibrosis}$; in the first case, axiom D1 has been modified; in the second case, axiom D1 has been removed. Applying Propositions 1 and 3, we can see that every subsumption that ceases to hold as a result of the change should be either of the form $\alpha = (\mathsf{Cystic\_Fibrosis} \sqsubseteq B)$ or $\alpha = (\mathsf{Pancreatic\_Fibrosis} \sqsubseteq B)$, whereas every new subsumption should be of the form $\alpha = (\mathsf{Cystic\_Fibrosis} \sqsubseteq B)$.

Algorithm 3 outlines an incremental classification procedure using locality-based modules. Given an ontology $\mathcal{O}^1$ and a change $\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$ consisting of the sets of removed and added axioms, the algorithm computes the subsumption partial order $\sqsubseteq_2$ for the resulting ontology $\mathcal{O}^2 = (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$ by reusing the subsumption relation $\sqsubseteq_1$ already computed for $\mathcal{O}^1$. In order to perform this operation,

Algorithm 3 Incremental Classification Using Modules

Procedure inc_classify_mod($\mathcal{O}^1, \Delta^+\mathcal{O}, \sqsubseteq_1$)
Input:
$\mathcal{O}^1$: an ontology
$\Delta\mathcal{O} = (\Delta^-\mathcal{O}, \Delta^+\mathcal{O})$: removed / added axioms
$\sqsubseteq_1$: subsumption relations in $\mathcal{O}^1$
$A \to \mathcal{O}^1_A$: a module for every $A \in \mathsf{CN}^\top(\mathcal{O}^1)$
Output:
$\mathcal{O}^2$: the result of applying the change $\Delta\mathcal{O}$ to $\mathcal{O}^1$
$\sqsubseteq_2$: subsumption relations in $\mathcal{O}^2$
$A \to \mathcal{O}^2_A$: a module for every $A \in \mathsf{CN}^\top(\mathcal{O}^2)$

1: $\mathcal{O}^2 := (\mathcal{O}^1 \setminus \Delta^-\mathcal{O}) \cup \Delta^+\mathcal{O}$
2: **for each** $D \in \mathsf{CN}(\mathcal{O}^2) \setminus \mathsf{CN}(\mathcal{O}^1)$ **do**
3: $\quad \mathcal{O}^1_D := \mathcal{O}^1_\top$
4: $\quad$ **for each** $\top \sqsubseteq_1 B$ **do** $D \sqsubseteq_1 B := \mathbf{true}$
5: $\quad$ **for each** $A \sqsubseteq_1 \bot$ **do** $A \sqsubseteq_1 D := \mathbf{true}$
6: **end for**
7: $\mathrm{M}^- := \emptyset \quad \mathrm{M}^+ := \emptyset$
8: **for each** $A \in \mathsf{CN}^\top(\mathcal{O}^2)$ **do**
9: $\quad$ **for each** $\alpha \in \Delta^-\mathcal{O}$ **do**
10: $\quad\quad$ **if not** local$(\alpha, \mathsf{Sig}(\mathcal{O}^1_A))$ **then**
11: $\quad\quad\quad \mathrm{M}^- := \mathrm{M}^- \cup \{A\}$
12: $\quad\quad$ **end if**
13: $\quad$ **end for**
14: $\quad$ **for each** $\alpha \in \Delta^+\mathcal{O}$ **do**
15: $\quad\quad$ **if not** local$(\alpha, \mathsf{Sig}(\mathcal{O}^1_A))$ **then**
16: $\quad\quad\quad \mathrm{M}^+ := \mathrm{M}^+ \cup \{A\}$
17: $\quad\quad$ **end if**
18: $\quad$ **end for**
19: **end for**
20: **for each** $A \in \mathsf{CN}^\top(\mathcal{O}^2)$ **do**
21: $\quad$ **if** $A \in \mathrm{M}^- \cup \mathrm{M}^+$ **then**
22: $\quad\quad \mathcal{O}^2_A := \mathsf{extract\_module}(\{A\}, \mathcal{O}^2)$
23: $\quad$ **else**
24: $\quad\quad \mathcal{O}^2_A := \mathcal{O}^1_A$
25: $\quad$ **end if**
26: $\quad$ **for each** $B \in \mathsf{CN}(\mathcal{O}^2) \cup \{\bot\}$ **do**
27: $\quad\quad$ **if** $(A \in \mathrm{M}^-$ **and** $A \sqsubseteq_1 B)$ **or**
28: $\quad\quad\quad (A \in \mathrm{M}^+$ **and** $A \not\sqsubseteq_1 B)$ **then**
29: $\quad\quad\quad A \sqsubseteq_2 B := \mathsf{subsumes}(\mathcal{O}^2_A, \langle A, B\rangle)$
30: $\quad\quad$ **else**
31: $\quad\quad\quad A \sqsubseteq_2 B := A \sqsubseteq_1 B$
32: $\quad\quad$ **end if**
33: $\quad$ **end for**
34: **end for**
35: **return** $\mathcal{O}^2, \sqsubseteq_2, A \to \mathcal{O}^2_A$

the algorithm internally maintains the modules $\mathcal{O}^1_A$ and $\mathcal{O}^2_A$ for every atomic concept A and the modules $\mathcal{O}^1_\top$ and $\mathcal{O}^2_\top$ for the empty signature. We will show that maintaining these additional modules does not involve a significant overhead in practice. The algorithm consists of the following phases:

1. *Processing of the new symbols (lines 2–6):* The modules $\mathcal{O}^1_A$ and the subsumption partial order $\sqsubseteq_1$ for $\mathcal{O}^1$ are extended for every newly introduced atomic concept

D. The module for D, about which nothing has been said yet, is equivalent to the module for the empty signature $\mathcal{O}^1_\top$. Thus, we have: (i) $\mathcal{O}^1_D = \mathcal{O}^1_\top$, (ii) $\mathcal{O}^1 \models D \sqsubseteq B$ iff $\mathcal{O}^1 \models \top \sqsubseteq B$, and (iii) $\mathcal{O}^1 \models A \sqsubseteq D$ iff $\mathcal{O}^1 \models A \sqsubseteq \bot$.

2. *Identifying the affected modules (lines 7–19):* The sets M^- and M^+ contain those $A \in \mathsf{CN}^\top(\mathcal{O}^1)$ for which the corresponding modules must be modified by removing and/or adding axioms. If α is removed from $\mathcal{O}^1$ and is non-local w.r.t. $\mathsf{Sig}(\mathcal{O}^1_A)$ then at least α should be removed from $\mathcal{O}^1_A$. If α is added to $\mathcal{O}^1$ and is non-local w.r.t. $\mathsf{Sig}(\mathcal{O}^1_A)$, then $\mathcal{O}^1_A$ needs to be extended at least with α.
3. *Computing new modules and subsumptions (lines 20–34):* The affected modules found in the previous phase are re-extracted and the others are just copied (lines 21–25). Then, every subsumption $A \sqsubseteq B$, using Proposition 1, is either recomputed against the module $\mathcal{O}^2_A$, or is reused from $\mathcal{O}^1$ (lines 26–33).

In Algorithm 3, the procedure $\mathsf{extract\_module}(\mathbf{S}, \mathcal{O})$ refers to Algorithm 2. The procedure $\mathsf{subsumes}(\mathcal{O}, \langle A, B\rangle)$ uses a reasoner to check if $\mathcal{O}$ entails the subsumption $A \sqsubseteq B$. The correctness of the algorithm is easy to prove using Proposition 1 and Proposition 3.

It is worth emphasizing that, in our algorithm, the reasoner is only used as a black box to answer subsumption queries. This provides two important advantages: first, the internals of the reasoner need not be modified and, second, *any* sound and complete reasoner for OWL 2 can be used, independently of the reasoning technique it is based on (tableaux, resolution, or any other).

We finally illustrate the execution of Algorithm 3 on the ontologies $\mathcal{O}^1$ and $\mathcal{O}^2$ in Table 6 from Section 3, where the sets $\Delta^-\mathcal{O}$ and $\Delta^+\mathcal{O}$ of removed and added axioms are given in the lower part of Table 6.

In our case, $\mathcal{O}^2$ doesn't introduce new atomic concepts w.r.t. $\mathcal{O}^1$. Thus, Phase 1 in Algorithm 3 can be skipped. The sets $\mathrm{M}^-, \mathrm{M}^+$ computed in Phase 2 are as follows: $\mathrm{M}^- = \{\mathsf{Cystic\_Fibrosis}, \mathsf{Pancreatic\_Fibrosis}\}$ and $\mathrm{M}^+ = \{\mathsf{Cystic\_Fibrosis}\}$ since the axiom in $\Delta^-\mathcal{O}$ (see Table 6) is not syntactically local w.r.t. the signature of the module in $\mathcal{O}^1$ for $\mathsf{Cystic\_Fibrosis}$ and $\mathsf{Pancreatic\_Fibrosis}$; analogously, the axiom in $\Delta^+\mathcal{O}$ is non-local w.r.t. the signature of the module in $\mathcal{O}^1$ for $\mathsf{Cystic\_Fibrosis}$ (see Table 10 for the modules in $\mathcal{O}^1$). In Phase 3, the modules for $\mathsf{Cystic\_Fibrosis}$ and $\mathsf{Pancreatic\_Fibrosis}$ are re-computed. In the former module, only the subsumption relations between $\mathsf{Cystic\_Fibrosis}$ and $\mathsf{Pancreatic\_Fibrosis}$ and their subsumers in $\mathcal{O}^1$ need to be recomputed; in the latter one, only the subsumption relations between the non-subsumers of $\mathsf{Cystic\_Fibrosis}$ in $\mathcal{O}^1$ are computed.

4.4 An Optimized Module Extraction Algorithm for $\mathcal{EL}^+$

In this section, we discuss optimizations of the incremental classification procedure presented in the previous section that are specific to $\mathcal{EL}^+$ ontologies.

It turns out that for a non-trivial $\mathcal{EL}^+$ axiom, testing syntactic locality for $\mathbf{S}$ can be reduced to checking whether the left hand side of the axiom contains a symbol that is not in $\mathbf{S}$.

Proposition 4 *Let $\alpha = \alpha_L \sqsubseteq \alpha_R$ be an $\mathcal{EL}^+$ axiom and $\mathbf{S}$ a signature. Then α is syntactically local for $\mathbf{S}$ iff either $\mathsf{Sig}(\alpha_L) \nsubseteq \mathbf{S}$ or $\alpha_R = \top \sqcap \cdots \sqcap \top$.*

Proof An $\mathcal{EL}^+$ axiom α can be either a RIA $w \sqsubseteq R$, or a GCI $C_1 \sqsubseteq C_2$.

If $\alpha = w \sqsubseteq R$ then, by Definition 5, α is syntactically local iff $w = R_1 \dots R_n$ and there exists i with $1 \leq i \leq n$ such that $R_i \notin \mathbf{S}$.

If $\alpha = C_1 \sqsubseteq C_2$ then by Definition 5, α is syntactically local iff either $C_1 \in \mathbf{Con}^{\emptyset}(\mathbf{S})$ or $C_2 \in \mathbf{Con}^{\Delta}(\mathbf{S})$. Since $\mathcal{EL}^+$ does not allow for concept negation, it is easy to see from the definition of $\mathbf{Con}^{\Delta}(\mathbf{S})$ that $C_2 \in \mathbf{Con}^{\Delta}(\mathbf{S})$ iff $C_2 = \top \sqcap \dots \sqcap \top$. By induction on definition of $\mathbf{Con}^{\emptyset}(\mathbf{S})$ it is easy to show that $C_1 \in \mathbf{Con}^{\emptyset}(\mathbf{S})$ iff C_1 contains either a role $R^0 \notin \mathbf{S}$ or an atomic concept $A^0 \notin \mathbf{S}$. □

Using Proposition 4, one can significantly simplify Algorithm 2 for extracting syntactic locality-based modules in $\mathcal{EL}^+$ ontologies. To formulate the optimized version, we introduce an auxiliary notion of reachable symbols with respect to an ontology and a signature.

Definition 6 (Reachability) Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology and $\mathbf{S}$ a signature. The set of symbols *reachable from* $\mathbf{S}$ *in* $\mathcal{O}$ is the smallest set $\mathsf{Reach}(\mathbf{S}, \mathcal{O})$ such that (i) $\mathbf{S} \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$, and (ii) for every axiom $\alpha \in \mathcal{O}$, $\mathsf{Sig}(\alpha_L) \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$ implies $\mathsf{Sig}(\alpha_R) \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$.

For example, given the ontology $\mathcal{O}^1$ on the left hand side of Table 6 and given $\mathbf{S} = \{\mathsf{Pancreatic\_Fibrosis}\}$, we have that $\mathsf{Reach}(\mathbf{S}, \mathcal{O}^1)$ consists of $\mathbf{S}$ plus the following symbols: Fibrosis, Pancreatic_Disorder, Disorder, Pancreas, and located_in.

Lemma 1 *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology, $\mathbf{S}$ a signature, and $\mathcal{O}_1$ a syntactic locality-based $\mathbf{S}$-module in $\mathcal{O}$. Then* $\mathsf{Reach}(\mathbf{S}, \mathcal{O}) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$.

Proof We demonstrate that the set $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$ is closed under the conditions (i) and (ii) of Definition 6, namely that (i) $\mathbf{S} \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$, and (ii) for every axiom $\alpha \in \mathcal{O}$, $\mathsf{Sig}(\alpha_L) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$ implies $\mathsf{Sig}(\alpha_R) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$. This implies that $\mathsf{Reach}(\mathbf{S}, \mathcal{O}) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$ since by this definition, $\mathsf{Reach}(\mathbf{S}, \mathcal{O})$ is the smallest set that satisfies conditions (i) and (ii).

Since (i) is obvious, we focus on the condition (ii). Assume that $\mathsf{Sig}(\alpha_L) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$ for some $\alpha \in \mathcal{O}$. If α is syntactically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$, then by Proposition 4, this is only possible when $\alpha_R = \top \sqcap \dots \sqcap \top$, and therefore $\mathsf{Sig}(\alpha_R) = \emptyset \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$. Otherwise, if α is not syntactically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$, then $\alpha \in \mathcal{O}_1$, and therefore $\mathsf{Sig}(\alpha_R) \subseteq \mathsf{Sig}(\alpha) \subseteq \mathsf{Sig}(\mathcal{O}_1) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$. □

Lemma 2 *Let $\mathcal{O}$ be an $\mathcal{EL}^+$ ontology and $\mathcal{O}_1$ the set of all axioms $\alpha \in \mathcal{O}$ such that* $\mathsf{Sig}(\alpha) \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$ *and $\alpha_R \neq \top \sqcap \dots \sqcap \top$. Then $\mathcal{O}_1$ is the smallest syntactic locality-based $\mathbf{S}$-module in $\mathcal{O}$.*

Proof In order to show that $\mathcal{O}_1$ is a syntactic locality-based $\mathbf{S}$-module in $\mathcal{O}$, we need to prove that every axiom $\alpha \in \mathcal{O} \setminus \mathcal{O}_1$ is syntactically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$. If $\alpha \in \mathcal{O} \setminus \mathcal{O}_1$ then by definition of $\mathcal{O}_1$, we have either $\mathsf{Sig}(\alpha) \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$ or $\alpha_R = \top \sqcap \dots \sqcap \top$. Then by Proposition 4, α is syntactically local for $\mathsf{Reach}(\mathbf{S}, \mathcal{O})$. Since $\mathbf{S} \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$ by Definition 6, and $\mathsf{Sig}(\mathcal{O}_1) \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$ by definition of $\mathcal{O}_1$, we have that α is syntactically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$.

Let $\mathcal{O}_2$ be the smallest syntactic locality-based $\mathbf{S}$-module in $\mathcal{O}$ (for example the one computed by Algorithm 2). We demonstrate that $\mathcal{O}_1 = \mathcal{O}_2$. Clearly $\mathcal{O}_2 \subseteq \mathcal{O}_1$ since $\mathcal{O}_1$ is a syntactic locality-based $\mathbf{S}$-module. In order to show that $\mathcal{O}_1 \subseteq \mathcal{O}_2$, pick any $\alpha \in \mathcal{O}_1$. We demonstrate that $\alpha \in \mathcal{O}_2$. By definition of $\mathcal{O}_1$, we have $\mathsf{Sig}(\alpha) \subseteq \mathsf{Reach}(\mathbf{S}, \mathcal{O})$ and $\alpha_R \neq \top \sqcap \cdots \sqcap \top$. Hence by Proposition 4, α is not syntactically local for $\mathsf{Reach}(\mathbf{S}, \mathcal{O})$. By Lemma 1, since $\mathcal{O}_2$ is a syntactic locality-based $\mathbf{S}$-module in $\mathcal{O}$, we have $\mathsf{Reach}(\mathbf{S}, \mathcal{O}) \subseteq \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_2)$. Therefore, α is not syntactically local for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_2)$, and thus $\alpha \in \mathcal{O}_2$. □

The advantage of reachability as a characterization of locality, is that the set of reachable symbols can be computed in linear time, and hence, the smallest syntactic locality-based module can be extracted in linear time.

Proposition 5 *Given an $\mathcal{EL}^+$ ontology $\mathcal{O}$ and a signature $\mathbf{S}$ the set $\mathsf{Reach}(\mathbf{S}, \mathcal{O})$ can be computed in linear time in $|\mathcal{O}| + |\mathbf{S}|$.*

Proof The computation of reachable symbols can be linearly reduced to unit propagation for propositional Horn clauses. Each $\alpha \in \mathcal{O}$ with $\mathsf{Sig}(\alpha_L) = \{l_1, \ldots, l_m\}$ and $\mathsf{Sig}(\alpha_R) = \{r_1, \ldots, r_n\}$ can be translated into the following set of Horn clauses, where t is a freshly introduced symbol: $\{l_1 \wedge \cdots \wedge l_m \rightarrow t\} \cup \bigcup_{1 \leq i \leq n} \{t \rightarrow r_i\}$. It is easy to see that $\mathsf{Reach}(\mathbf{S}, \mathcal{O})$ corresponds to the set of propositional from $\mathbf{S}$ that are implied by these Horn clauses. This set of propositional atoms can be computed in linear time (e.g., using the Dowling-Gallier algorithm [15]). □

Corollary 1 *There exists a linear-time algorithm that given an $\mathcal{EL}^+$ ontology $\mathcal{O}$ and a signature $\mathbf{S}$, computes the minimal syntactic locality-based $\mathbf{S}$-module in $\mathcal{O}$.*

Proof Immediate from Proposition 5 and Lemma 2. □

5 A Direct Incremental Classification of $\mathcal{EL}^+$ Ontologies Modulo Additions

In this section, we describe a modification of the classification algorithm for $\mathcal{EL}^+$ from Section 2.2 to allow for incremental reasoning over additions of axioms to ontologies.

The idea is quite simple. Assume that an ontology $\mathcal{O}^1$ has been classified using Algorithm 1 by applying the completion rules in Table 4. In order to classify an ontology $\mathcal{O}^2 = \mathcal{O}^1 \cup \Delta^+\mathcal{O}$ obtained from $\mathcal{O}^1$ by adding new axioms $\Delta^+\mathcal{O}$, it is sufficient to add the normalization of axioms in $\Delta^+\mathcal{O}$ to the result of the completion for $\mathcal{O}^1$ and apply the inference rules in Table 4 that involve the newly added axioms.

This idea is formalized in Algorithm 4. Given the original ontology $\mathcal{O}^1$ in normal form, the relation $\sqsubseteq_1$ for $\mathcal{O}^1$ computed by Algorithm 1, and the new axioms $\Delta^+\mathcal{O}$, the algorithm computes the classification $\sqsubseteq_2$ for $\mathcal{O}^2 = \mathcal{O}^1 \cup \Delta^+\mathcal{O}$ using the completion rules in Table 11. These completion rules are restrictions of the completion rules for Algorithm 1 in Table 4 (when applied for the union of $\sqsubseteq_1$ and $\sqsubseteq_2$ and $\mathcal{O}^2$) for the cases when one of the premises is either a normalized axiom from $\Delta^+\mathcal{O}$, or a new subsumption $C_1 \sqsubseteq_2 C_2$ since the remaining inferences produce subsumptions for $\mathcal{O}^1$

Algorithm 4 Incremental Classification of $\mathcal{EL}^+$ Ontologies Modulo Additions

Procedure inc_classify_elp($\mathcal{O}^1, \Delta^+\mathcal{O}, \sqsubseteq_1$)

Input:
$\mathcal{O}^1, \Delta^+\mathcal{O}$: $\mathcal{EL}^+$ ontology in normal form;
$\sqsubseteq_1$: output of classify($\mathcal{O}^1$);

Output:
$\sqsubseteq_2$: relation;

1: $\Delta^+\mathcal{O} := \mathsf{Norm}(\Delta^+\mathcal{O})$
2: $\mathcal{O}^2 := \mathcal{O}^1 \cup \Delta^+\mathcal{O}$
3: $\sqsubseteq_2 := \emptyset$
4: **repeat**
5: Apply rules from Table 11
6: **until** no other rule applies
7: $\sqsubseteq_2 := \sqsubseteq_1 \cup \sqsubseteq_2$
8: **return** $\sqsubseteq_2$

I1	$A \in \mathsf{CN}(\mathcal{O}^2) \setminus \mathsf{CN}(\mathcal{O}^1) \vdash$	$A \sqsubseteq_2 A$
I2	$A \in \mathsf{CN}(\mathcal{O}^2) \setminus \mathsf{CN}(\mathcal{O}^1) \vdash$	$A \sqsubseteq_2 \top$
CR1	$A \sqsubseteq_1 B, B \sqsubseteq C \in \Delta^+\mathcal{O} \vdash$	$A \sqsubseteq_2 C$
	$A \sqsubseteq_2 B, B \sqsubseteq C \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 C$
CR2	$A \sqsubseteq_1 B_1, A \sqsubseteq_1 B_2, B_1 \sqcap B_2 \sqsubseteq C \in \Delta^+\mathcal{O} \vdash$	$A \sqsubseteq_2 C$
	$A \sqsubseteq_2 B_1, A \sqsubseteq_1 B_2, B_1 \sqcap B_2 \sqsubseteq C \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 C$
	$A \sqsubseteq_1 B_1, A \sqsubseteq_2 B_2, B_1 \sqcap B_2 \sqsubseteq C \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 C$
	$A \sqsubseteq_2 B_1, A \sqsubseteq_2 B_2, B_1 \sqcap B_2 \sqsubseteq C \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 C$
CR3	$A \sqsubseteq_1 B, B \sqsubseteq \exists R.C \in \Delta^+\mathcal{O} \vdash$	$A \sqsubseteq_2 \exists R.C$
	$A \sqsubseteq_2 B, B \sqsubseteq \exists R.C \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 \exists R.C$
CR4	$A \sqsubseteq_1 \exists R.B, B \sqsubseteq_1 C, \exists R.C \sqsubseteq D \in \Delta^+\mathcal{O} \vdash$	$A \sqsubseteq_2 D$
	$A \sqsubseteq_2 \exists R.B, B \sqsubseteq_1 C, \exists R.C \sqsubseteq D \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 D$
	$A \sqsubseteq_1 \exists R.B, B \sqsubseteq_2 C, \exists R.C \sqsubseteq D \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 D$
	$A \sqsubseteq_2 \exists R.B, B \sqsubseteq_2 C, \exists R.C \sqsubseteq D \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 D$
CR5	$A \sqsubseteq_1 \exists R.B, R \sqsubseteq S \in \Delta^+\mathcal{O} \vdash$	$A \sqsubseteq_2 \exists S.B$
	$A \sqsubseteq_2 \exists R.B, R \sqsubseteq S \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 \exists S.B$
CR6	$A \sqsubseteq_1 \exists R.B, B \sqsubseteq_1 \exists S.C, RS \sqsubseteq T \in \Delta^+\mathcal{O} \vdash$	$A \sqsubseteq_2 \exists T.C$
	$A \sqsubseteq_2 \exists R.B, B \sqsubseteq_1 \exists S.C, RS \sqsubseteq T \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 \exists T.C$
	$A \sqsubseteq_1 \exists R.B, B \sqsubseteq_2 \exists S.C, RS \sqsubseteq T \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 \exists T.C$
	$A \sqsubseteq_2 \exists R.B, B \sqsubseteq_2 \exists S.C, RS \sqsubseteq T \in \mathcal{O}^2 \vdash$	$A \sqsubseteq_2 \exists T.C$

Table 11 Completion rules for incremental classification in $\mathcal{EL}^+$

which have already been computed. The correctness of Algorithm 4 follows directly from the correctness of Algorithm 1.

Although Algorithm 4 is easier to implement than Algorithm 2, its usefulness for incremental ontology reasoning is somewhat limited due to the fact that the deletion or modification of axioms is not considered. There are, however, several use cases when the incremental classification algorithm just for additions can be used.

A practically relevant scenario is when the ontology under development can be partitioned as $\mathcal{O} = \mathcal{O}_p \cup \mathcal{O}_t$ where $\mathcal{O}_p$ is a *permanent* part of $\mathcal{O}$ that the user is not supposed to modify, and $\mathcal{O}_t$ is *temporary* part of $\mathcal{O}$ which the user is currently working on. This assumption is reasonable in many practical use cases, e.g., when $\mathcal{O}_p$ is an ontology that is being reused / imported, or when $\mathcal{O}_p$ consists of axioms that cover a domain on which the current user is not an expert. Under this assumption, it is possible to classify an ontology $\mathcal{O} = \mathcal{O}_p \cup \mathcal{O}_t$ by reusing the result of the classification for $\mathcal{O}_p$, which does not change. If the size of $\mathcal{O}_t$ is small compared to the size of $\mathcal{O}_p$, then the incremental classification is likely to be more efficient than the classification of $\mathcal{O}$ from scratch.

Another situation where incremental classification modulo additions can be very useful is the computation of subsumption queries involving complex concepts. The classification algorithm for $\mathcal{EL}^+$ computes only subsumptions between atomic concepts and (possibly existentially restricted) atomic concepts. However, a reasoner is often required to check subsumptions between more complex concepts, such as (2).

$$\mathsf{Pancreatic\_Fibrosis} \sqcap \exists \mathsf{has\_Origin}.\mathsf{Genetic\_Origin} \sqsubseteq \mathsf{Disorder} \sqcap \exists \mathsf{located\_In}.\mathsf{Pancreas} \quad (2)$$

Reasoners for expressive DLs reduce subsumption queries between complex concepts to concept satisfiability using the following property: $\mathcal{O} \models C \sqsubseteq D$ iff $C \sqcap \neg D$ is not satisfiable w.r.t. $\mathcal{O}$. Since $\mathcal{EL}^+$ does not allow for negations, checking the entailment $\mathcal{O} \models C \sqsubseteq D$ can be reduced to checking $\mathcal{O} \cup \{X \sqsubseteq C,\ D \sqsubseteq Y\} \models X \sqsubseteq Y$, where X and Y are fresh atomic concepts. This subsumption can be checked incrementally by classifying $\mathcal{O} \cup \{X \sqsubseteq C,\ D \sqsubseteq Y\}$ using Algorithm 4 given the classification for $\mathcal{O}$. For example, in order to check whether the complex subsumption (2) is entailed by the ontology $\mathcal{O}^1$ in Table 1, we need to check whether the subsumption $X \sqsubseteq Y$ holds with respect to $\mathcal{O}^1$ enriched with the following axioms:

$$X \sqsubseteq \mathsf{Pancreatic\_Fibrosis} \sqcap \exists \mathsf{has\_Origin}.\mathsf{Genetic\_Origin} \quad (3)$$

$$\mathsf{Disorder} \sqcap \exists \mathsf{located\_In}.\mathsf{Pancreas} \sqsubseteq Y \quad (4)$$

Axioms (3) and (4) are normalized to axioms Ax15–Ax18 below:

Ax15: $X \sqsubseteq \mathsf{Pancreatic\_Fibrosis}$
Ax16: $X \sqsubseteq \exists \mathsf{has\_Origin}.\mathsf{Genetic\_Origin}$
Ax17: $\exists \mathsf{located\_In}.\mathsf{Pancreas} \sqsubseteq \mathsf{Aux3}$
Ax18: $\mathsf{Disorder} \sqcap \mathsf{Aux3} \sqsubseteq Y$

Now, the additional subsumption relations $\sqsubseteq_2$ for $\mathcal{O}^1$ extended with the new axioms can be computed as in Table 12 by applying the rules in Table 11 to the subsumption relations for $\sqsubseteq_1$ derived in Table 5 and the new axioms Ax15–Ax18. As we can see the subsumption $X \sqsubseteq_2 Y$ has been derived. Therefore, the complex subsumption (2) is entailed by $\mathcal{O}^1$.

		New subsumption in $\sqsubseteq_2$	Rule Applied
	N0	$X \sqsubseteq_2 X; Y \sqsubseteq_2 Y$	I1$[X]$; I1$[Y]$
	N1	$X \sqsubseteq_2 \top; Y \sqsubseteq_2 \top$	I2$[X]$; I2$[Y]$
	N2	$X \sqsubseteq_2$ Pancreatic_Fibrosis	CR1(N0,Ax15)
	N3	$X \sqsubseteq_2 \exists$has_Origin.Genetic_Origin	CR1(N0,Ax16)
	N4	$X \sqsubseteq_2$ Pancreatic_Disorder	CR1(N2,Ax10)
	N5	$X \sqsubseteq_2$ Fibrosis	CR1(N2,Ax9)
	N6	$X \sqsubseteq_2 \exists$located_In.Pancreas	CR3(N4,Ax14)
	N7	$X \sqsubseteq_2$ Aux1	CR4(N6,S0,Ax4)
	N8	$X \sqsubseteq_2$ Cystic_Fibrosis	CR2(N7,N5,Ax3)
	N9	$X \sqsubseteq_2$ Disorder	CR1(N4,Ax13)
	N10	$X \sqsubseteq_2$ Pancreatic_Fibrosis	CR2(N5,N4,Ax11)
	N11	$X \sqsubseteq_2$ Aux2	CR4(N3,S0,Ax8)
	N12	$X \sqsubseteq_2$ Genetic_Fibrosis	CR2(N5,N11,Ax7)
	N13	$X \sqsubseteq_2$ Genetic_Disorder	CR1(N12,Ax12)
	N14	$X \sqsubseteq_2$ Aux3	CR4(N6,S0,Ax17)
	N15	Cystic_Fibrosis $\sqsubseteq_2$ Aux3	CR4(S3,S0,Ax17)
	N16	Pancreatic_Disorder $\sqsubseteq_2$ Aux3	CR4(S10,S0,Ax17)
	N17	Pancreatic_Fibrosis $\sqsubseteq_2$ Aux3	CR4(S13,S0,Ax17)
	N18	Pancreatic_Fibrosis $\sqsubseteq_2 Y$	CR2(S11,N17,Ax18)
	N19	Pancreatic_Disorder $\sqsubseteq_2 Y$	CR2(S9,N8,Ax18)
✓	N20	$X \sqsubseteq_2 Y$	CR2(N14,N9,Ax18)

Table 12 Incremental Classification of an $\mathcal{EL}^+$ ontology. The axioms Ax1–Ax15 are from Table 3 and the inferences S0–S16 from Table 5

6 Empirical Evaluation

In this section, we present an empirical evaluation of the incremental reasoning techniques described in this paper over a collection of commonly used ontologies. In Section 6.1 we describe our test ontologies and experimental setup. In Section 6.2 we evaluate our module-based reasoning algorithm from Section 4. Section 6.3 is devoted to $\mathcal{EL}^+$ ontologies and reasoning modulo additions using the technique described in Section 5. Finally, we compare the incremental reasoning techniques described in Sections 4 and 5 and make some concluding remarks.

6.1 Test Ontologies and Experimental Environment

The experiments have been performed using the OWL DL reasoner Pellet (v.1.5.2), which implements a tableau-based procedure, and the $\mathcal{EL}^+$ reasoner CEL (v.1.0), which implements Algorithm 1 in Section 2.2. Both Pellet and CEL have been used as a "black box" to evaluate the module-based technique from Section 4 — that is, we did not modify the internals of the reasoners. In contrast, in order to evaluate the incremental procedure modulo additions described in Section 5, we have extended the internals of CEL with the new functionality.

As a test suite, we have selected a set of well-known and commonly used ontologies, which are given in Table 13. The table provides some basic information about the test ontologies, including the language in which they are expressed, the number of atomic concepts and axioms they contain, the time required for both reasoners to

Ontology	DL	No.Atom. Conc.	No. Axioms	Classif. Time Pellet	Cel	% Subs	Modul. Time	Mod. Size (Avg/Max)
SWEET	$\mathcal{SHOIF}$	1387	2206	3.0	-	0.37	P: 9.2	414 / 480
NotGalen	$\mathcal{SHF}$	2748	4529	19.2	-	0.37	P: 4.5	75 /530
NotGalen$^-$	$\mathcal{EL}^+$	2748	3575	15.8	1.7	0.37	P: 3.9	10 /125
GO	$\mathcal{EL}^+$	20465	28897	79.0	1.1	0.04	P: 69.6	18 / 161
FullGalen$^-$	$\mathcal{EL}^+$	23136	26084	-	120.0	0.08	-	- / -
NCI	$\mathcal{EL}^+$	27652	46940	33.5	2.2	0.03	P: 70.6	29 / 436
SNOMED	$\mathcal{EL}^+$	379691	379704	-	685.1	0.00	C: 3110	31 / 261

Table 13 Information about test ontologies. Time in seconds. Fractional values are rounded. The precise % of positive subsumptions in SNOMED is 0.0008%.

classify them, and an estimation of the total number of entailed subsumptions relative to the number of possible non-trivial subsumptions. Note that for each of the test ontologies this latter value is smaller than 1%. Finally, the last two columns of Table 13 provide, for each of the ontologies evaluated in Section 6.2, the time required to compute the locality-based module for each atomic concept in the ontology using Algorithm 2, and the average and maximum size of the extracted modules.

NASA's SWEET ontology[1] is the smallest of the tested ontologies, but also the one that uses the largest language. NotGalen[2] is based on an early version of well-known Galen ontology.[3] NotGalen$^-$ is obtained from NotGalen by removing functionality assertions on roles. The Gene Ontology (GO) [4] is one of the largest Open Biomedical Ontologies.[5] FullGalen$^-$ is obtained from a recent version of Galen[6] by removing functionality and inverses. NCI[7] is a widely used ontology developed at the National Cancer Institute. Finally, SNOMED[8] is probably the largest and the most widely used of the medical ontologies currently available. All ontologies, except for SWEET and NotGalen are expressible in $\mathcal{EL}^+$ and can be handled by CEL. Classification results for FullGalen$^-$ and SNOMED using Pellet have been omitted as the classification time exceeded our 20 minute threshold. The module extraction times for all ontologies except for FullGalen$^-$ and SNOMED have been obtained using our extension of Pellet and will be relevant to experiments in Section 6.2. We have also computed modules for SNOMED using an extension of CEL, which will be relevant to experiments in Section 6.4 where we compare the module-based and direct approaches for $\mathcal{EL}^+$ ontologies.

For our experiments with Pellet, we have used a 3GHz PC with 4GB RAM operated by Red Hat Enterprise Linux AS (release 4) with Java v.1.5 using 2GB of memory. For the experiments with CEL, we have used a similar machine with a 3.16GHz

[1] `http://sweet.jpl.nasa.gov/ontology/`
[2] `http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/galen.owl`
[3] `http://www.opengalen.org/`
[4] `http://www.geneontology.org`
[5] `http://www.obofoundry.org/`
[6] `http://www.co-ode.org/galen/`
[7] `http://www.mindswap.org/2003/CancerOntology/nciOncology.owl`
[8] `http://www.snomed.org/`

	n	1: ♯ Mod. Affected (Av / Mx)	2: ♯ Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: ♯ New (Non)Sub. (Av / Mx)	7: ♯ Mod. (Non)Sub (Av / Mx)
SWEET	1	116/1387	530/2139	1.6/8.7	0.16/2.3	1.8/10.5	180/8554	29.3/1334
SWEET	2	200/1387	669/2139	2.1/8.5	0.2/1.2	2.3/9.8	209/9454	46/1373
SWEET	4	241/1387	745/2139	2.2/7.8	0.2/1.1	2.4/9.1	198/7999	38/1274
SWEET	8	614/1387	1214/2136	4.4/9.6	0.5/1.2	4.9/10.9	107/1477	34.5/367
NotGalen	1	102/933	678/3020	0.6/3.5	2.1/17.9	2.6/21.5	193/4776	17/326
NotGalen	2	232/1267	1185/3175	1.2/4.1	4.8/19.3	6.05/22.5	128/1565	17.1/152
NotGalen	4	294/1755	1503/3659	1.5/4.1	6.6/27.1	8.1/28	340/5394	49.9/770
NotGalen	8	507/2721	2106/4244	2.2/4.3	8.4/17.5	10.8/21.6	564/7931	138.2/2721
GO	1	28/757	113/1921	0.24/5.6	0.05/1.4	0.3/7.1	209/8159	22.7/753
GO	2	59/1720	284/3562	0.6/9.8	0.1/4.4	0.9/14.4	489/14307	52.1/1649
GO	4	48/269	328/902	0.6/0.22	0.1/0.4	0.9/2.8	359/5107	32.6/267
GO	8	205/3463	925/9916	2/26.8	0.3/6.1	2.8/33.6	897/26414	127.3/3329
NCI	1	11 / 225	183 / 2767	0.18 / 4.45	0.1 / 1.0	0.3 / 5.6	39.8 / 268	5.3/50
NCI	2	52/ 886	434/2811	0.42/3.6	0.15/1.3	0.72/5.1	392/12924	30.1/859
NCI	4	190/2123	1324/9375	1.82/17.9	0.61/5.7	2.74/23.8	1653/21625	132.1/1813
NCI	8	320/6578	2092/18999	3.03/33.3	0.99/12.1	4.6/56.1	1268/12181	119.5/1264

Table 14 Module-based incremental classification using Pellet. Time in seconds.

processor, 2GB RAM, and operated by Ubuntu Linux. Due to some technical reasons, CEL cannot allocate more than 800MB of memory.

6.2 Module-Based Incremental Classification

In order to evaluate the performance of our module-based incremental reasoning approach described in Section 4, we have implemented Algorithm 3 and used Pellet for evaluating subsumption queries over modules. Our implementation is, however, independent from Pellet, and our results intend to determine the usefulness of our approach for optimizing any reasoner. Our system implements a slightly simplified version of Algorithm 3. In particular, once the affected modules have been identified, our implementation simply reclassifies the union of these modules using Pellet to determine the new subsumption relations, instead of using the procedure described in lines 20–34 of Algorithm 3.

We have performed the following experiment for the ontologies in Table 13 that Pellet can classify: for various values of n, we have: **1)** removed n random axioms; **2)** classified the resulting ontology using Pellet; then, we have repeated the following two steps 50 times: **3)** extracted the minimal locality-based module for each atomic concept, **4)** randomly removed an additional n axioms, added back the previously removed n axioms, and reclassified the ontology using our incremental algorithm.

Our goal is to simulate the ontology evolution process when n axioms have been modified (each of these modifications can be viewed as a simultaneous deletion and addition of an axiom). All the results have been gathered during step **4)** of the experiment. We have considered different types of axioms to be modified, namely concept definitions, GCIs and role axioms.

	n	1: ♯ Mod. Affected (Av / Mx)	2: ♯ Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: ♯ New (Non)Sub. (Av / Mx)	7: ♯ Mod. (Non)Sub (Av / Mx)
SWEET	1	341/1387	586/2140	1.8/7.2	0.3/3.8	2.2/11.1	177/8851	27.7/1381
SWEET	2	570/1387	957/2140	3.1/7.7	0.4/1	3.5/8.8	0.7/37	0.3/14
NotGalen	1	159 / 1252	980/3292	0.9/4.1	3.4/20.8	4.3/21.8	236/5403	18.6/427
NotGalen	2	481/1906	1758/3781	1.9/4.6	7.5/21.5	9.5/25.2	442.1/9892	34.8/844
NCI	1	1027 / 10035	5752 / 28843	10.8 / 57.7	4.1 / 28	15 / 85.9	2783 / 13094	198 / 9301
NCI	2	2076 /10437	10934/29481	21.6/59.4	8.5/28.9	30.4/88.7	405/7130	32.5/407

Table 15 Module-based incremental classification using Pellet for varying role axioms. Time in seconds.

	n	1: ♯ Mod. Affected (Av / Mx)	2: ♯ Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)	6: ♯ New (Non)Sub. (Av / Mx)	7: ♯ Mod. (Non)Sub (Av / Mx)
SWEET	1	66/1387	478/2139	1.25/7.1	0.1/1.6	1.3/8.6	167/7245	39/1257
SWEET	2	125/1387	586/2139	1.6/7.4	0.1/1.1	1.6/8.5	206/8972	37/1357
NotGalen	1	153/1742	766/3653	0.6/4.1	2.8/19.9	3.5/24	176/5366	25/305
NotGalen	2	106/916	914/2688	0.7/3.1	2.9/15.4	3.7/17.8	213/2701	21/282
NCI	1	9/82	209/829	0.1/0.9	0.06/0.2	0.3/1.3	40/408	5.2/46
NCI	2	25/223	401/3183	0.4/4.8	0.1/1.1	0.7/6.1	237/4902	22.6/220

Table 16 Module-based incremental classification using Pellet for varying concept axioms. Time in seconds.

Table 14 summarizes the results of our experiments for $n = 1, 2, 4,$ and 8. Columns 1 and 2 indicate the number of affected modules and their total size, respectively. It can be observed that, for large ontologies such as GO and NCI, only a very small number of the modules are affected for a given update. These values correlate with the percentage of positive subsumption relations in ontologies and probably indicate that the concepts in these ontologies are "weakly inter-connected". Column 3 provides the total time spent in locating and re-extracting the affected modules. Column 4 shows the (re)-classification time for the union of the affected modules. Unsurprisingly, for GO and NCI, classification of modules is significantly faster that the full classification given in Table 13. For SWEET and NotGalen, the difference is not that substantial. For NotGalen, the maximum module classification time is actually larger than the time needed to classify the entire ontology. The fact that for NotGalen module classification is slower than ontology classification is unexpected. However, it is possible that after removing certain types of axioms from an ontology (e.g., role functionality axioms), tableau reasoners end up producing larger models, and consequently become slower. Column 5 presents the total time spent in updating the modules, loading them into the reasoner, and reclassifying them. Column 6 shows the sum of new subsumption and non-subsumption relations for each ontology, and Column 7 provides the number of modules which contain a new subsumption or non-subsumption after a change. The number of new (non)subsumptions is, in average, very small, which supports our hypothesis that changes do not typically affect a large portion of the original ontology. In the case of GO and NCI, more than 40% of the computed modules result in new (non)subsumptions. In the case of SWEET and NotGalen, this ratio varies from 5% to 25%.

Tables 15 and 16 contain our results for the particular case where the changes involve role axioms or concept axioms only, and $n = 1, 2$.[9] Unsurprisingly, the changes in role axioms have a much more substantial impact on both the number of affected modules and the number of new (non)subsumptions. Surprisingly, however, this impact is so strong in the case of NCI that the module-based approach does not outperform the full classification anymore.

To sum up, the results from our experiments suggest that the module-based incremental reasoning approach is especially useful in the following situations:

1. The ontology is large but formulated in a simple language such as $\mathcal{EL}^+$.
2. The ontology does not induce many dependencies between classes, and, in particular, has a small percentage of entailed subsumption relationships.
3. The changes do not involve role axioms.

6.3 Direct Incremental Classification for $\mathcal{EL}^+$-ontologies

In this section, we focus specifically on $\mathcal{EL}^+$ ontologies and evaluate the direct incremental reasoning procedure described in Section 5.

When trying to adapt the experimental setup described in Section 6.2 to the procedure described in Section 5, we are faced with two problems. First, the procedure assumes that the changes to the ontology involve additions of new axioms only. Second, as discussed in Section 5, the intended application scenario for our procedure is when the ontology is partitioned into a permanent and a temporary part. Thus, the temporary part as a whole should be considered as the ontology increment as opposed to just the modified axioms. In particular, we can no longer assume that the ontology increment involves just a few axioms.

Instead of varying the number of modified axioms in experiments, we vary the percentage p of the axioms in the temporary part of the ontology. For every tested value p and every $\mathcal{EL}^+$ ontology in Table 13, we have: **1)** removed $p\%$ random axioms; **2)** classified the resulting ontology using CEL; and **3)** added back the previously removed $p\%$ axioms (thus obtaining the full ontology) and reclassified it incrementally using Algorithm 4. We have repeated steps **1)**–**3)** five times for every ontology and averaged the obtained timings.[10] All results have been gathered during steps **2)** and **3)** of the experiment.

Table 17 summarizes the results of the experiments for values of p ranging between 0.5 and 5.0 with step 0.5. For each ontology and each value of p, we have measured the time required to classify the "permanent" part of the ontology (obtained after removing $p\%$ of axioms), and the time spent on the incremental classification of the additional $p\%$ of axioms. The important value in this table is the incremental classification time, since the permanent part of the ontology is assumed to be classified only once. Clearly, the larger is the temporary part, the longer it takes to compute the incremental classification. When compared with the classification times given in

[9] GO has not been included in Table 15 as it only contains one role axiom.

[10] Because each iteration involves full classification at step **2)**, these experiments are more time consuming

% Temp.	GO		NCI		NotGalen$^-$		FullGalen$^-$		SNOMED	
Ax. (p)	Perm.	Inc.	Perm.	Inc.	Perm.	Inc.	Perm.	Inc.	Perm.	Inc.
0.5	1.01	0.10	1.86	0.29	1.72	0.06	114.7	7.19	649.7	141.3
1.0	0.98	0.15	1.85	0.44	1.68	0.16	112.5	13.11	638.9	225.9
1.5	0.98	0.24	1.85	0.59	1.60	0.33	107.5	23.13	631.0	348.3
2.0	0.95	0.28	1.81	0.80	1.53	0.52	106.9	23.50	580.4	470.4
2.5	0.94	0.32	1.78	0.99	1.56	0.52	94.5	47.13	580.1	566.7
3.0	0.94	0.38	1.77	1.14	1.58	0.42	101.9	34.47	532.5	683.2
3.5	0.92	0.45	1.73	1.58	1.37	0.91	87.2	64.11	539.1	728.4
4.0	0.86	0.51	1.69	1.75	1.40	0.83	89.5	58.12	468.1	849.9
4.5	0.86	0.51	1.69	1.75	1.24	1.19	80.8	77.20	461.4	916.5
5.0	0.84	0.73	1.64	1.95	1.29	1.19	87.1	63.21	478.8	941.1

Table 17 Direct incremental classification using CEL. Time in seconds.

Table 13, we can clearly see that for all ontologies except for SNOMED the incremental classification time is smaller than the total classification time. For SNOMED the incremental approach provides benefits only when the temporary part contains at most 3% of the axioms in the ontology, which amounts to more than 11000 axioms. It can also be observed that the classification times vary almost linearly with p.

So far, we have considered in our experiments random modifications to ontologies. In practice, however, the performance of the incremental reasoning procedures presented here should be much better, since modifications are likely to involve axioms containing related concepts and, therefore, affect less modules and subsumption relations. The experiments performed in Section 6.2 suffer from this issue to a lesser degree since the number of the modified axioms is relatively small. The experiments in this section, however, can be considerably affected by random selections since they involve hundreds and even thousands of axioms.

To estimate a practical performance of the direct procedure, we have performed experiments with several historic releases of SNOMED. In the original sources of SNOMED, each atomic concept is annotated with dates, which indicate when the definition for this concept has been introduced or modified. The dates range between the years 1994 and 2008. From this information, we have reconstructed 16 distinct versions of SNOMED corresponding to the mentioned dates. The version for a date d consists of the axioms containing atomic concepts annotated either with d, or with an earlier date. Such reconstruction is by no means accurate since we cannot obtain the previous versions of the axioms, but it should provide a reasonable approximation to the actual releases of SNOMED.

The experimental results with the above-mentioned SNOMED releases are summarized in Table 18. The number of atomic concepts in these versions varies from about 92000 up to about 360000, and the classification times vary from around 8 seconds up to around 10 minutes. In our experiments, we have measured the incremental classification time between each pair of successive versions. Since the release dates were not evenly distributed, the differences between the successive versions vary significantly ranging from just a few axioms to almost 178000 axioms (an increment of 140%). Clearly, larger changes have resulted in a larger number of new subsumption relations and larger incremental classification times. However, when compared with the results in Table 17 for SNOMED, we can observe that the times for the same

Release Number	♯ Atom. Concepts	Class. Time	Release Transition	♯ New Atom. Concepts	Increm. Class. Time	New Subs.
v.1	92724	8.0	v.1-v.2	2611 (+2.82%)	22.7	34555
v.2	95335	8.9	v.2-v.3	597 (+0.63%)	5.7	6380
v.3	95932	9.0	v.3-v.4	668 (+0.70%)	10.6	9873
v.4	96600	9.1	v.4-v.5	4107 (+4.25%)	32.8	35032
v.5	100707	10.6	v.5-v.6	3910 (+3.88%)	138.9	111600
v.6	104617	12.7	v.6-v.7	18229 (+17.42%)	306.0	982725
v.7	122846	51.4	v.7-v.8	5 (+0.00%)	0.4	10
v.8	122851	52.2	v.8-v.9	4443 (+3.62%)	15.8	29428
v.9	127294	52.0	v.9-v.10	178875 (+140.%)	2330.4	4130819
v.10	306169	271.1	v.10-v.11	19687 (+6.43%)	406.1	1420405
v.11	325856	420.6	v.11-v.12	7469 (+2.29%)	302.3	376095
v.12	333325	461.7	v.12-v.13	11224 (+3.37%)	209.1	431405
v.13	344549	511.1	v.13-v.14	8112 (+2.35%)	390.8	742571
v.14	352661	602.7	v.14-v.15	4474 (+1.27%)	153.1	292081
v.15	357137	615.6	v.15-v.16	4689 (+1.31%)	149.3	276475
v.16	361824	637.4	v.16-full	17867 (+4.94%)	261.8	398477

Table 18 Direct incremental classification of SNOMED releases using CEL. Time in seconds.

% have actually improved. In particular, the incremental classification time between v.16 of SNOMED and full SNOMED which involves the addition of 4.94% new axioms is only 261.8 seconds as opposed to 941.1 seconds for changes involving 5% of random axioms. This confirms our hypothesis that the practical performance of incremental reasoning algorithms should be better than the performance obtained for tests involving random modifications.

6.4 Comparison of the Proposed Techniques

As previously discussed, it is difficult to directly compare the two incremental classification techniques presented in this paper because they make different assumptions on the ontology languages and have different application scenarios. The goal of this section is to discuss possible trade-offs for each method and obtain general guidelines for their applicability in practical scenarios. To this end, we will use the SNOMED ontology, for which incremental reasoning seems especially useful.

Since Pellet currently fails to classify SNOMED using our hardware setup, we have re-implemented the module-based approach described in Section 4 to use CEL instead. In Table 19 we present the results of the experiment with SNOMED as described in Section 6.2. The results correlate with those presented in Table 14 for $\mathcal{EL}^+$ ontologies, namely, the incremental classification is considerably faster than the full classification.

We have also performed the experiments with the historical versions of SNOMED as described in Section 6.3, but using the module-based approach instead. The results are given in Table 20. These results, however, cannot be directly compared with those in Table 18 since it is unlikely that users will modify hundreds or thousands axioms in one session. As we can see, the results are not very promising and incremental classification is often slower than the full classification.

	n	1: ♯ Mod. Affected (Av / Mx)	2: ♯ Axioms in Aff. Mod. (Av / Mx)	3: Update Aff. Mod. (Av / Mx)	4: Re-class. Aff. Mod. (Av / Mx)	5. Total Time (Av / Mx)
SNOMED	1	5/9	62/84	44.1/51.6	13.2/26.2	57.3/77.8
SNOMED	2	6.2/11	53.3/141	45.5/60.8	8.4/24.0	53.9/84.8
SNOMED	4	12.3/14	31.3/84	70.9/89.1	20.1/33.5	91.0/122.6

Table 19 Module-based incremental classification using CEL. Time in seconds.

Release Transition	♯ New Atomic Concepts	♯ Aff. Mod. (Avg)	♯ Axioms Aff. Mod. (Av / Mx)	Module Extraction Time	Module Class. Time	Total Time
v.1-v.2	2611	10450	11.5/49	53.9	129.1	183.0
v.2-v.3	597	1520	10.1/51	14.3	10.3	24.6
v.3-v.4	668	1574	11.4/53	15.3	15.6	30.9
v.4-v.5	4107	6974	7.4/54	57.9	49.8	107.7
v.5-v.6	3910	30091	14.9/60	137.1	212.3	349.5
v.6-v.7	18229	91235	16.3/104	324.4	334.1	658.5
v.7-v.8	5	5	1/1	1.9	2.4	4.3
v.8-v.9	4443	6150	6.4/70	69.8	63.3	133.1
v.9-v.10	178875	261231	21.1/178	2956.6	1009.3	3965.9
v.10-v.11	19687	260612	28.0/206	1402.1	852.7	2254.9
v.11-v.12	7469	56643	33.4/206	541.1	155.6	696.6
v.12-v.13	11224	68381	38.3/214	735.1	234.1	969.2
v.13-v.14	8112	55708	53.1/237	742.5	1624.5	2637.3
v.14-v.15	4474	150318	42.9/241	1158.1	558.62	1716.8
v.15-v.16	4689	200800	39.4/249	mem out	-	-
v.16-full	17867	65070	37.3/253	868.4	214.7	1083.1

Table 20 Incremental classification of SNOMED releases using modules in CEL. Time in seconds.

To summarize, we have evaluated the two incremental reasoning techniques presented in this paper over a set of real-world ontologies. The following main conclusions can be drawn from the experiments:

- Both techniques provide substantial benefits in the case of large ontologies that are "loosely inter-connected".
- The module-based approach is especially useful when only a few (up to 10) axioms have been modified before re-classification. This technique is thus better suited for manual ontology editing, since users rarely modify a large number of axioms in one session.
- The direct approach is especially useful with "batch" updates involving a large number of axioms, e.g., when importing a new ontology into the current one, or when updating the ontology from an external repository to a new release. This technique, however, assumes that no axiom has been deleted from the ontology, and it is most useful when the number of the new axioms does not exceed 2% of the size of the ontology.

It might be possible to combine both techniques to deal with more complex situations; however, we leave this investigation for future work. It is also worth pointing out that, in contrast to ontology reasoners, the implementations of our incremental

reasoning procedures are rather prototypical and not heavily optimized. Further optimizations could provide significant performance improvements.

7 Related Work

While there has been substantial work on optimizing reasoning services for DLs (see [5] for an overview), the topic of reasoning through evolving DL knowledge bases remains relatively unaddressed. Notable exceptions include [18–20,31]; these papers, however, investigate the problem of incremental reasoning using model-caching techniques in application scenarios that involve changes *only* in the ABox.

There has been substantial work on incremental query and view maintenance in databases (e.g., [6,38,40]) and rule-based systems (e.g., Datalog [13,14]). While related, our work addresses a more expressive formalism. Furthermore the problem of incremental maintenance in database systems has been mostly considered with respect to changes in the data, (corresponding to a DL ABox) and not with respect to the database schema (corresponding to a DL TBox). Our technique, however, focuses on schema reasoning.

There is also extensive work on Truth Maintenance Systems (TMSs) for logical theories (e.g., [12,16]). As pointed out in Section 3, a justification-based approach would be advantageous for incremental classification only if the number of positive subsumptions was larger than the number of non-subsumptions; that is, if most of the formulas the justifications keep track of were provable. This is, however, not the case, as typically there are far more non-subsumptions than subsumptions. Additionally, a TMS system designed to support non-subsumptions (e.g., by caching models) would most likely be impractical due to the potentially large size of these models and substantial modifications likely to be caused by changes in general axioms; however, in our approach, maintaining locality-based modules introduces limited overhead. Finally, the representation language in practical TMSs is mostly propositional logic, whereas we focus on much more expressive languages.

8 Conclusions and Future Works

In this paper, we have proposed two techniques for incremental classification of ontologies. The first technique is based on a notion of locality-based modules and can be used in combination with any reasoner. This technique can be applied for arbitrary modifications to any ontology. In contrast, the second technique requires the modification of the reasoner and is only applicable to $\mathcal{EL}^+$ ontologies and the additions of new axioms. We have implemented both techniques and evaluated them on a range of commonly used ontologies. Our experiments demonstrate that both technique can help to improve the classification time in a number of practically relevant scenarios, and are especially useful for large and simple ontologies like NCI and SNOMED.

There are several possible directions for the future research. The module-based procedure could possibly be extended for incremental ABox query answering. Locality based modules as considered in this paper are not well-suited for ABox reasoning

because ABox assertions are not local. Other types of modules considered in [10] can be used instead. Also, the variants of locality-based modules proposed in [34] could be used to reduce the size of the extracted modules and it would be interesting to explore their suitability for incremental reasoning. Finally, the direct incremental reasoning procedure for $\mathcal{EL}^+$ can possibly be extended for deletions using axiom tracing techniques for keeping track of the inferences. Whenever an axiom is deleted, the procedure could also delete all axioms that have been derived from it.

Acknowledgements: This work has been supported by the European Union 6th Framework Program project TONES (Thinking ONtologieES, ref:IST-007603) and by the British Engineering and Physical Sciences Research Council projects RINO (Reasoning Infrastructure for Ontologies and Instances) and ConDOR (Consequence-Driven Ontology Reasoning). The first author is also supported by a Royal Society University Research Fellowship.

References

1. F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence IJCAI-05*, Edinburgh, UK, 2005. Morgan-Kaufmann Publishers.
2. F. Baader, C. Lutz, and B. Suntisrivaraporn. Is tractable reasoning in extensions of the description logic el useful in practice? In *Proceedings of the 2005 International Workshop on Methods for Modalities (M4M-05)*, 2005.
3. F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL - a polynomial-time reasoner for life science ontologies. In *IJCAR 2006, Proceedings of the Third International Joint Conference on Automated Reasoning*, volume 4130 of *Lecture Notes in Computer Science*, pages 287–291. Springer, 2006.
4. F. Baader, C. Lutz, and B. Suntisrivaraporn. Efficient reasoning in $\mathcal{EL}+$. In *Proceedings of the 19th International Workshop on Description Logics (DL-06)*, volume 189 of *CEUR Workshop Proceedings*, 2006.
5. F. Baader and W. Nutt. Basic description logics. In F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 43–95. Cambridge University Press, 2003.
6. J. A. Blakeley, P.-A. Larson, and F. W. Tompa. Efficiently updating materialized views. In *Proc. of SIGMOD '86: ACM SIGMOD International Conference on Management of Data*, pages 61–71, 1986.
7. B. Cuenca Grau, C. Halaschek-Wiener, and Y. Kazakov. History matters: Incremental ontology reasoning using modules. In *6th International Semantic Web Conference (ISWC)*, volume 4825 of *Lecture Notes in Computer Science*, pages 183–196. Springer, 2007.
8. B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Just the right amount: Extracting modules from ontologies. In *Proc. of the 16th International World Wide Web Conference (WWW2007)*, 2007.
9. B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. A logical framework for modularity of ontologies. In *Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*, pages 298–303. AAAI Press, 2007.
10. B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *Journal of Artificial Intelligence Research (JAIR)*, 31:273–318, 2008.
11. B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.
12. J. de Kleer. An assumption-based TMS. *Artificial Intelligence.*, 28(2):127–162, 1986.
13. G. Dong, J. Su, and R. W. Topor. Nonrecursive incremental evaluation of datalog queries. *Annals of Mathematics and Artificial Intelligence*, 14(2-4), 1995.
14. G. Dong and R. W. Topor. Incremental evaluation of datalog queries. In *Proc. of the 4th Int. Conference on Database Theory*, 1992.
15. W. Dowling and J. Gallier. Linear-time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming*, 3(1):267–284, 1984.

16. J. Doyle. A truth maintenance system. *Readings in nonmonotonic reasoning*, pages 259–279, 1987.
17. V. Haarslev and R. Moeller. Racer system description. In *Proceedings of the First International Joint Conference on Automated Reasoning (IJCAR 2001). Volume 2083 of Lecture Notes in Artificial Intelligence, pages 701-705*, 2001.
18. V. Haarslev and R. Möller. Incremental query answering for implementing document retrieval services. In *Proc. of DL-2003*, pages 85–94, 2003.
19. C. Halaschek-Wiener and J. Hendler. Toward expressive syndication on the web. In *Proc. of the 16th International World Wide Web Conference (WWW 2007)*, 2007.
20. C. Halaschek-Wiener, B. Parsia, and E. Sirin. Description logic reasoning with syntactic updates. In *Proc. of ODBase2006*, 2006.
21. I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and rdf to owl: the making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.
22. U. Hustadt, B. Motik, and U. Sattler. Deciding Expressive Description Logics in the Framework of Resolution. *Information & Computation*, 206(5):579–601, 2008.
23. A. Kalyanpur, B. Parsia, M. Horridge, and E. Sirin. Finding all justifications of owl dl entailments. In *6th International Semantic Web Conference*, volume 4825 of *Lecture Notes in Computer Science*, pages 267–280. Springer, 2007.
24. Y. Kazakov. $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In *Proc. of the Eleventh International Conference on the Principles of Knowledge Representation and Reasoning (KR-2008)*, pages 274–284, 2008.
25. O. Kutz, I. Horrocks, and U. Sattler. The Even More Irresistible $\mathcal{SROIQ}$. In P. Doherty, J. Mylopoulos, and C. A. Welty, editors, *Proceedings of the 10th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 68–78, Lake District, UK, June 2–5 2006. AAAI Press.
26. C. Lutz, D. Walther, and F. Wolter. Conservative extensions in expressive description logics. In *Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, 2007.
27. C. Lutz and F. Wolter. Conservative extensions in the lightweight description logic $\mathcal{EL}$. In F. Pfenning, editor, *Proceedings of the 21th Conference on Automated Deduction (CADE-21)*, volume 4603 of *Lecture Notes in Artificial Intelligence*, pages 84–99. Springer-Verlag, 2007.
28. B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. L. (Editors). OWL 2 Web Ontology Language Profiles. *W3C Recommendation*, 2009.
29. B. Motik, P. F. Patel-Schneider, and B. P. (Editors). OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax. *W3C Recommendation*, 2009.
30. B. Motik, R. Shearer, and I. Horrocks. Optimized Reasoning in Description Logics using Hypertableaux. In *Proc. of the 21st Conference on Automated Deduction (CADE-21)*, volume 4603 of *LNAI*, pages 67–83, Bremen, Germany, July 17–20 2007. Springer.
31. B. Parsia, C. Halaschek-Wiener, and E. Sirin. Towards incremental reasoning through updates in OWL-DL. In *Reasoning on the Web Workshop*, 2006.
32. B. Parsia, E. Sirin, and A. Kalyanpur. Debugging OWL ontologies. In *Proceedings of the 14th International World Wide Web Conference (WWW 2005)*, pages 633–640, 2005.
33. P. Patel-Schneider, P. Hayes, and I. Horrocks. Web ontology language OWL Abstract Syntax and Semantics. *W3C Recommendation*, 2004.
34. U. Sattler, T. Schneider, and M. Zakharyaschev. Which kind of module should i extract? In *Proc. of the 22nd International Workshop on Description Logics (DL 2009)*, volume 477 of *CEUR Workshop Proceedings*, 2009.
35. S. Schlobach and R. Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In *Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI, 2003)*, pages 355–362. Morgan Kaufmann, 2003.
36. S. Schlobach, Z. Huang, R. Cornet, and F. van Harmelen. Debugging incoherent terminologies. *Journal Automated Reasoning*, 39(3):317–349, 2007.
37. E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5(2):51–53, 2007.
38. M. Stonebraker. Implementation of integrity constraints and views by query modification. In *SIGMOD '75: Proc. of the 1975 ACM SIGMOD international conference on Management of data*, pages 65–78, New York, NY, USA, 1975.
39. B. Suntisrivaraporn. Module extraction and incremental classification: A pragmatic approach for ontologies. In *5th European Semantic Web Conference (ESWC)*, volume 5021 of *Lecture Notes in Computer Science*, pages 230–244. Springer, 2008.

40. D. B. Terry, D. Goldberg, D. Nichols, and B. M. Oki. Continuous queries over append-only databases. In *Proceedings of the International Conference on Management of Data*, 1992.
41. D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proc. of the Third International Joint Conference on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 292–297. Springer, 2006.