Zooming in on Ontologies: Minimal Modules and Best Excerpts*

Jieying Chen,[1] Michel Ludwig, Yue Ma[1] and Dirk Walther

[1] Laboratoire de Recherche en Informatique, Université Paris-Sud, France
{jieying.chen,yue.ma}@lri.fr, {michel.ludwig,dirkww}@gmail.com

Abstract. Ensuring access to the most relevant knowledge contained in large ontologies has been identified as an important challenge. To this end, minimal modules (sub-ontologies that preserve all entailments over a given vocabulary) and excerpts (certain, small number of axioms that best capture the knowledge regarding the vocabulary by allowing for a degree of semantic loss) have been proposed. In this paper, we introduce the notion of subsumption justification as an extension of justification (a minimal set of axioms needed to preserve a logical consequence) to capture the subsumption knowledge between a term and all other terms in the vocabulary. We present algorithms for computing subsumption justifications based on a simulation notion developed for the problem of deciding the logical difference between ontologies. We show how subsumption justifications can be used to obtain minimal modules and to compute best excerpts by additionally employing a partial Max-SAT solver. This yields two state-of-the-art methods for computing all minimal modules and all best excerpts, which we evaluate over large biomedical ontologies.

1 Introduction

Knowledge about a complex system represented in ontologies yields a collection of axioms that are too large for human users to browse, let alone to comprehend or reason about it. In this paper, we propose a computational framework to zoom in on large ontologies by providing users with either the necessary axioms that act as explanations for sets of entailments, or fix-sized sub-ontologies containing the most relevant information over a vocabulary.

Various approaches to extracting knowledge from ontologies have been suggested including ontology summarization [23,25,29], ontology modularization [9, 14,26–28], ontology decomposition [6,20], and consequence justifications [3,11]. Existing ontology summarization systems focus on producing an abridged version of RDF/S ontologies by identifying the most important nodes and their links under certain numeric measures, e.g., in/out degree centrality of a node [25]. In contrast, ontology modularization and decomposition developed for Description Logics (DLs) [2] is to identify ontological axioms needed to define the relationships between concept and role names contained in a given signature. Modules are sub-ontologies that preserve all logical consequences over a given signature,

* This work is partially funded by the ANR project GoAsQ (ANR-15-CE23-0022).

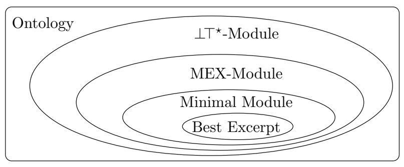

Fig. 1: Zooming in on an ontology

$$\begin{aligned}
\mathit{Decreased\_blood\_volume} &\sqsubseteq \mathit{Cardiovascular\_finding} && (1)\\
\mathit{Cardiovascular\_shunt} &\sqsubseteq \mathit{Cardiovascular\_finding} && (2)\\
\mathit{Cardiac\_shunt} &\sqsubseteq \mathit{Cardiovascular\_shunt} \sqcap && (3)\\
&\quad \exists RG.(\exists FS.\mathit{Heart\_structure}) &&\\
\mathit{Cardiovascular\_structure} &\sqsubseteq \mathit{Body\_system\_structure} && (4)
\end{aligned}$$

Fig. 2: Example axioms in Snomed CT (FS for $\mathit{Finding\_site}$, RG for $\mathit{Role\_Group}$)

and ontology decomposition partitions an ontology into atoms that are never split by different modules. Computing minimal modules is known to be hard. Hence, existing systems are either restricted to tractable DLs [5, 13, 15] or they compute approximations of minimal modules [6, 9, 21]. This has resulted in two important module notions: the semantics-based modules computed by the system MEX [13] and the syntactic locality-based $\bot\top^\star$-modules [22]. Figure 1 shows the set inclusion relationship between these notions, where we focus on MEX-modules, minimal modules and best excerpts (see below) in this paper. A justification for a particular logical consequence is a minimal set of axioms that preserve the entailment. Although computing all justifications is generally a hard task, different approaches have been shown promising for this task [1, 10, 17, 30].

Different module notions and justifications share the property that the number of the axioms they contain is not bounded (besides the size of the entire ontology). Even minimal modules for small signatures may be large rendering human understanding more difficult. To this end, the notion of best excerpts [4] has been introduced as size-bounded subsets of ontologies that preserve as much knowledge about a given signature as possible.

The following real-world example illustrates possible benefits of best excerpts. Suppose a user is concerned with the cardiovascular disease defined in the Snomed CT[1] ontology $\mathcal{T}$ consisting of 317 891 axioms. The user then selects the terms *Cardiovascular_finding*, *Decreased_blood_volume* and *Cardiac_shunt* from $\mathcal{T}$ as her signature Σ of interest. To help the user zoom in on $\mathcal{T}$ for Σ, we can extract, for instance, the $\bot\top^\star$-module and obtain 51 axioms, or the *smallest* minimal modules, which yields a further reduction down to 15 axioms,[2] among which are the axioms given in Figure 2. Arguably our user still feels overwhelmed by the amount of 15 axioms. This is where the notion of best k-excerpt steps in. By setting $k = 3$, the user can get a best 3-excerpt $\mathcal{E}_1$ consisting of the axioms 1–3 listed above. By zooming in further, say extracting one-sized excerpts, she obtains $\mathcal{E}_2$ consisting of the first axiom. As a best excerpt, $\mathcal{E}_1$ guarantees all logical entailments over the terms *Cardiac_shunt* and *Decreased_blood_volume*. And the singleton $\mathcal{E}_2$ keeps the complete information over the term *Decreased_blood_volume*. Note that $\mathcal{E}_2$ is returned due to the fact that it needs more than two axioms to preserve the full information for any other

[1] `http://www.ihtsdo.org/snomed-ct`

[2] Refer to `https://goo.gl/o1QFGm` for the whole list and for cases where larger minimal modules appear in practice.

concept in Σ. Moreover, axiom 4 is in $\mathcal{M}$ but missing in $\mathcal{E}_1$ and in $\mathcal{E}_2$. This is because they merely serve to provide background knowledge for reasoning over, thus not directly linked to, the user's input terms Σ, which are excluded from best excerpts due to the size restriction. In this way, the user gains control over a large ontology. An approximate approach to computing ontology excerpts based on information retrieval was introduced in [4]. However, it cannot guarantee to compute the best excerpt.

In this paper, we generalise the notion of a justification to *subsumption justification* as a minimal set of axioms needed to define the relationship between a selected term to the remaining terms in a given vocabulary. Inspired by a proof-theoretic solution to the logical difference problem between ontologies [7,18], we develop recursive algorithms to compute subsumption justifications. A minimal module preserving the knowledge about a vocabulary can now be characterised as the union of subsumption justifications, one for each term in the vocabulary. By taking the union of subsumption justification for as many terms as possible without exceeding a given size limitation yields a best excerpt. The algorithm operates in two stages: First, for every term in the vocabulary, all subsumption justifications are computed. Similarly to modules, no bound on the size of such justifications exists. Second, minimal modules are obtained by taking the union of one subsumption justification for every term, and best k-excerpts, for $k > 0$, are obtained by packing a subsumption justification for as many terms as possible into a space of at most k axioms. The latter is solved via an encoding into a partial Max-SAT problem [8]. Note that [4] only considers excerpts based on information retrieval, which provide an approximate solution that can be computed rather quickly, albeit not capturing the knowledge in an optimal way. In this paper, however, we provide an algorithm for computing best excerpts via subsumption justifications. Best excerpts can be used as a benchmark to evaluate the quality of other excerpt or incomplete module notions.

Our contribution is three-fold: (i) We define the notion of subsumption justification and then introduce two of its applications (Section 3): computing minimal modules and best excerpts; (ii) moreover, we present algorithms of computing subsumption justifications (Section 4); (iii) finally, we evaluate the performance of overall algorithms (Section 5). Our algorithm for computing minimal modules outperformed the search-based approach from [5], and as the first best excerpt extraction algorithm, we can obtain the excerpts of a better quality than the excerpts based on information retrieval [4].

2 Preliminaries

Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be mutually disjoint (countably infinite) sets of concept names and role names. We use A, B, X, Y, Z to denote concept names, and r, s for role names. The set of *$\mathcal{ELH}$-concepts* C and the set of *$\mathcal{ELH}$-inclusions* α are built by the following grammar rules: $C ::= \top \mid A \mid C \sqcap C \mid \exists r.C$, $\alpha ::= C \sqsubseteq C \mid C \equiv C \mid r \sqsubseteq s$, where $A \in \mathsf{N_C}$ and $r, s \in \mathsf{N_R}$. An *$\mathcal{ELH}$-TBox* is a finite set of $\mathcal{ELH}$-inclusions (also called axioms).

The semantics is defined using interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the domain $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function mapping each concept name A

to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and every role name r to a binary relation $r^{\mathcal{I}}$ over $\Delta^{\mathcal{I}}$. The *extension* $C^{\mathcal{I}}$ of a possibly complex concept C is defined inductively as: $(\top)^{\mathcal{I}} := \Delta^{\mathcal{I}}$, $(C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}$, and $(\exists r.C)^{\mathcal{I}} := \{x \in \Delta^{\mathcal{I}} \mid \exists y \in C^{\mathcal{I}} : (x, y) \in r^{\mathcal{I}}\}$.

An interpretation $\mathcal{I}$ *satisfies* a concept C, an axiom $C \sqsubseteq D$, $C \equiv D$, or $r \sqsubseteq s$ iff $C^{\mathcal{I}} \neq \emptyset$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $C^{\mathcal{I}} = D^{\mathcal{I}}$, or $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$, respectively. An interpretation $\mathcal{I}$ is a *model* of $\mathcal{T}$ if $\mathcal{I}$ satisfies all axioms in $\mathcal{T}$. An axiom α *follows* from $\mathcal{T}$, written $\mathcal{T} \models \alpha$, if for all models $\mathcal{I}$ of $\mathcal{T}$, it holds that $\mathcal{I}$ satisfies α.

An $\mathcal{ELH}$-terminology $\mathcal{T}$ is an $\mathcal{ELH}$-TBox consisting of axioms of the form $A \sqsubseteq C$, $A \equiv C$, $r \sqsubseteq s$, where A is a concept name, r and s are role names, C is an $\mathcal{ELH}$-concept and no concept name A occurs more than once on the left-hand side of an axiom of the form $A \equiv C$. To simplify the presentation we assume that terminologies do not contain any occurrence of $\top$ and no axioms of the form $A \equiv B$ (after having removed multiple B-conjuncts) for concept names A and B. Note that the material presented in the paper can easily be extended to take $\top$ into account. A terminology is said to be *acyclic* iff it can be unfolded (i.e., the process of substituting each concept name A by the right-hand side C of its defining axiom $A \equiv C$ terminates).

We say that a concept name A is *conjunctive in* $\mathcal{T}$ iff there exist concept names $B_1, \ldots, B_n$, $n > 0$, such that $A \equiv B_1 \sqcap \ldots \sqcap B_n \in \mathcal{T}$; otherwise A is said to be *non-conjunctive in* $\mathcal{T}$. An $\mathcal{ELH}$-terminology $\mathcal{T}$ is *normalised* iff it only contains axioms of the forms $A \sqsubseteq B_1 \sqcap \ldots \sqcap B_m$, $A \equiv B_1 \sqcap \ldots \sqcap B_n$, $A \sqsubseteq \exists r.B$ and $A \equiv \exists r.B$, where $m \geq 1$, $n \geq 2$, A, B, B_i are concept names, and each conjunct B_i is non-conjunctive in $\mathcal{T}$. Every $\mathcal{ELH}$-terminology $\mathcal{T}$ can be normalised in polynomial time into a terminology $\mathcal{T}'$ such that for all $\mathcal{ELH}$-inclusions α formulated using concept and role names from $\mathcal{T}$ only, it holds that $\mathcal{T} \models \alpha$ iff $\mathcal{T}' \models \alpha$. Note that each axiom $\alpha \in \mathcal{T}$ is transformed individually into a set of normalised axioms. Moreover, we assume that when $\mathcal{T}$ is normalised, a denormalisation function $\delta_{\mathcal{T}} \colon \mathcal{T}' \to 2^{\mathcal{T}}$ is computed that maps every normalised axiom $\beta \in \mathcal{T}'$ to a set of axioms $\delta_{\mathcal{T}}(\alpha) \subseteq \mathcal{T}$ that consists of all axioms $\alpha \in \mathcal{T}$ that generated β during their normalisation.

We denote the number of axioms in a TBox $\mathcal{T}$ with $|\mathcal{T}|$. A signature Σ is a finite subset of $\mathsf{N_C} \cup \mathsf{N_R}$. For a syntactic object χ (i.e., a concept, an axiom, or a TBox), $\mathrm{sig}(\chi)$ is the set of concept and role names occurring in χ. We denote with $\mathrm{sig}^{\mathsf{N_C}}(\chi)$ the set of concept names in $\mathrm{sig}(\chi)$. We write $\mathcal{ELH}_{\Sigma}$ to denote the set of $\mathcal{ELH}$-concepts C such that $\mathrm{sig}(C) \subseteq \Sigma$. A subset $M \subseteq \mathcal{T}$ is called a *justification for an $\mathcal{ELH}$-concept inclusion α from $\mathcal{T}$* iff $M \models \alpha$ and $M' \not\models \alpha$ for every $M' \subsetneq M$. We denote the set of all justifications for an $\mathcal{ELH}$-concept inclusion α from an $\mathcal{ELH}$-terminology $\mathcal{T}$ with $\mathrm{Just}_{\mathcal{T}}(\alpha)$. Note that $\mathrm{Just}_{\mathcal{T}}(\alpha)$ may contain exponentially many justifications in the number of axioms in $\mathcal{T}$.

The logical difference between two $\mathcal{ELH}$-terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$, denoted as $\mathsf{cDiff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$, is the set of all $\mathcal{ELH}$-inclusions α of the form $C \sqsubseteq D$ for $\mathcal{ELH}$-concepts C and D such that $\mathrm{sig}(\alpha) \subseteq \Sigma$, $\mathcal{T}_1 \models \alpha$, and $\mathcal{T}_2 \not\models \alpha$.

If two terminologies are logically different, the set $\mathsf{cDiff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$ consists of infinitely many concept inclusions. The *primitive witnesses theorems* from [12] allow us to consider only certain inclusions of a simpler syntactic form. It states that if $\alpha \in \mathsf{cDiff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$, where $\mathcal{T}_1$ and $\mathcal{T}_2$ are $\mathcal{ELH}$-terminologies and Σ a signature, then either $A \sqsubseteq D$ or $C \sqsubseteq A$ is a member of $\mathsf{cDiff}_{\Sigma}(\mathcal{T}_1, \mathcal{T}_2)$, where

$A \in \mathrm{sig}^{\mathsf{N_C}}(\alpha)$ and C, D are $\mathcal{ELH}$-concepts occurring in α. We call such concepts A *witnesses* and denote the set of witnesses with $\mathsf{cWtn}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. It holds that $\mathsf{cWtn}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$ iff $\mathsf{cDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$.

A k-excerpt of $\mathcal{T}$ w.r.t. Σ is a subset $\mathcal{E}$ of $\mathcal{T}$ such that $\mid \mathcal{E} \mid \leq k$. Let μ be an incompleteness measure, we say a k-excerpt $\mathcal{E}$ is the best excerpt of $\mathcal{T}$ w.r.t. Σ if $\mu(\mathcal{T}, \Sigma, \mathcal{E}) = \min\{\, \mu(\mathcal{T}, \Sigma, \mathcal{E}') \mid \mathcal{E}'$ is a k-excerpt of $\mathcal{T}\,\}$. In this paper, we use the size of concept witness $\mathsf{cWtn}_\Sigma(\mathcal{T}, \mathcal{E})$ as the incompleteness measure.

3 Application of Subsumption Justification

In this section, we introduce the notion of subsumption justification, and give two applications of this notion. The algorithms for computing subsumption justifications are given separately in Section 4.

We assume that $\mathcal{T}$, $\mathcal{T}_1$, and $\mathcal{T}_2$ are acyclic normalised $\mathcal{ELH}$-terminologies, Σ is a signature, $X \in \mathsf{N_C}$ is concept names.

Definition 1. *We say that* $\mathcal{M} \subseteq \mathcal{T}$ *is an* $\langle X, \Sigma\rangle$-subsumee module of $\mathcal{T}$ *iff for every* $C \in \mathcal{ELH}_\Sigma$, $\mathcal{T} \models C \sqsubseteq X$ *implies* $\mathcal{M} \models C \sqsubseteq X$. *Similarly, we define the notion of an* $\langle X, \Sigma\rangle$-subsumer module $\mathcal{M}$ of $\mathcal{T}$ *to be a subset of* $\mathcal{T}$ *such that for every* $D \in \mathcal{ELH}_\Sigma$, $\mathcal{T} \models X \sqsubseteq D$ *implies* $\mathcal{M} \models X \sqsubseteq D$.

Additionally, a set $\mathcal{M}$ *is called an* $\langle X, \Sigma\rangle$-subsumption module of $\mathcal{T}$ *iff* $\mathcal{M}$ *is an* $\langle X, \Sigma\rangle$*-subsumee and* $\langle X, \Sigma\rangle$*-subsumer module of* $\mathcal{T}$*. An* $\langle X, \Sigma\rangle$-subsumee (resp. subsumer, subsumption) justification *is an* $\langle X, \Sigma\rangle$*-subsumee (resp. subsumer, subsumption) module of* $\mathcal{T}$ *that is minimal w.r.t.* $\subsetneq$.

We denote the set of all $\langle X, \Sigma\rangle$-subsumee (resp. subsumer, subsumption) justifications as $\mathcal{J}^{\leftarrow}_{\mathcal{T}}(X, \Sigma)$ (resp. $\mathcal{J}^{\rightarrow}_{\mathcal{T}}(X, \Sigma), \mathcal{J}_{\mathcal{T}}(X, \Sigma)$). Note that there may exist multiple $\langle X, \Sigma\rangle$- (subsumer, subsumee) subsumption justifications.

Example 1. Let $\Sigma = \{A_1, A_2, B\}$ and let $\mathcal{T} = \{\, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7\}$, where $\alpha_1 = X \equiv Y \sqcap Z$, $\alpha_2 = Y \sqsubseteq B$, $\alpha_3 = Z \equiv Z_1 \sqcap Z_2$, $\alpha_4 = A_1 \sqsubseteq Y$, $\alpha_5 = A_2 \sqsubseteq Z$, $\alpha_6 = A_2 \sqsubseteq Z_1$, and $\alpha_7 = A_2 \sqsubseteq Z_2$. Then the sets $\mathcal{M}_1 = \{\alpha_1,\, \alpha_3,\, \alpha_4,\, \alpha_6,\, \alpha_7\}$, $\mathcal{M}_2 = \{\alpha_1,\, \alpha_4,\, \alpha_5\}$, and $\mathcal{T}$ are all $\langle X, \Sigma\rangle$-subsumee modules of $\mathcal{T}$, whereas only $\mathcal{M}_1$ and $\mathcal{M}_2$ are $\langle X, \Sigma\rangle$-subsumee justifications of $\mathcal{T}$. The set $\mathcal{M}_3 = \{\alpha_1,\, \alpha_2\}$ is an $\langle X, \Sigma\rangle$-subsumer justification of $\mathcal{T}$. Finally, the sets $\mathcal{M}_1 \cup \mathcal{M}_3$ and $\mathcal{M}_2 \cup \mathcal{M}_3$ are $\langle X, \Sigma\rangle$-subsumption justifications of $\mathcal{T}$.

Proposition 1. $\mathcal{M}$ *is an* $\langle X, \Sigma\rangle$*-subsumption module of* $\mathcal{T}$ *iff* $X \notin \mathsf{cWtn}_\Sigma(\mathcal{T}, \mathcal{M})$.

Proposition 1 follows from the *primitive witnesses theorems* [12] and Definition 1.

3.1 Application 1: Computing Minimal Modules

A module is a subset of an ontology that can act as a substitute for the ontology w.r.t. a given signature. In this paper, we consider the notion of basic modules from [5] for acyclic $\mathcal{ELH}$-terminologies.

Definition 2 (Basic Module [5]). *Let $\mathcal{T}$ be an $\mathcal{ELH}$-terminology, and let Σ be a signature. A subset $\mathcal{M} \subseteq \mathcal{T}$ is called a* basic $\mathcal{ELH}$-module of $\mathcal{T}$ w.r.t. Σ *iff* $\mathsf{cDiff}_{\Sigma}(\mathcal{T}, \mathcal{M}) = \emptyset$.

To apply subsumption justifications for computing all modules that are minimal w.r.t. $\subsetneq$, we define the operator $\otimes$ to combine subsumption justifications of $\mathcal{T}$ for all Σ-concept names, as follows: Given a set S and $\mathbb{S}_1, \mathbb{S}_2 \subseteq 2^S$, $\mathbb{S}_1 \otimes \mathbb{S}_2 := \{ S_1 \cup S_2 \mid S_1 \in \mathbb{S}_1,\ S_2 \in \mathbb{S}_2 \}$. For instance, if $\mathbb{S}_1 = \{\{\alpha_1, \alpha_2\}, \{\alpha_3\}\}$ and $\mathbb{S}_2 = \{\{\alpha_2, \alpha_3\}, \{\alpha_4, \alpha_5\}\}$, then $\mathbb{S}_1 \otimes \mathbb{S}_2 = \{\{\alpha_1, \alpha_2, \alpha_3\}, \{\alpha_1, \alpha_2, \alpha_4, \alpha_5\}, \{\alpha_2, \alpha_3\}, \{\alpha_3, \alpha_4, \alpha_5\}\}$. Note that the $\otimes$ operator is associative and commutative.

For a set $\mathbb{M}$ of sets, we define a function $\text{Minimise}_{\subseteq}(\mathbb{M})$ as follows: $\mathcal{M} \in \text{Minimise}_{\subseteq}(\mathbb{M})$ iff. $\mathcal{M} \in \mathbb{M}$ and there does not exist a set $\mathcal{M}' \in \mathbb{M}$ such that $\mathcal{M}' \subsetneq \mathcal{M}$. Finally, we can use the $\otimes$ operator and the $\text{Minimise}_{\subseteq}(\mathbb{M})$ function to combine sets of subsumer and sets of subsumee modules to obtain a set of subsumption modules, whose correctness is guaranteed by Proposition 1.

Theorem 1. *Let $\mathbb{M}_{\Sigma}^{\mathcal{T}}$ be the set of all minimal basic $\mathcal{ELH}$-modules of $\mathcal{T}$ w.r.t. Σ. Then* $\mathbb{M}_{\Sigma}^{\mathcal{T}} := \text{Minimise}_{\subseteq}(\otimes_{X \in \Sigma \cap \mathsf{N_C}}(\mathcal{J}_X^{\rightarrow}(X, \Sigma) \otimes \mathcal{J}_X^{\leftarrow}(X, \Sigma)))$.

Please note that, given a TBox and a signature, MEX-module is unique [14] but there may exist exponential many minimal basic modules in theory. A relation between basic module and MEX module is given below:

Proposition 2. *Let $\mathcal{M}$ be the MEX-module of $\mathcal{T}$ w.r.t. Σ. It holds that for every minimal basic $\mathcal{EL}$-module $\mathcal{M}'$ of $\mathcal{T}$ w.r.t. Σ, $\mathcal{M}' \subseteq \mathcal{M}$.*

Intuitively, Proposition 1 follows from the fact that MEX-modules are based on a semantic inseparability notion [14], whereas the notion of basic modules uses a weaker, deductive inseparability notion based on $\mathcal{EL}$-inclusions [5]; see, e.g., [19] for more on inseparability.

3.2 Application 2: Computing Best Excerpts

Based on subsumption justifications, in this section, we present an encoding of the best k-excerpt problem in a partial Max-SAT problem, with the aim of delegating the task of finding the best excerpt to a Max-SAT solver. In that way we can leverage the decades of research efforts dedicated to developing efficient SAT solvers for our problem setting. We continue with reviewing basic notions relating to propositional logic and Max-SAT.

Partial Max-SAT is an extension of the Boolean Satisfiability (SAT) to optimization problems. Formally, a partial Max-SAT problem $\mathcal{P}$ is pair $\mathcal{P} = (H, S)$ where H and S are finite sets of clauses, called *hard* and *soft* clauses, respectively. We say that a valuation v is a solution of P iff. v satisfies all the clauses in H and there does not exist a valuation v' that satisfies all the clauses in H and $\sum_{\psi \in S} v'(\psi) > \sum_{\psi \in S} v(\psi)$.

The objective of a partial Max-SAT problem is hence to find a propositional valuation that satisfies all the hard clauses in H and that satisfies a maximal number of the soft clauses in S. Note that a partial Max-SAT problem may nevertheless admit several solutions.

We now describe of our encoding of the best k-excerpt problem into partial Max-SAT. For every axiom $\alpha \in \mathcal{T}$, we introduce a fresh propositional variable p_α. Consequently, each solution v to our partial Max-SAT problem yields a best excerpt that consists of all axioms α such that $v(p_\alpha) = 1$.

For a $\langle A, \Sigma\rangle$-subsumption justification $j \in \mathcal{J}_\mathcal{T}(A, \Sigma)$, we introduce the formula $F_j \coloneqq \bigwedge_{\alpha \in j} p_\alpha$. Consequently, F_j is valued 1 iff. p_α is valued 1, equivalently, each axiom in j is selected to be contained in a best excerpt.

For the set of $\langle A, \Sigma\rangle$-subsumption justifications $\mathcal{J} = \mathcal{J}_\mathcal{T}(A, \Sigma)$, we define $G_\mathcal{J} \coloneqq \bigvee_{j \in \mathcal{J}} F_j$. For instance, let $\mathcal{T} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$, $\mathcal{J} = \{\{\alpha_2, \alpha_3\}, \{\alpha_1, \alpha_4\}\}$, and $j = \{\alpha_2, \alpha_3\}$. Then $F_j = p_{\alpha_2} \wedge p_{\alpha_3}$ and $G_\mathcal{J} \coloneqq (p_{\alpha_2} \wedge p_{\alpha_3}) \vee (p_{\alpha_1} \wedge p_{\alpha_4})$.

Definition 3 (Encoding of the Best Excerpt Problem). *For every $A \in \Sigma$, let $\mathcal{J}_A(X, \Sigma)$ be the set of all the $\langle A, \Sigma\rangle$-subsumption justifications of a terminology $\mathcal{T}$, and let q_A be a fresh propositional variable. The* partial Max-SAT problem for finding best k-excerpts of $\mathcal{T}$ w.r.t. Σ, *denoted with $P_k(\mathcal{T}, \Sigma)$, is defined as follows. We set $P_k(\mathcal{T}, \Sigma) \coloneqq (H_k(\mathcal{T}), S_k(\mathcal{T}, \Sigma))$, where*

$$H_k(\mathcal{T}) \coloneqq \mathit{Card}(\mathcal{T}, k) \cup \bigcup_{A \in \Sigma \cap \mathsf{N_C}} \text{Clauses}(q_A \leftrightarrow G_{\mathcal{J}_A}),$$
$$S_k(\mathcal{T}, \Sigma) \coloneqq \{\, q_A \mid A \in \Sigma \cap \mathsf{N_C} \,\},$$

and $\mathit{Card}(\mathcal{T}, k)$ is the set of clauses specifying that at most k clauses from the set $\{\, p_\alpha \mid \alpha \in \mathcal{T} \,\}$ must be satisfied.

In the hard part of our partial Max-SAT problem, the clauses in $\text{Card}(\mathcal{T}, k)$ specify that the cardinality of the resulting excerpt $\mathcal{E} \subseteq \mathcal{T}$ must be equal to k. We do not fix a certain encoding that should be used to obtain $\text{Card}(\mathcal{T}, k)$, but we note that there exist several techniques that require a polynomial number of clauses in k and in the size of $\mathcal{T}$ (see e.g. [24]). Moreover, for every concept name $A \in \Sigma$, the variable q_A is set to be equivalent to the formula $G_\mathcal{J}$, i.e. q_A will be satisfied in a valuation iff the resulting excerpt will have the property that the knowledge of A w.r.t. Σ in $\mathcal{T}$ is preserved ($A \in \text{Preserved}_\Sigma(\mathcal{T}, \mathcal{E})$). Finally, the set $S_k(\mathcal{T}, \Sigma)$ of soft clauses specifies that a maximal number of q_A must be satisfied, enforcing that the resulting excerpt $\mathcal{E}$ will yield the smallest possible number of difference witnesses (whilst obeying the constraint that $|\mathcal{E}| = k$).

We can now show the correctness of our encoding, i.e. a best k-excerpt can be obtained from any solution to the partial Max-SAT problem $P_k(\mathcal{T}, \Sigma)$.

Theorem 2 (Correctness & Completeness). *Let $\mathcal{T}$ be a normalised $\mathcal{ELH}$-terminology, let Σ be a signature, and let $0 \leq k \leq |\mathcal{T}|$. It holds that $\mathcal{E} \subseteq \mathcal{T}$ is a best k-excerpt of $\mathcal{T}$ w.r.t. Σ iff there exists a solution v of the partial Max-SAT problem $P_k(\mathcal{T}, \Sigma)$ such that $\mathcal{E} = \{\, \alpha \in \mathcal{T} \mid v(p_\alpha) = 1 \,\}$.*

Algorithm 1 shows how best excerpts are computed by using partial Max-SAT encoding. In Line 7, the algorithm iterates over every concept name A in Σ and the set of all subsumption justifications $\mathcal{J}_\mathcal{T}(A, \Sigma)$ are computed. The formula $G_{\mathcal{J}_A}$ is computed next and stored in a set S. After the iteration over all the concept names A in Σ is complete, the partial Max-SAT problem $P_k(\mathcal{T}, \Sigma)$ is constructed with the help of the formulas $G_{\mathcal{J}_A}$ that are stored in S. Subsequently, a solution v of $P_k(\mathcal{T}, \Sigma)$ is computed using a partial Max-SAT solver and the best k-excerpt is returned by analysing which variables p_α have been set to 1 in the valuation v.

Our algorithm of computing subsumption justifications given below runs in exponential time in the size of $\mathcal{T}$ and Σ. Hence, we have that Algorithm 1 overall requires exponential time in the size of $\mathcal{T}$ and Σ in the worst case.

Algorithm 1: Computing Best k-Excerpts

```
function ComputeBestExcerpt($\mathcal{T}, \Sigma, k$)
    if $k = 0$ then
        return $\emptyset$
    if $k = |\mathcal{T}|$ then
        return $\mathcal{T}$
    $S := \emptyset$
    for every $A \in \Sigma \cap \mathsf{N_C}$ do
        Compute $\langle A, \Sigma\rangle$-subsumption justifications of $\mathcal{T}$: $\mathcal{J}_{\mathcal{T}}(A, \Sigma)$
        Transfer $\langle A, \Sigma\rangle$-subsumption justifications of $\mathcal{T}$ to its propostional formula $G_{\mathcal{J}_A}$
        $S := S \cup \{G_{\mathcal{J}_A}\}$
    Compute $P_k(\mathcal{T}, \Sigma)$ using $S$
    Find the set of solutions $V$ of $P_k(\mathcal{T}, \Sigma)$ using partial Max-SAT solver
    return $\{ \alpha \in \mathcal{T} \mid v(p_\alpha) = 1, v \in V \}$
```

4 Algorithms of Computing Subsumption Justifications

In the following subsections, we present algorithms for computing subsumer and subsumee justifications. The algorithms use the following notion of a cover of a set of sets. For a finite set S and a set $\mathbb{T} \subseteq 2^S$, we say that a set $\mathbb{M} \subseteq 2^S$ is a *cover of* $\mathbb{T}$ iff $\mathbb{M} \subseteq \mathbb{T}$ and there exists $\mathcal{M}' \in \mathbb{M}$ such that $\mathcal{M}' \subseteq \mathcal{M}$ for every $\mathcal{M} \in \mathbb{T}$. In other words, a cover is a subset of $\mathbb{T}$ containing all sets from $\mathbb{T}$ that are minimal w.r.t. $\subsetneq$. Therefore, a cover of the set of all subsumption modules also contains all subsumption justifications. We will use covers to characterise the output of our algorithms to ensure that all justifications have been computed.

The algorithms expect the input terminologies to be normalised. Thus, we have to normalise our terminologies first if they are not yet normalised (cf. Section 2). The denormalisation function $\delta_{\mathcal{T}}$ that we obtain from the process of normalisation is then applied to the outputs of the algorithms to obtain the subsumer and subsumee justifications of the original terminology. More precisely, each subsumer or subsumee justification $\mathcal{M} = \{\beta_1, \dots, \beta_n\}$ of the normalised terminology is transformed into the set $\{ \{\gamma\} \mid \gamma \in \delta_{\mathcal{T}}(\beta_1) \} \otimes \ldots \otimes \{ \{\gamma\} \mid \gamma \in \delta_{\mathcal{T}}(\beta_n) \}$ to obtain subsumer or subsumee justifications of the original terminology, respectively. In what follows we assume that $\mathcal{T}$, $\mathcal{T}_1$, and $\mathcal{T}_2$ are acyclic normalised $\mathcal{ELH}$-terminologies.

4.1 Computing Subsumer Justifications

The algorithm for computing subsumer justifications relies on the notion of a subsumer simulation between terminologies from [7,18], which we introduce first.

Definition 4 (Subsumer Simulation). *A relation $S \subseteq sig^{\mathsf{N_C}}(\mathcal{T}_1) \times sig^{\mathsf{N_C}}(\mathcal{T}_2)$ is called a Σ-*subsumer simulation from $\mathcal{T}_1$ to $\mathcal{T}_2$ *if the following conditions hold:*

($S_1^{\rightarrow}$) *if $(X_1, X_2) \in S$, then for every $B \in \Sigma$ with $\mathcal{T}_1 \models X_1 \sqsubseteq B$ it holds that $\mathcal{T}_2 \models X_2 \sqsubseteq B$; and*

($S_2^{\rightarrow}$) *if $(X_1, X_2) \in S$, then for each $Y_1 \bowtie_1 \exists r.Z_1 \in \mathcal{T}_1$ with $\mathcal{T}_1 \models X_1 \sqsubseteq Y_1$, $\mathcal{T}_1 \models r \sqsubseteq s$, $s \in \Sigma$, $\bowtie_1 \in \{\sqsubseteq, \equiv\}$, there exists $Y_2 \bowtie_2 \exists r'.Z_2 \in \mathcal{T}_2$ with $\mathcal{T}_2 \models X_2 \sqsubseteq Y_2$, $\bowtie_2 \in \{\sqsubseteq, \equiv\}$, $\mathcal{T}_2 \models r' \sqsubseteq s$, and $(Z_1, Z_2) \in S$.*

We write $\mathrm{sim}_{\rightarrow}^{\Sigma}([\mathcal{T}_1, X_1], [\mathcal{T}_2, X_2])$ *iff there is a Σ-subsumer simulation S from $\mathcal{T}_1$ to $\mathcal{T}_2$ with $(X_1, X_2) \in S$; and in the case of $\mathcal{T}_2 \subseteq \mathcal{T}_1$ we write* $\mathrm{sim}_{\rightarrow}^{\mathcal{T}_1,\Sigma}(X_1, X_2)$.

A subsumer simulation conveniently captures the set of subsumers in the following sense: If a Σ-subsumer simulation from $\mathcal{T}_1$ to $\mathcal{T}_2$ contains the pair (X_1, X_2), then X_2 entails w.r.t. $\mathcal{T}_2$ all subsumers of X_1 w.r.t. $\mathcal{T}_1$ that are formulated in the signature Σ. Formally, we obtain the following theorem from [18].

Theorem 3. *It holds that* $\mathrm{sim}_{\rightarrow}^{\Sigma}([\mathcal{T}_1, X_1], [\mathcal{T}_2, X_2])$ *iff for all $D \in \mathcal{ELH}_{\Sigma}$: $\mathcal{T}_1 \models X_1 \sqsubseteq D$ implies $\mathcal{T}_2 \models X_2 \sqsubseteq D$.*

Guided by the subsumer simulation notion, we can device our algorithm for computing subsumer justifications. Algorithm 2 computes the subsumer justifications for an acyclic normalised $\mathcal{ELH}$-terminology $\mathcal{T}$, a signature Σ, and a concept name X. Lines 3–10 of the algorithm compute all $\langle X, \Sigma\rangle$-subsumption modules of $\mathcal{T}$. To ensure that the returned modules are minimal w.r.t. $\subsetneq$, the algorithm calls the function $\mathrm{Minimise}_{\subseteq}(\mathbb{M}_X)$ in Line 11, which removes any set in $\mathbb{M}_X$ that is not minimal.

We illustrate Algorithm 2 with the following two examples. First example, let $\mathcal{T} = \{X \sqsubseteq B, X \sqsubseteq Y, Y \sqsubseteq B\}$ and $\Sigma = \{B\}$. Consider the execution of $\textsc{Cover}_{\rightarrow}(\mathcal{T}, X, \Sigma)$. In Line 4, $\mathbb{M}_X^{\rightarrow}$ is set to $\mathrm{Just}_{\mathcal{T}}(X \sqsubseteq B)$, where $\mathrm{Just}_{\mathcal{T}}(X \sqsubseteq B) = \{\{X \sqsubseteq B\}, \{X \sqsubseteq Y, Y \sqsubseteq B\}\}$. Since there are no axioms of the form $Y \sqsubseteq \exists r.Z \in \mathcal{T}$ or $Y \equiv \exists r.Z \in \mathcal{T}$, the lines 5–10 have no effect. Finally, the algorithm returns $\mathbb{M}_X^{\rightarrow}$ in Line 11.

For the second example, let $\mathcal{T} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$ and $\Sigma = \{A, B, s\}$, where $\alpha_1 = X \sqsubseteq \exists r.A$, $\alpha_2 = X \sqsubseteq \exists r.B$, $\alpha_3 = X \sqsubseteq \exists r.Y$, $\alpha_4 = Y \equiv A \sqcap B$, and $\alpha_5 = r \sqsubseteq s$. We consider again the execution of $\textsc{Cover}_{\rightarrow}(\mathcal{T}, X, \Sigma)$. We proceed to Line 5 as there are no concept names in Σ entailed by X w.r.t. $\mathcal{T}$. However, the concepts $\exists r.A$, $\exists r.B$ and $\exists r.Y$ are entailed by X w.r.t. $\mathcal{T}$. It holds that $\mathrm{sim}_{\rightarrow}^{\mathcal{T},\Sigma}(Z, Z')$ for every $(Z, Z') \in \{(A, A), (B, B), (Y, Y), (A, Y), (B, Y)\}$, whereas $\mathrm{sim}_{\rightarrow}^{\mathcal{T},\Sigma}(Z, Z')$ does not hold for any $(Z, Z') \in \{(A, B), (B, A), (Y, A), (Y, B)\}$. Therefore, for every $Z \in \{A, B, Y\}$ the recursive call $\textsc{Cover}_{\rightarrow}(\mathcal{T}, \Sigma, Z)$ is made in Line 8. The following sets are computed in lines 6–10: $\mathbb{M}_A^{\rightarrow} = \{\emptyset\}$, $\mathbb{M}_B^{\rightarrow} = \{\emptyset\}$, and $\mathbb{M}_Y^{\rightarrow} = \{\{\alpha_4\}\}$ as well as

$$\begin{aligned}
\mathbb{M}_{\exists s.A}^{\rightarrow} &= (\{\alpha_1, \alpha_5\} \otimes \mathbb{M}_A^{\rightarrow}) \cup (\{\alpha_3, \alpha_5\} \otimes \mathbb{M}_Y^{\rightarrow}) = \{\{\alpha_1, \alpha_5\}, \{\alpha_3, \alpha_4, \alpha_5\}\}\\
\mathbb{M}_{\exists s.B}^{\rightarrow} &= (\{\alpha_2, \alpha_5\} \otimes \mathbb{M}_B^{\rightarrow}) \cup (\{\alpha_3, \alpha_5\} \otimes \mathbb{M}_Y^{\rightarrow}) = \{\{\alpha_2, \alpha_5\}, \{\alpha_3, \alpha_4, \alpha_5\}\}\\
\mathbb{M}_{\exists s.Y}^{\rightarrow} &= \{\alpha_3, \alpha_5\} \otimes \mathbb{M}_Y^{\rightarrow} = \{\{\alpha_3, \alpha_4, \alpha_5\}\}\\
\mathbb{M}_X^{\rightarrow} &= \mathbb{M}_{\exists r.A}^{\rightarrow} \otimes \mathbb{M}_{\exists s.B}^{\rightarrow} \otimes \mathbb{M}_{\exists r.Y}^{\rightarrow} = \{\{\alpha_3, \alpha_4, \alpha_5\}\}.
\end{aligned}$$

Algorithm 2: Computing a Cover of all Subsumer Justifications

```
1  function $\textsc{Cover}_{\rightarrow}$ $(\mathcal{T}, X, \Sigma)$
2      $\mathbb{M}^{\rightarrow}_{X} = \{\emptyset\}$
3      for every $B \in \Sigma \cap \mathsf{N_C}$ such that $\mathcal{T} \models X \sqsubseteq B$ do
4          $\mathbb{M}^{\rightarrow}_{X} \coloneqq \mathbb{M}^{\rightarrow}_{X} \otimes \mathrm{Just}_{\mathcal{T}}(X \sqsubseteq B)$
5      for every $Y \bowtie_1 \exists r.Z \in \mathcal{T}$ $(\bowtie_1 \in \{\sqsubseteq, \equiv\})$ and $s \in \Sigma \cap \mathsf{N_R}$ such that $\mathcal{T} \models X \sqsubseteq Y$ and $\mathcal{T} \models r \sqsubseteq s$ do
6          $\mathbb{M}^{\rightarrow}_{\exists s.Z} \coloneqq \emptyset$
7          for every $Y' \bowtie_2 \exists r'.Z' \in \mathcal{T}$ $(\bowtie_2 \in \{\sqsubseteq, \equiv\})$ such that $\mathcal{T} \models X \sqsubseteq Y'$, $\mathcal{T} \models r' \sqsubseteq s$, and $\mathrm{sim}^{\mathcal{T},\Sigma}_{\rightarrow}(Z, Z')$ do
8              $\mathbb{M}^{\rightarrow}_{Z'} \coloneqq \textsc{Cover}_{\rightarrow}(\mathcal{T}, Z', \Sigma)$
9              $\mathbb{M}^{\rightarrow}_{\exists s.Z} \coloneqq \mathbb{M}^{\rightarrow}_{\exists r.Z} \cup \big(\{\{Y' \bowtie_2 \exists r'.Z'\}\} \otimes \mathbb{M}^{\rightarrow}_{Z'} \otimes \mathrm{Just}_{\mathcal{T}}(X \sqsubseteq Y') \otimes \mathrm{Just}_{\mathcal{T}}(r' \sqsubseteq s)\big)$
10         $\mathbb{M}^{\rightarrow}_{X} \coloneqq \mathbb{M}^{\rightarrow}_{X} \otimes \mathbb{M}^{\rightarrow}_{\exists s.Z}$
11     return $\mathrm{Minimise}_{\subseteq}(\mathbb{M}^{\rightarrow}_{X})$
```

Algorithm 3: Computing a Cover of all Subsumee Justifications – Conjunctive Case

```
1  function $\textsc{Cover}^{\sqcap}_{\leftarrow}$ $(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
2      let $\alpha_{X_1} \coloneqq X_1 \equiv Y_1 \sqcap \ldots \sqcap Y_m \in \mathcal{T}_1$
3      $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \emptyset$
4      for $\Gamma \in \mathit{DefForest}^{\sqcap}_{\mathcal{T}_2}(X_2)$ do
5          Let $\delta_{\Gamma} \coloneqq \{\, \mathrm{def}^{\sqcap}_{\mathcal{T}_2}(X') \mid X' \in \mathrm{leaves}(\Gamma) \cap \mathrm{def}^{\sqcap}_{\mathcal{T}_2} \,\}$
6          $\mathbb{M}^{\leftarrow}_{\Gamma} \coloneqq \{\Gamma\}$
7          for $X'_2 \in \mathit{leaves}(\Gamma)$ do
8              $\mathbb{M}^{\leftarrow}_{X'_2} \coloneqq \emptyset$
9              for every $X'_1 \in \mathit{non\text{-}conj}_{\mathcal{T}_1}(X_1)$ do
10                 if $\mathrm{sim}^{\Sigma}_{\leftarrow}([\mathcal{T}_1, X'_1], [\mathcal{T}_2 \setminus \delta_{\Gamma}, X'_2])$ then
11                     $\mathbb{M}^{\leftarrow}_{X'_2} \coloneqq \mathbb{M}^{\leftarrow}_{X'_2} \cup \textsc{Cover}_{\leftarrow}(\mathcal{T}_1, X'_1, \Sigma, \mathcal{T}_2 \setminus \delta_{\Gamma}, X'_2)$
12             $\mathbb{M}^{\leftarrow}_{\Gamma} \coloneqq \mathbb{M}^{\leftarrow}_{\Gamma} \otimes \mathbb{M}^{\leftarrow}_{X'_2}$
13         $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \mathbb{M}^{\leftarrow}_{(X_1,X_2)} \cup \mathbb{M}^{\leftarrow}_{\Gamma}$
14     return $\mathbb{M}^{\leftarrow}_{(X_1,X_2)}$
```

Algorithm 4: Computing a Cover of all Subsumee Justifications

```
1  function $\textsc{Cover}_{\leftarrow}$ $(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
2      if $X_1$ is not $\Sigma$-entailed w.r.t. $\mathcal{T}_1$ then
3          return $\{\emptyset\}$
4      $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \textsc{Cover}^{\mathsf{N_C}}_{\leftarrow}(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
5      if $X_1$ is not complex $\Sigma$-entailed in $\mathcal{T}_1$ then
6          return $\mathbb{M}^{\leftarrow}_{(X_1,X_2)}$
7      if $X_1 \equiv \exists r.Y \in \mathcal{T}_1$, and $r, Y$ are $\Sigma$-entailed w.r.t. $\mathcal{T}_1$ then
8          $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \mathbb{M}^{\leftarrow}_{(X_1,X_2)} \otimes \textsc{Cover}^{\exists}_{\leftarrow}(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
9      else if $X_1 \equiv Y_1 \sqcap \ldots \sqcap Y_m \in \mathcal{T}_1$ then
10         $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \mathbb{M}^{\leftarrow}_{(X_1,X_2)} \otimes \textsc{Cover}^{\sqcap}_{\leftarrow}(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
11     return $\mathrm{Minimise}_{\subseteq}(\mathbb{M}^{\leftarrow}_{(X_1,X_2)})$
```

Algorithm 5: Computing a Cover of all Subsumee Justifications – Local Case

```
1  function $\textsc{Cover}^{\mathsf{N_C}}_{\leftarrow}$ $(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
2      $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} = \{\emptyset\}$
3      for every $B \in \Sigma \cap \mathsf{N_C}$ such that $\mathcal{T}_1 \models B \sqsubseteq X_1$ do
4          $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \mathbb{M}^{\leftarrow}_{(X_1,X_2)} \otimes \mathrm{Just}_{\mathcal{T}_2}(B \sqsubseteq X_2)$
5      return $\mathbb{M}^{\leftarrow}_{(X_1,X_2)}$
```

Algorithm 6: Computing a Cover of all Subsumee Justifications – Existential Case

```
1  function $\textsc{Cover}^{\exists}_{\leftarrow}$ $(\mathcal{T}_1, X_1, \Sigma, \mathcal{T}_2, X_2)$
2      Let $\alpha_{X_1} \coloneqq X_1 \equiv \exists r.Y_1 \in \mathcal{T}_1$
3      $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \{\text{max-tree}^{\sqcap}_{\mathcal{T}_2}(X_2)\}$
4      for every $s \in \Sigma \cap \mathsf{N_R}$ such that $\mathcal{T}_1 \models s \sqsubseteq r$ do
5          $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \mathbb{M}^{\leftarrow}_{(X_1,X_2)} \otimes \mathrm{Just}_{\mathcal{T}_2}(s \sqsubseteq r)$
6      for every $X'_2 \in \text{non-conj}_{\mathcal{T}_2}(X_2)$ do
7          Let $\alpha_{X'_2} \coloneqq X'_2 \equiv \exists s.Y'_2 \in \mathcal{T}_2$
8          $\mathbb{M}^{\leftarrow}_{Y'_2} \coloneqq \textsc{Cover}_{\leftarrow}(\mathcal{T}_1, Y_1, \Sigma, \mathcal{T}_2, Y'_2)$
9          $\mathbb{M}^{\leftarrow}_{(X_1,X_2)} \coloneqq \mathbb{M}^{\leftarrow}_{(X_1,X_2)} \otimes \{\{\alpha_{X'_2}\}\} \otimes \mathbb{M}^{\leftarrow}_{Y'_2}$
10     return $\mathbb{M}^{\leftarrow}_{(X_1,X_2)}$
```

Fig. 3: Algorithms of computing subsumer and subsumee justifications

Finally, $\textsc{Cover}_{\rightarrow}(\mathcal{T}, X, \Sigma)$ returns $\text{Minimise}_{\subseteq}(\mathbb{M}^{\rightarrow}_{X}) = \{\{\alpha_3,\ \alpha_4,\ \alpha_5\}\}$ in Line 11.

The following theorem shows that Algorithm 2 indeed computes the set of subsumer modules, thus producing a cover of subsumer justifications.

Theorem 4. *Let $\mathbb{M}^{\rightarrow}_{X} \coloneqq \textsc{Cover}_{\rightarrow}(\mathcal{T}, X, \Sigma)$. Then $\mathbb{M}^{\rightarrow}_{X}$ is a cover of the set of $\langle X, \Sigma\rangle$-subsumer justifications of $\mathcal{T}$.*

Observe that $\textsc{Cover}_{\rightarrow}(\mathcal{T}, X, \Sigma)$ may be called several times during the execution of Algorithm 2. The algorithm can be optimised by caching the return value of the first execution, and retrieving it from memory for subsequent calls.

4.2 Computing Subsumee Justifications

The algorithm for computing subsumee justifications relies on the notion of subsumee simulation between terminologies [7, 18]. First we present some auxiliary notions for handling conjunctions on the left-hand side of subsumptions.

We define for each concept name X a so-called *definitorial forest* consisting of sets of axioms of the form $Y \equiv Y_1 \sqcap \ldots \sqcap Y_n$ which can be thought of as forming *trees*. Any $\langle X, \Sigma\rangle$-subsumee justification contains the axioms of a selection of these trees, i.e., one tree for every conjunction formulated over Σ that entails X w.r.t. $\mathcal{T}$. Formally, we define a set of a $\text{DefForest}^{\sqcap}_{\mathcal{T}}(X) \subseteq 2^{\mathcal{T}}$ to be the smallest set closed under the following conditions: $\emptyset \in \text{DefForest}^{\sqcap}_{\mathcal{T}}(X)$; $\{\alpha\} \in \text{DefForest}^{\sqcap}_{\mathcal{T}}(X)$ for $\alpha = X \equiv X_1 \sqcap \ldots \sqcap X_n \in \mathcal{T}$; and $\Gamma \cup \{\alpha\} \in \text{DefForest}^{\sqcap}_{\mathcal{T}}(X)$ for $\Gamma \in \text{DefForest}^{\sqcap}_{\mathcal{T}}(X)$ with $Z \equiv Z_1 \sqcap \ldots \sqcap Z_k \in \Gamma$ and $\alpha = Z_i \equiv Z^1_i \sqcap \ldots \sqcap Z^n_i \in \mathcal{T}$. Given $\Gamma \in \text{DefForest}^{\sqcap}_{\mathcal{T}}(X)$, we set $\text{leaves}(\Gamma) \coloneqq \text{sig}(\Gamma) \setminus \{\, X \in \text{sig}(C) \mid X \equiv C \in \Gamma\,\}$ if $\Gamma \neq \emptyset$; and $\{X\}$ otherwise. We denote the maximal element of $\text{DefForest}^{\sqcap}_{\mathcal{T}}(X)$ w.r.t. $\subseteq$ with $\text{max-tree}^{\sqcap}_{\mathcal{T}}(X)$. Finally, we set $\text{non-conj}_{\mathcal{T}}(X) \coloneqq \text{leaves}(\text{max-tree}^{\sqcap}_{\mathcal{T}}(X))$.

For example, let $\mathcal{T} = \{\alpha_1, \alpha_2, \alpha_3\}$, where $\alpha_1 = X \equiv Y \sqcap Z$, $\alpha_2 = Y \equiv Y_1 \sqcap Y_2$, and $\alpha_3 = Z \equiv Z_1 \sqcap Z_2$. Then $\text{DefForest}^{\sqcap}_{\mathcal{T}}(X) = \{\emptyset, \{\alpha_1\}, \{\alpha_1, \alpha_2\}, \{\alpha_1, \alpha_3\}, \{\alpha_1, \alpha_2, \alpha_3\}\}$. We have that $\text{leaves}(\{\alpha_1, \alpha_3\}) = \{Y, Z_1, Z_2\}$, $\text{max-tree}^{\sqcap}_{\mathcal{T}}(X) = \{\alpha_1, \alpha_2, \alpha_3\}$, and $\text{non-conj}_{\mathcal{T}}(X) = \{Y_1, Y_2, Z_1, Z_2\}$.

We say that $X \in \mathsf{N_C}$ is *Σ-entailed w.r.t. $\mathcal{T}$* iff there exists $C \in \mathcal{EL}_{\Sigma}$ with $\mathcal{T} \models C \sqsubseteq X$. We say that $r \in \mathsf{N_R}$ is *Σ-entailed w.r.t. $\mathcal{T}$* iff there exists $s \in \Sigma \cap \mathsf{N_R}$ with $\mathcal{T} \models s \sqsubseteq r$. Moreover, we say that X is *complex Σ-entailed w.r.t. $\mathcal{T}$* iff for every $Y \in \text{non-conj}_{\mathcal{T}}(X)$ one of the following conditions holds:

(i) there exists $B \in \Sigma$ such that $\mathcal{T} \models B \sqsubseteq Y$ and $\mathcal{T} \not\models B \sqsubseteq X$;
(ii) there exists $Y \equiv \exists r.Z \in \mathcal{T}$ such that r and Z are both Σ-entailed in $\mathcal{T}$.

For example, let $\mathcal{T} = \{X \equiv X_1 \sqcap X_2,\ B_1 \sqsubseteq X_1,\ X_2 \equiv \exists r.Z,\ B_2 \sqsubseteq Z, s \sqsubseteq r\}$. We have that $\text{non-conj}_{\mathcal{T}}(X) = \{X_1, X_2\}$, then r is Σ-entailed w.r.t. $\mathcal{T}$; X is complex Σ-entailed w.r.t. $\mathcal{T}$ for $\Sigma = \{B_1, B_2, s\}$; but X is not complex Σ'-entailed w.r.t. $\mathcal{T}$, where Σ' ranges over $\{B_1, B_2\}$, $\{B_1, s\}$, $\{B_2, s\}$. Additionally, X is not complex Σ-entailed w.r.t. $\mathcal{T} \cup \{B_1 \sqsubseteq X\}$.

Definition 5 (Subsumee Simulation). *We say that a relation $S \subseteq sig^{\mathsf{N_C}}(\mathcal{T}_1) \times sig^{\mathsf{N_C}}(\mathcal{T}_2)$ is a* Σ-subsumee simulation from $\mathcal{T}_1$ to $\mathcal{T}_2$ *iff the following conditions are satisfied:*

($S_1^{\leftarrow}$) *if* $(X_1, X_2) \in S$*, then for every* $B \in \Sigma$ *with* $\mathcal{T}_1 \models B \sqsubseteq X_1$ *it holds that* $\mathcal{T}_2 \models B \sqsubseteq X_2$*;*

($S_2^{\leftarrow}$) *if* $(X_1, X_2) \in S$ *and* $X_1 \equiv \exists r.Y_1 \in \mathcal{T}_1$ *such that* $\mathcal{T}_1 \models s \sqsubseteq r, s \in \Sigma$ *and* Y_1 *is* Σ*-entailed in* $\mathcal{T}_1$*, then for every* $X_2' \in$ *non-conj*$_{\mathcal{T}_2}(X_2)$ *there exists* $X_2' \equiv \exists r'.Y_2 \in \mathcal{T}_2$*, such that* $(Y_1, Y_2) \in S$ *and* $\mathcal{T}_2 \models s \sqsubseteq r'$*;*

($S_3^{\leftarrow}$) *if* $(X_1, X_2) \in S$ *and* $X_1 \equiv Y_1 \sqcap \ldots \sqcap Y_n \in \mathcal{T}_1$*, then for every* $X_2' \in$ *non-conj*$_{\mathcal{T}_2}(X_2)$ *there exists* $X_1' \in$ *non-conj*$_{\mathcal{T}_1}(X_1)$ *with* $(X_1', X_2') \in S$*.*

We write $\mathrm{sim}_{\leftarrow}^{\Sigma}([\mathcal{T}_1, X_1], [\mathcal{T}_2, X_2])$ *iff there exists a* Σ*-subsumee simulation* S *from* $\mathcal{T}_1$ *to* $\mathcal{T}_2$ *with* $(X_1, X_2) \in S$*. Moreover, we write* $\mathrm{sim}_{\leftarrow}^{\mathcal{T}_1, \Sigma}(X_1, X_2)$ *iff there exists a* Σ*-subsumee simulation* S *from* $\mathcal{T}_1$ *to* $\mathcal{T}_1$ *with* $(X_1, X_2) \in S$*.*

Analogously to subsumer simulations, a subsumee simulation captures the set of subsumees as it is made precise in the following theorem from [18].

Theorem 5. *It holds that* $\mathrm{sim}_{\leftarrow}^{\Sigma}([\mathcal{T}_1, X_1], [\mathcal{T}_2, X_2])$ *iff for every* $D \in \mathcal{ELH}_{\Sigma}$*:* $\mathcal{T}_1 \models D \sqsubseteq X_1$ *implies* $\mathcal{T}_2 \models D \sqsubseteq X_2$*.*

Using the notion of a subsumee simulation, we can device Algorithm 4 for computing a cover of the subsumee justifications for a given $\mathcal{ELH}$-terminology $\mathcal{T}$, a concept name X, and a signature Σ. The correct function call for obtaining the $\langle X, \Sigma\rangle$-subsumee justifications of $\mathcal{T}$ is $\textsc{Cover}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$. Note that Algorithm 3, Algorithm 5, and Algorithm 6 are called as subroutines in Line 4, 8 and 10 in Algorithm 4. The four different parameters for Algorithm 4 are needed due to the recursive calls in Algorithm 3 (Line 11) and Algorithm 6 (Line 8).

We illustrate Algorithm 4 with the following example. Let $\mathcal{T} = \{X \equiv \exists r.Y,\ Y \equiv \exists s.Z,\ Z \equiv A \sqcap Z',\ A \sqsubseteq B,\ B \sqsubseteq Z',\ Z' \sqsubseteq A\}$ be an $\mathcal{EL}$-terminology, and let $\Sigma = \{A, B, r, s\}$ be a signature. It can easily be seen that $\mathcal{T}$ is normalised.

Consider the execution of $\textsc{Cover}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$. As X is (complex) Σ-entailed, $\textsc{Cover}_{\leftarrow}^{\mathsf{N_C}}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$ is called in Line 4. The for-loop in lines 3–4 of Algorithm 5 does not apply as $\mathcal{T} \not\models A \sqsubseteq X$ and $\mathcal{T} \not\models B \sqsubseteq X$. We obtain $\textsc{Cover}_{\leftarrow}^{\mathsf{N_C}}(\mathcal{T}, X, \Sigma, \mathcal{T}, X) = \{\emptyset\}$ backtracking to Line 4 of $\textsc{Cover}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$. The if-statement in Line 7 applies as $\mathcal{T}$ contains an axiom of the form $X \equiv \exists r.Y$, where X and r are each Σ-entailed. We proceed with $\textsc{Cover}_{\leftarrow}^{\exists}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$ in Line 8. We obtain $\mathbb{M}_{(X,X)}^{\leftarrow} := \{\text{max-tree}_{\mathcal{T}}^{\sqcap}(X)\} = \{\emptyset\}$ in Line 3 of Algorithm 6. Since $\text{non-conj}_{\mathcal{T}}(X) = \{X\}$ and $X \equiv \exists r.Y \in \mathcal{T}$, the recursive call $\textsc{Cover}_{\leftarrow}(\mathcal{T}, Y, \Sigma, \mathcal{T}, Y)$ in Line 8 of Algorithm 6 is made.

Then, in Line 8 of Algorithm 4, $\textsc{Cover}_{\leftarrow}^{\exists}(\mathcal{T}, Y, \Sigma, \mathcal{T}, Y)$ is called as Y is complex Σ-entailed w.r.t. $\mathcal{T}$, $Y \equiv \exists s.Z \in \mathcal{T}$, and s, Z are each Σ-entailed.

Similar to $\textsc{Cover}_{\leftarrow}^{\exists}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$, the execution of $\textsc{Cover}_{\leftarrow}^{\exists}(\mathcal{T}, Y, \Sigma, \mathcal{T}, Y)$ invokes $\textsc{Cover}_{\leftarrow}(\mathcal{T}, Z, \Sigma, \mathcal{T}, Z)$ from Line 8 of Algorithm 6.

As Z is Σ-entailed w.r.t. $\mathcal{T}$, we have that $\textsc{Cover}_{\leftarrow}^{\mathsf{N_C}}(\mathcal{T}, Z, \Sigma, \mathcal{T}, Z)$ is executed. The for-loop in Line 3 of Algorithm 5 applies as $\mathcal{T} \models A \sqsubseteq Z$ and $\mathcal{T} \models B \sqsubseteq Z$ so that we have $\mathbb{M}_Z^{\leftarrow} := \text{Just}_{\mathcal{T}}(A \sqsubseteq Z) \otimes \text{Just}_{\mathcal{T}}(B \sqsubseteq Z)$, where $\text{Just}_{\mathcal{T}}(A \sqsubseteq Z) = \text{Just}_{\mathcal{T}}(B \sqsubseteq Z) = \{Z \equiv A \sqcap Z',\ A \sqsubseteq B,\ B \sqsubseteq Z',\ Z' \sqsubseteq A\}$. This finishes the call $\textsc{Cover}_{\leftarrow}^{\mathsf{N_C}}(\mathcal{T}, Z, \Sigma, \mathcal{T}, Z)$, and we backtract to Line 4 of $\textsc{Cover}_{\leftarrow}(\mathcal{T}, Z, \Sigma, \mathcal{T}, Z)$. As Z is not complex Σ-entailed, this finishes the call $\textsc{Cover}_{\leftarrow}(\mathcal{T}, Z, \Sigma, \mathcal{T}, Z)$ with $\mathbb{M}_Z^{\leftarrow} = \{Z \equiv A \sqcap Z',\ A \sqsubseteq B,\ B \sqsubseteq Z',\ Z' \sqsubseteq A\}$.

Ontologies	Snomed CT		NCI	
$(\lvert\Sigma \cap \mathsf{N_C}\rvert, \lvert\Sigma \cap \mathsf{N_R}\rvert)$	(10,10)	(30,10)	(10,10)	(30,10)
Nb. of all Subsumption Justifications	1.0 / 19.0 / 1.0 / 0.9	1.0 / 1328.0 / 1.0 / 10.7	1.0 / 136.0 / 1.0 / 3.5	1.0 / 7008.0 / 1.0 / 41.8
Card. of a Subsumption Justification	0.0 / 18.0 / 0.0 / 1.7	0.0 / 15.0 / 0.0 / 3.2	0.0 / 15.0 / 0.0 / 3.2	0.0 / 27.0 / 0.0 / 8.1
Success Rate	88.7%	82.4%	84.8%	91.7%
Computation Time (s)	0.2 / 519.7 / 0.4 / 59.2	0.7 / 576.3 / 1.6 / 28.5	0.2 / 472.4 / 1.3 / 66.9	0.2 / 577.3 / 7.6 / 97.0

Table 1: The statistics of experiments on computing all subsumption justifications for signatures generated at random, 1000 signatures of each size (minimal / maximal / median / standard deviation)

Sub-Task	JUST	Reasoner	Simulation Check	others
Percentage (%)	94.60	1.79	1.57	2.04

Table 2: Percentage of computation time consumed by sub-task of the algorithm for computing subsumption justifications

We backtrack to Line 8 of $\text{Cover}^{\exists}_{\leftarrow}(\mathcal{T}, Y, \Sigma, \mathcal{T}, Y)$ and set $\mathbb{M}^{\leftarrow}_{Y} := \mathbb{M}^{\leftarrow}_{Y} \otimes \{\{Y \equiv \exists s.Z\}\} \otimes \mathbb{M}^{\leftarrow}_{Z}$ which yields $\mathbb{M}^{\leftarrow}_{Y} = \{\{Y \equiv \exists s.Z,\ Z \equiv A \sqcap Z',\ A \sqsubseteq B,\ B \sqsubseteq Z',\ Z' \sqsubseteq A\}\}$. This finishes the call $\text{Cover}^{\exists}_{\leftarrow}(\mathcal{T}, Y, \Sigma, \mathcal{T}, Y)$ and it backtracks to Line 8 and ends the call $\text{Cover}_{\leftarrow}(\mathcal{T}, Y, \Sigma, \mathcal{T}, Y)$. We set $\mathbb{M}^{\leftarrow}_{X} := \mathbb{M}^{\leftarrow}_{X} \otimes \{\{X \equiv \exists r.Y\}\} \otimes \mathbb{M}^{\leftarrow}_{Y}$ in Line 9 of Algorithm 6 for $\text{Cover}^{\exists}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$. Thus $\text{Cover}^{\exists}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$ returns $\mathbb{M}^{\leftarrow}_{X} = \{\{X \equiv \exists r.Y,\ Y \equiv \exists s.Z,\ Z \equiv A \sqcap Z',\ A \sqsubseteq B,\ B \sqsubseteq Z',\ Z' \sqsubseteq A\}\}$ and we backtrack to Line 10 of Algorithm 4. Finally, all sets that are not minimal w.r.t. $\subsetneq$ are removed from $\mathbb{M}^{\leftarrow}_{X}$ in Line 11, which ends the execution of $\text{Cover}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$.

The following theorem shows that Algorithm 4 indeed computes a cover of the set of subsumee modules. Thus every subsumee justification is guaranteed to be among the computed sets of axioms.

Theorem 6. *Let $\mathbb{M}^{\leftarrow}_{X} \coloneqq \text{Cover}_{\leftarrow}(\mathcal{T}, X, \Sigma, \mathcal{T}, X)$. Then $\mathbb{M}^{\leftarrow}_{X}$ is the set of all $\langle X, \Sigma\rangle$-subsumee justifications of $\mathcal{T}$.*

5 Evaluation

We have implemented our algorithms for computing subsumption justifications, minimal (basic) modules, and best excerpts in Java. The performance of the implementation has been evaluated using the $\mathcal{EL}$-fragment of two prominent biomedical ontologies: Snomed CT (version Jan 2016), a terminology consisting of 317 891 axioms, and NCI (version 16.03d),[3] a terminology containing 165 341 axioms. To compute the sets $\text{Just}_{\mathcal{T}}(\alpha)$, we deployed the SAT-based tool BEACON [1], which uses an efficient group-MUS enumerator. To solve our partial Max-SAT problem, we made use of the system Sat4j [16]. All experiments were conducted with a timeout of 10 minutes on machines equipped with an Intel Xeon Core 4 Duo CPU running at 2.50 GHz and with 64 GiB of RAM.

Computation of all Subsumption Justifications Table 1 shows the results obtained for computing all subsumption justifications. The first row indicates the ontology used in each experiment. The experiments are divided into four categories according to the numbers of concept and role names included in an input signature, as specified in the second row. For each category, we generated 1000

[3] `http://evs.nci.nih.gov/ftp1/NCI_Thesaurus`

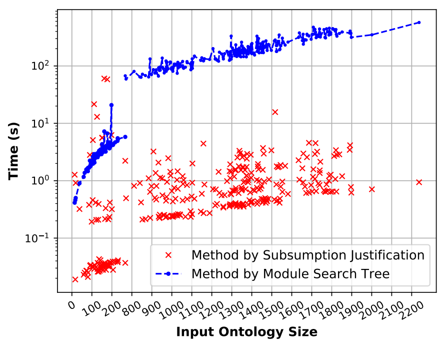

Fig. 4: Time comparison of computing minimal modules by our method (subsumption justification based approach, cf. Theorem 1) and the existing module search tree based approach [5] over different sized input ontologies

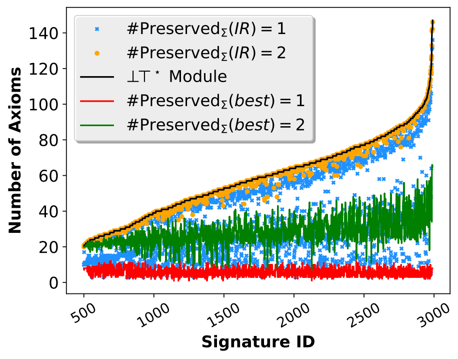

Fig. 5: Comparison of the best excerpts (our approach) and the approximating excerpts (IR approach [4]) over 2500 signatures, each of which consists of a concept name from Snomed CT and its TOP-concept named *SNOMED CT Concept*

random signatures and computed the corresponding subsumption justifications for each concept name in the signature. Row 3 shows that multiple subsumption justifications can exist in real-world ontologies, e.g., there are 1328 subsumption justifications for a random signature consisting of 30 concept and 10 role names in Snomed CT. Meanwhile, Row 4 reports the cardinality of subsumption justifications, e.g., the largest one having 27 axioms for a signature of 30 concept and 10 role names from NCI. Row 5 shows that the subsumption justifications for more than 82.4% of random signatures can be computed within 10 mins, whereas the statics of the actual computation times is given in Row 6. Moreover, Table 2 details how the computation time was spent on different sub-tasks which determined the bottleneck of our tool. Indeed, 94.6% of the computation time was spent by BEACON on computing all justifications for concept name inclusions. Therefore, a considerable boost in performance of our tool can be expected by precomputing such justifications.

Computation of all Minimal Basic Modules We compare our approach for computing all minimal basic modules with the search algorithm proposed in [5] in terms of computation time, as depicted in Figure 4. The x-axis stands for the sizes of input ontologies. To obtain different sized input ontologies, we used random signatures to extract their MEX-modules [14], yielding 328 sub-ontologies of sizes ranging from 14 to 2 271. Our method (red squares) was generally about 10 times faster than the search-based approach (blue triangles) except for 11 small sized input ontologies. This indicates that our approach is suitable for computing all minimal basic modules, esp. for large ontologies.

Computation of Best Excerpts We compare the size of locality based modules with the number of axioms in IR-excerpts [4] and best excerpts needed to preserve the same amount of knowledge. We denote with $\#\text{Preserved}_\Sigma(IR) = n$ and $\#\text{Preserved}_\Sigma(best) = n$, for $n \in \{1, 2\}$, the minimal number of axioms needed to preserve the knowledge of n concept names w.r.t. the signature Σ by an IR-excerpt and best excerpt, respectively. In this experiment, instead of using random signatures, we consider a scenario where a user searches for sub-ontologies of Snomed CT related to a particular concept name. We compute 2 500 different signatures each consisting of a concept name related to diseases, the TOP-concept and all role names of Snomed CT.

In Figure 5, these 2 500 signatures are ranked increasingly by the sizes of their $\bot\top^*$-local modules (the black line) along the x-axis. The y-axis represents the number of axioms in the module and excerpts for a signature. The red (resp. green) line presents the sizes of best excerpts that preserve the knowledge for one (resp. two) concept names, i.e., $\#Preserved_{\Sigma}(best) = 1$ (resp. $\#Preserved_{\Sigma}(best) = 2$); similarly, the blue (resp. orange) dots for IR-excerpts. We can see that the red line is below all blue dots and the green line is below all orange dots. Consequently, the best excerpts are always smaller than IR-based excerpts for preserving same degree of information. In other words, best excerpts provide a more concise way to zoom in on an ontology. Our experiment also shows that our Max-SAT encoding works efficiently. After computing the subsumption justifications for all concept names in a signature, it only takes 0.15 s on average to compute best excerpts.

6 Conclusion

We have presented algorithms of computing subsumption justifications, minimal modules and best excerpts for an acyclic $\mathcal{ELH}$-terminology and a signature. Minimal modules and best excerpts can be applied in the ontology selection process and they can be used for ontology summarization and visualization. We have conducted an evaluation with large biomedical ontologies that demonstrates the viability of our algorithms in practice. It turns out that in most cases the set of all minimal modules can be computed faster than with another algorithm based on search [5]. Best excerpts can be used to evaluate the quality of ontology excerpts based on Information Retrieval or of other (incomplete) module notions. We expect that the algorithms can be extended to deal with cyclic terminologies, domain and range restrictions in order to be applicable for, e.g., linked data summarization by providing small sized basic modules.

References

1. Arif, M.F., Mencía, C., Ignatiev, A., Manthey, N., Peñaloza, R., Marques-Silva, J.: BEACON: an efficient sat-based tool for debugging EL^+. In: Proc. of SAT'16, pp. 521–530 (2016)
2. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, 2nd edn. (2010)
3. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic EL^+. In: Proc. of KI'07, pp. 52–67 (2007)
4. Chen, J., Ludwig, M., Ma, Y., Walther, D.: Towards extracting ontology excerpts. In: Proc. of KSEM'15, pp. 78–89 (2015)
5. Chen, J., Ludwig, M., Walther, D.: On computing minimal EL-subsumption modules. In: Proc. of WOMoCoE'16 (2016)

6. Del Vescovo, C., Peñaloza, R.: Dealing with ontologies using cods. In: Proc. of DL'14, pp. 157–168 (2014)
7. Ecke, A., Ludwig, M., Walther, D.: The concept difference for EL-terminologies using hypergraphs. In: Proc. of DChanges'13 (2013)
8. Fu, Z.: Extending the Power of Boolean Satisfiability: Techniques and Applications. Ph.D. thesis, Princeton University (2007)
9. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. JAIR 31(1), 273–318 (2008)
10. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding all justifications of OWL DL entailments. In: Proc. of ISWC'07 & ASWC'07, pp. 267–280 (2007)
11. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A.: Debugging unsatisfiable classes in OWL ontologies. J. Web Sem. 3(4), 268–293 (2005)
12. Konev, B., Ludwig, M., Walther, D., Wolter, F.: The logical difference for the lightweight description logic EL. JAIR 44, 633–708 (2012)
13. Konev, B., Lutz, C., Walther, D., Wolter, F.: Semantic modularity and module extraction in description logics. In: Proc. of ECAI'08, pp. 55–59 (2008)
14. Konev, B., Lutz, C., Walther, D., Wolter, F.: Model-theoretic inseparability and modularity of description logic ontologies. Artificial Intelligence 203, 66–103 (2013)
15. Kontchakov, R., Pulina, L., Sattler, U., Schneider, T., Selmer, P., Wolter, F., Zakharyaschev, M.: Minimal module extraction from DL-lite ontologies using qbf solvers. In: Proc. of DL'09, pp. 836–841 (2009)
16. Le Berre, D., Parrain, A.: The Sat4j library, release 2.2. Journal on Satisfiability, Boolean Modeling and Computation 7(2-3), 59–64 (2010)
17. Ludwig, M.: Just: a tool for computing justifications w.r.t. ELH ontologies. In: Proc. of ORE'14, pp. 1–7 (2014)
18. Ludwig, M., Walther, D.: The logical difference for ELHr-terminologies using hypergraphs. In: Proc. of ECAI'14, pp. 555–560 (2014)
19. Lutz, C., Wolter, F.: Deciding inseparability and conservative extensions in the description logic EL. Journal of Symbolic Computation 45(2), 194–228 (2010)
20. Martín-Recuerda, F., Walther, D.: Fast modularisation and atomic decomposition of ontologies using axiom dependency hypergraphs. In: Proc. of ISWC'14, pp. 49–64 (2014)
21. Romero, A.A., Kaminski, M., Grau, B.C., Horrocks, I.: Module extraction in expressive ontology languages via datalog reasoning. JAIR 55, pp. 499–564 (2016)
22. Sattler, U., Schneider, T., Zakharyaschev, M.: Which kind of module should I extract? In: Proc. of DL'09 (2009)
23. Schlicht, A., Stuckenschmidt, H.: Criteria-based partitioning of large ontologies. In: Proc. of K-CAP'07, pp. 171–172 (2007)
24. Sinz, C.: Towards an optimal cnf encoding of boolean cardinality constraints. In: Proc. of CP'05), pp. 827–831 (2005)
25. Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D.: RDF digest: Efficient summarization of RDF/S kbs. In: Proc. of ESWC'15, pp. 119–134 (2015)
26. Vescovo, C.D., Gessler, D., Klinov, P., Parsia, B., Sattler, U., Schneider, T., Winget, A.: Decomposition and modular structure of bioportal ontologies. In: Proc. of ISWC'11, pp. 130–145 (2011)
27. Vescovo, C.D., Parsia, B., Sattler, U.: Logical relevance in ontologies, In: Proc. of DL'12 (2012)
28. Vescovo, C.D., Parsia, B., Sattler, U., Schneider, T.: The modular structure of an ontology: Atomic decomposition, In: Proc. of IJCAI'11, pp. 2232–2237. (2011)
29. Zhang, X., Cheng, G., Qu, Y.: Ontology summarization based on RDF sentence graph. In: Proc. of WWW'07, pp. 707–716 (2007)
30. Zhou, Z., Qi, G., Suntisrivaraporn, B.: A new method of finding all justifications in OWL 2 EL. In: Proc. of WI'13, pp. 213–220 (2013)