Knowledge extraction from large ontologies

Hui Yang

université PARIS-SACLAY

Knowledge Extraction from Large Ontologies

Extraction de connaissances à partir d'ontologies de grandes tailles

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique
Graduate School : Informatique et sciences du numérique, Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **Laboratoire interdisciplinaire des sciences du numérique (Université Paris-Saclay, CNRS)**, sous la direction de **Nicole BIDOIT TOLLU**, professeure, le co-encadrement de **Yue MA**, maître de conférences

Thèse soutenue à Paris-Saclay, le 16 mai 2023, par

Hui YANG

Composition du jury
Membres du jury avec voix délibérative

Marie-Christine ROUSSET Professeure, Université de Grenoble	Présidente
François GOASDOUÉ Professeur, Université de Rennes	Rapporteur & examinateur
Meghyn BIENVENU Directrice de recherche, Université de Bordeaux	Rapporteure & examinatrice
Marie-Laure MUGNIER Professeure, Université de Montpellier	Examinatrice
Michaël THOMAZO Chargé de recherche, Ecole normale supérieure de Paris	Examinateur

THESE DE DOCTORAT
NNT : 2023UPASG033

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Extraction de connaissances à partir d'ontologies de grandes tailles
Mots clés: Ontologies, Raisonnement, Logiques de description

Résumé: Parce que les ontologies du monde réel, largement utilisées en pratique, sont souvent complexes et très volumineuses, concevoir des outils permettant aux utilisateurs de se concentrer sur des sous-ontologies correspondant à leurs centres d'intérêts est devenu un défi majeur. Dans ce contexte, ce travail étudie trois approches différentes pour extraire des connaissances à partir de grandes ontologies : (1) *Les justifications* qui sont des sous-ontologies minimales de l'ontologie d'origine permettant de dériver une conclusion spécifique ; (2) *Les modules déductifs* qui sont des sous-ontologies de l'ontologie d'origine et qui préservent toutes les implications relatives à un vocabulaire donné, ce vocabulaire traduisant l'intérêt de l'utilisateur ; et (3) *Modules généraux* qui sont de nouvelles ontologies (pas nécessairement des sous-ontologies) dont l'ensemble des implications relatives à un vocabulaire donné est identique à celui de l'ontologie d'origine.

Pour le calcul de justifications et de modules déductifs, cette thèse propose de nouvelles méthodes basées sur la résolution. Ce sont des méthodes qui procèdent en deux étapes : (i) le codage de la dérivation des justifications, resp. des modules déductifs sous forme de clauses Horn ; (ii) le calcul des justifications, resp. des modules déductifs, par résolution sur les clauses de Horn obtenues. Pour encoder la dérivation des justifications, nous exploitons une représentation par graphe des ontologies et introduisons un système de règles d'inférence qui sont plus compactes que les règles des systèmes connus. Pour coder la dérivation des modules déductifs, nous introduisons une nouvelle notion, appelée forêt, qui est une représentation par graphe qui quant à elle capture toutes les implications logiques relatives à un vocabulaire donné.

Pour le calcul des modules généraux, nous étudions une nouvelle méthode basée sur la résolution, inspirée de l'approche existante pour le calcul des interpolants uniformes. Cette méthode est, en général, plus efficace et produit des ontologies de meilleure qualité.

Enfin, dans cette thèse, toutes nos approches sont évaluées en implémentant des prototypes qui servent à tester des ontologies du monde réel de grande taille. Les résultats expérimentaux sont comparés avec ceux des méthodes existantes les plus efficaces et permettent de valider l'efficacité et la quaité de nos méthodes.

Title: Knowledge Extraction from Large Ontologies
Keywords: Ontologies, Reasoning, Description Logics

Abstract: Because widely used real-world ontologies are often complex and large, one crucial challenge has emerged: designing tools for users to focus on sub-ontologies corresponding to their specific interests. To this end, this work investigates three different approaches for extracting knowledge from large ontologies: (1) *Justification*, a minimal sub-ontology of the original ontology that derives a specific conclusion; (2) *Deductive module*, a sub-ontology that preserves all entailments wrt a given vocabulary capturing the user interest; and (3) *General module*, a new ontology not necessarily a sub-ontology, that ensures to perform the same set of entailments as the original one over a given vocabulary.

For computing justifications and deductive modules, we propose SAT-based methods that are conducted in two steps: (i) encoding the derivation of justifications (resp. deductive modules) as Horn-clauses; (ii) computing justifications (resp. deductive modules) by resolution over these Horn-clauses. For encoding the derivation of justifications, we construct a graph representation of ontologies and propose a new set of inference rules, which are more compact than existing ones. For encoding the derivation of deductive modules, we introduced a new notion called the forest, which relies on a graph representation, capturing all the logical entailments over a given vocabulary.

For computing general modules, we proposed a new resolution-based method inspired by the existing approach for computing *uniform interpolants*. This method is, in general, more efficient and generates modules of better quality.

Finally, each proposed method has been evaluated by implementing a prototype used to test large real-world ontologies and the experimental results have been compared to those obtained with state-of-the-art methods, showing the advantages of our method in terms of efficiency and quality.

Synthèse en français

La conception et l'utilisation d'ontologies sont devenues courantes pour des applications du monde réel de divers types telles que la recherche dans le Web sémantique, la gestion des connaissances, la représentation de l'information et l'interopérabilité des systèmes. Cependant, les ontologies volumineuses et complexes peuvent poser des problèmes aux utilisateurs qui cherchent à se concentrer sur des sous-ontologies pertinentes pour leurs domaines d'intérêt.

Pour résoudre ce problème, des outils ont été développés pour extraire des connaissances à partir de grandes ontologies et permettre aux utilisateurs de se concentrer sur les sous-ontologies pertinentes. Cette thèse étudie trois approches différentes pour extraire de telles connaissances : les justifications, les modules déductifs et les modules généraux.

Les justifications sont des sous-ontologies minimales de l'ontologie originale qui participent à la déduction d'une conclusion spécifique. Les modules déductifs sont des sous-ontologies de l'ontologie originale qui conservent toutes les conclusions liées à un vocabulaire donné qui représente l'intérêt de l'utilisateur. Les modules généraux sont de nouvelles ontologies qui ne sont pas nécessairement des sous-ontologies mais qui ont le même ensemble de conclusions liées à un vocabulaire donné que l'ontologie originale.

Pour calculer les justifications, nous utilisons une méthode basée sur la résolution. Cette méthode consiste en deux étapes : (i) encoder la dérivation des justifications sous forme de clauses de Horn, et (ii) calculer les justifications par résolution sur les clauses de Horn obtenues. Nous utilisons une représentation graphique des ontologies et introduisons un système de règles d'inférence basé sur cette représentation pour coder la dérivation des justifications. Le principal avantage de notre approche est que notre système de règles d'inférence est plus compact que les systèmes existants, ce qui se traduit par un ensemble plus petit de clauses de Horn qui accélère le calcul de la résolution dans la plupart des cas.

Le calcul des modules déductifs suit la même démarche que les justifications. Cependant, contrairement aux justifications qui ne préservent que la conclusion donnée, les modules déductifs doivent conserver toutes les conclusions possibles sur le vocabulaire donné, déduit de l'ontologie originale. Pour surmonter ce défi, nous introduisons une nouvelle notion appelée forêt, qui est une autre représentation graphique qui capture, de manière finie, toutes les conclusions logiques liées à un vocabulaire donné. Sur la base de la notion de forêt, nous pouvons coder la dérivation des modules déductifs et les calculer pour le vocabulaire donné. De plus, nous proposons deux notions différentes de modules déductifs appelés modules pseudo-minimaux et modules complets pour assurer un compromis entre qualité du résultat et efficacité de calcul. Les modules pseudo-minimaux sont de bonne qualité mais nécessitent plus de temps, tandis que les modules complets peuvent être calculés efficacement mais ont une qualité relativement inférieure.

En autorisant de nouveaux axiomes qui ne font pas partie des axiomes originaux, les modules généraux peuvent être plus concis que les modules déductifs. Pour calculer les modules généraux, nous avons développé une nouvelle méthode basée sur des règles de résolution inspirées de l'approche existante pour le calcul des interpolants uniformes. Comme le montrent les résultats de l'évaluation, notre méthode est généralement plus efficace et produit des ontologies de meilleure qualité. De plus, sur la base de ce calcul, nous avons développé une nouvelle méthode de calcul des modules déductifs et des interpolations uniformes. Ces méthodes offrent une variété de choix aux utilisateurs pour extraire les connaissances qui les intéressent.

Nos résultats expérimentaux démontrent que toutes nos méthodes peuvent extraire efficacement des connaissances pertinentes à partir de grandes ontologies. Les approches proposées offrent des avantages tels que la compacité et l'efficacité de calcul tout en maintenant la qualité des ontologies extraites.

Contents

1 - Introduction

1.1 . Context

This thesis is conducted in the context of the R&D project AIDA (Artificial Intelligence for Digital Automation) led by IBM and financed by Bpifrance. The AIDA project aims to develop a platform that better integrates advanced artificial intelligence (AI) technologies into real-world applications for the companies participating in the project. Concretely, the goal is threefold: find new automation potentials to increase productivity; improve efficiency of automated systems; and make better recommendations and decisions.

The AI challenges targeted by the AIDA project are diverse:

- Limited data validity: in business decision-making, the environment (objectives, rules) is constantly changing, which requires AI solutions capable of adapting to rapid changes.
- Trust: to guarantee decision qualities, AI must be able to provide explainable, controllable, and auditable results.
- Context complexity: since processes and decisions are highly interdependent, AI needs to address complex contexts.

The AIDA project is divided into six work packages. This thesis belongs to the work package B that concerns extracting data from texts and domain ontologies in order to produce a structured representation of the information contained in various data sources. It is broken down into three sub-tasks:

- B.1 Extraction of structured data, from unstructured data, to be usable with pre-existing semantic sources (e.g. ontologies);
- B.2 Acquisition of knowledge (ontologies, rules, constraints, goals) from natural language sources and existing structured resources (ontologies, knowledge bases);
- B.3 Definition of a corpus for learning and validating algorithms.

Within the AIDA project, as a part of Task B.2, our interest is to acquire expressive Description Logics (DLs) based ontologies. Using formal knowledge has shown potential benefits in domain-specific research to help knowledge discovery. For example, the formal ontology FMA about human anatomy (with over 70,000 concepts) has been used to infer potential internal and hidden injuries from injuries visible in images [66]. In another case, the authors use reasoning over yeast metabolism to generate novel hypotheses [38], where the necessary background

knowledge and reasoning framework form a crucial part of a "robot scientist" that autonomously executes and evaluates scientific experiments. As these examples demonstrate, formalized domain knowledge can assist cutting-edge applications.

By definition, constructing ontologies consists of several steps: formulating concept names, property names, individuals, and axioms. In this thesis, we assume that concept and property names are provided as input by experts or extracted from texts for the concerned domain via automatic tools. Then the main task remains to generate a set of axioms that can capture the knowledge for the considered domain. Axioms can be simple concept hierarchy or can be more complex in the form of primitive and full definitions. The most complex axiom in the context of DLs is general concept inclusions (GCIs). The more complex the form of an axiom, the more difficult it is to construct both manually and automatically.

There have been many kinds of research on *ontogy learning* [56, 50] based on various methods, including *association rule mining* [11, 73, 21] and *formal concept analysis* [67, 8, 11, 7]. Additionally, there have been many studies on learning ontologies from textual resources, both in open domains [72, 17] and specific domains [64]. However, we note that it is also a common phenomenon that formal ontologies exist for particular domains. If this is the case, information in such ontologies is often interesting to be explored and can serve as a reasonable basis for constructing a domain ontology for a given application. Consider one of the application scenarios of AIDA, the financial domain. We can indeed make this assumption because we have publicly accessible ontologies, such as FIBO (`http://www.fibo.org`) and FinRegOnt (`https://finregont.com`). Therefore, this thesis focuses on providing efficient methods that allow extracting only relevant information from existing, in general, very large ontologies to improve efficiency in its downstream applications.

1.2 . Contributions of the Thesis

Effective knowledge representation and handling is one of the most important challenges of artificial intelligence. To this end, a wide variety of knowledge representation languages have been proposed, offering formal semantics and reasoning techniques for drawing implicit conclusions from elements that are explicitly represented. Description logic-based ontologies [9], one of the most successful knowledge representation languages, have been widely studied and is used in many areas, including medicine, biology, and finance. It provides structured representations of domain knowledge that are suitable for AI reasoning. Usually, ontologies contain a set of statements (axioms) about concept and role names (unary and binary predicates). Using a formalization based on Description Logics (DLs) allows DL reasoners to infer implicit information from an ontology. Modern ontologies are often large and complex, which can make ontology engineering challenging. For example, as of 3 January 2023, the medical ontology SNOMED CT [19], used in

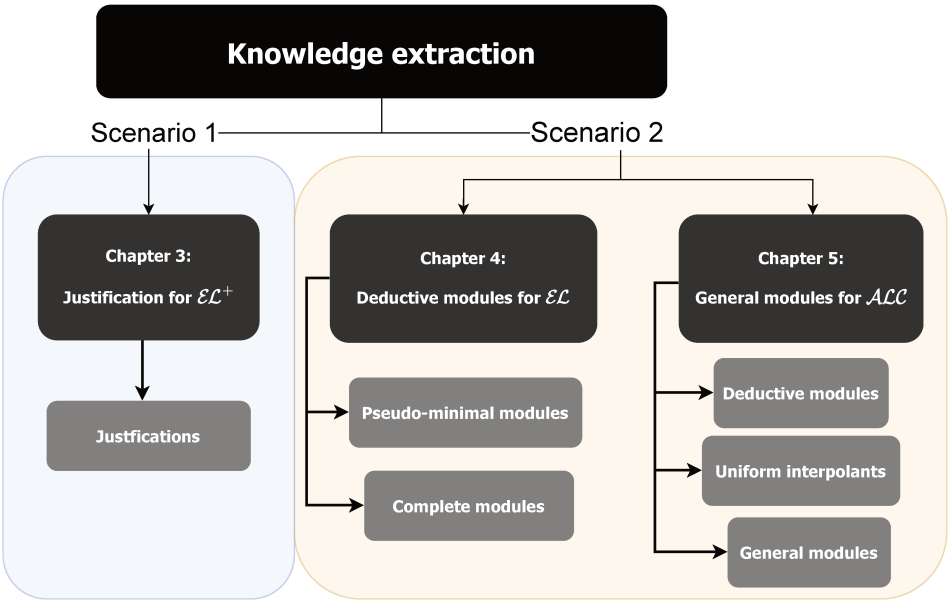


Figure 1.1: Main structure of the thesis.

the health-care systems of many countries, formalizes over 360,000 concepts, and the BioPortal repository of ontologies from the bio-medical domain [60] currently hosts 1,043 ontologies that use over 14 million concepts. Often, one is not interested in the entire content of an ontology, but only in a fragment. In this work, we are interested in the following two scenarios, which are also illustrated in Figure 1.1.

Scenario 1 One wants an explanation of a conclusion based on the given ontology. In this case, one may provide a particular conclusion and would like to compute a smaller sub-ontology $\mathcal{M}$ of the original ontology $\mathcal{O}$ that suffices to support this conclusion. In this case, one may use ***justifications*** (see Section 1.2.1). A justification $\mathcal{J}$ is a minimal sub-ontology of the original ontology $\mathcal{O}$ that derives the given conclusion, and thus, provides users with a concise explanation of the conclusion.

Scenario 2 One wants to analyze what the ontology states about a given set of concept and role names. In such a case, one may provide a *signature* Σ of concept and role names of interest and would like to compute a smaller ontology $\mathcal{M}$ that captures all the logical entailments of the original ontology $\mathcal{O}$ expressed using only the names in Σ. This problem has received a lot of attention in the past few years, and for an $\mathcal{M}$ satisfying those requirements, one may use ***deductive modules*** (see Section 1.2.2) or ***general modules*** (see Section 1.2.3). Deductive modules are classical ontology modules, which are subsets of the original ontology that preserve

logical information within the given signature Σ. In contrast, general modules generalize classical modules by allowing axioms that do not occur in the original ontology, which could lead to additional conciseness. Both deductive modules and general modules provide users with a concise view of the original ontology wrt to the given signature.

1.2.1 . Ontology justification

In Description Logics, a *justification* of a given conclusion is a minimal sub-ontology that derives the conclusion. Computing justifications has been widely explored for different tasks, for instance, for debugging ontologies [2, 31, 34] and computing ontology modules [16]. Extracting just one justification can be easy for tractable ontologies, such as $\mathcal{EL}^+$ [62]. For instance, we can find one justification by deleting unnecessary axioms one by one. However, there may exist more than one justification for a given conclusion. Computing all such justifications is computationally complex and reveals itself to be a challenging problem [63].

There are mainly two different approaches [62] to compute all justifications for a given conclusion, the *black-box* approach and the *glass-box* approach. The *black-box* approach [34] relies only on a reasoner and, as such, can be used for ontologies in any existing Description Logics. For example, a simple (naive) *black-box* approach would check all the subsets of the ontology using an existing reasoner and then filter the subset-minimal ones (i.e., justifications). Meanwhile, the glass-box approaches have achieved better performances over certain specific ontology languages (such as $\mathcal{EL}^+$-ontology) by going deep into the reasoning process. Among them, the class of SAT-based methods [37, 2, 3, 4, 57] performs the best. The main idea developed by SAT-based methods is to trace, in the first step, a *complete set of inferences* (*complete set* for short) that contribute to the derivation of a given conclusion, and then, in a second step, to use SAT tools or resolution to extract all justifications from these inferences.

In this thesis (Chapter 3), we propose a new glass-box approach for computing all justifications based on a new set of inference rules built from the graph representation of $\mathcal{EL}$-ontologies. In brief, we propose a *hypergraph representation of $\mathcal{EL}$-ontologies* and reformulated inferences that derive all possible given conclusions. By using our inferences, we can decrease the size of the *complete sets* of inferences which leads to smaller inputs for the SAT-based algorithm extracting justifications from the *complete set*.

1.2.2 . Ontology classical modules

Classical modules are subsets of an ontology that preserve entailments within a given signature Σ (a set of concept and role names). There is a variety of notions of module and properties they can satisfy that have been investigated in the literature [27, 43]. *Semantic modules* preserve all models of the ontology modulo the given signature Σ. This makes them undecidable already for light-weight DLs such as $\mathcal{EL}$ [44], which is why existing methods are often only able to

compute approximations of minimal semantic modules [23, 65]. A popular example are *locality-based modules*, which can be computed in a very short time [27]. However, locality-based modules can be large, even if the provided signature is small [15]. In contrast to semantic modules, *deductive modules* are decidable, and focus only on entailments in Σ that can be expressed in the DL under consideration. Deductive modules are a weaker form of semantic modules. However, computing subset-minimal deductive modules such as *minimal modules* [16] remains highly complex and time-consuming in practice.

In this work (Chapter 4), we propose to compute two types of deductive modules called *pseudo-minimal modules* and *complete modules* for $\mathcal{EL}$-ontologies that balance the computation cost and the result quality. Our method is inspired by the SAT-based approach [3, 57] developed to compute *justifications*. The main idea is to encode the derivations of a given entailment as a set of Horn-clauses, then to enumerate all the justifications of this entailment by SAT tools or *resolution* [37]. However, the computation of deductive modules is much more complex: First, the input is a vocabulary instead of an entailment and generating all the entailments over a given vocabulary can be complicated; Second, there may exist (even infinitely) many entailments over a given vocabulary. Therefore, instead of using justifications of the entailments directly, one has to find other proper ways to tackle the computation of deductive modules.

Our contribution is twofold: **(i)** We associate a forest with each given ontology and vocabulary to efficiently capture the entailments over the vocabulary. The definition of *forest* is inspired by the *regular tree grammar* developed in [59]. We can regard the forest and the regular tree grammar as a set of *derivation trees* and *derivation rules* that generate entailments over a given vocabulary, respectively. Moreover, we are able to deal with the case of infinitely many entailments by considering a finite subset of trees from our forest. **(ii)** We introduce two novel notions of deductive modules called *pseudo-minimal modules* and *complete modules*, and we develop an efficient SAT-based algorithm to compute them based on the notion of forest. Our pseudo-minimal modules are quite interesting approximations of minimal modules: (1) They are indeed minimal modules when there are finitely many entailments over a given vocabulary; (2) Moreover, our algorithm is 99.79 times faster on average than the state-of-the-art algorithm `Zoom` [16] which computes all the minimal modules but only for $\mathcal{EL}$-terminologies. Compared to pseudo-minimal modules, our complete modules are less concise but easier to compute. They are far more concise than the $\top\bot^*$-module, as demonstrated by our experiments, but their calculation time remains comparable.

1.2.3 . Ontology general modules

General modules were first investigated for the lightweight DLs $\mathcal{EL}$ in [58]. In brief, a *general module* for an ontology is an ideally substantially smaller ontology that preserves all entailments for a given signature. General modules generalize classical modules by allowing axioms not explicitly present in the input ontology,

which could bring additional conciseness. For instance, consider the ontology

$$\mathcal{O} = \{A_1 \sqsubseteq \exists s.A_1\} \cup \{A_i \sqsubseteq A_{i+1} \mid 0 \leq i \leq 8\}$$

and signature $\Sigma = \{s, A_0, A_9\}$, then the following ontology $\mathcal{M}$ is a general module for $\mathcal{O}$ and Σ:

$$\begin{aligned}\mathcal{M} = \{\ & A_0 \sqsubseteq A_1, \\ & A_1 \sqsubseteq \exists s.A_1 \\ & A_1 \sqsubseteq A_9\ \}.\end{aligned}$$

Here, $\mathcal{M}$ contains a new axiom $A_1 \sqsubseteq A_9$ that does not belong to $\mathcal{O}$. In contrast, for this example, the only deductive module for $\mathcal{O}$ and Σ is $\mathcal{O}$ itself, which contains more concept names and more axioms than the general module $\mathcal{M}$.

Classical modules are specific general modules that utilize only the axioms occuring in the original ontology. Now, if we require the general module to only utilize names from the specified signature, the modules obtained are known as *uniform interpolants*. The axioms of uniform interpolants may not occur in the input ontology. Uniform interpolants are useful for many tasks, such as for logical difference [52], abduction [18], information hiding [26], and proof generation [1]. While using only axioms in the specified signature could in principle allow uniform interpolants to be smaller than modules, the strict requirement on the signature means that uniform interpolants may not always exist (e.g., for $\mathcal{O}$ and Σ above), and, in case of $\mathcal{ALC}$, can be of a size that is triple exponential in the size of the input [55]. Despite this high complexity, practical implementations for computing uniform interpolants exist [78, 47]. However, their computation times are much higher than for module extraction and can produce very complex axioms.

In this work (Chapter 5), we present a method for computing *general modules* for $\mathcal{ALC}$-ontologies. We also provide a formal analysis of some properties of general modules we compute. Moreover, our method is able to compute both uniform interpolants and deductive modules for $\mathcal{ALC}$-ontologies in a significantly shorter time than the state-of-the-art. Our evaluation shows that all our methods, including the one for uniform interpolation, can compete with the run times of locality-based module extraction, while at the same time resulting in subtantially smaller ontologies.

1.3 . Structure of the Thesis

The structure of the thesis is as follows:

Chapter 2 In this chapter, we introduce some basic notions, definitions, and methods of Description Logics. We start by recalling some basic definitions regarding the Description Logics $\mathcal{EL}$, $\mathcal{EL}^+$, $\mathcal{ALC}$. Then we introduce ontology classification, justification and general modules.

Chapter 3 In this chapter, based on a graph representation of $\mathcal{EL}^+$-ontologies, we propose a new set of *inference rules* (called H-rules) and take advantage of them for providing a new method for computing all justifications for a given conclusion. We implemented a prototype in Python (version 3.7) called `minH` and validate our approach by running real-world ontology experiments. These experiments show that our graph-based approach outperforms PULi [37], the state-of-the-art algorithm, in most cases.

Chapter 4 In this chapter, we propose to compute two different kinds of deductive modules called *pseudo-minimal modules* and *complete modules* for $\mathcal{EL}$-ontology, and to this end, we develop an efficient SAT-based algorithm. We implemented a prototype in Python (version 3.7) called `ForMod`. Our experiments show that our pseudo-minimal modules are indeed minimal modules in almost all cases (98.9%), and computing pseudo-minimal modules is more efficient (99.79 times faster on average) than the state-of-the-art method `Zoom` for computing minimal modules. Also, our complete modules are more compact than $\top\bot^*$-modules, but their computation time remains comparable. Finally, note that our proposal applies to $\mathcal{EL}$-ontologies while `Zoom` only works for $\mathcal{EL}$-terminologies.

Chapter 5 In this chapter, we present a method for extracting general modules for ontologies formulated in the Description Logic $\mathcal{ALC}$. Our method is based on uniform interpolation and supported by some new theoretical results. Moreover, our method can be used for, and in fact, improves the computation of uniform interpolants and classical modules. We implemented a prototype in Python (version 3.7) called GEMO. Our evaluation indicates that our general modules are often smaller than classical modules and uniform interpolants computed by the state-of-the-art, and compared with uniform interpolants, can be computed in a significantly shorter time.

Chapter 6 In this chapter, we conclude the thesis by summarising the results and discussing possible future works.

2 - Preliminaries

2.1 . Description Logics

In this section, we introduce some basic notions of Description Logic that are used throughout this thesis. More details can be found in [9]. Description Logics is a formal knowledge representation language that facilitates the exploration of properties and relationships within a subject area through the use of fundamental concepts such as *concepts*, *roles*, and *individuals*. In this thesis, we are working with three Description Logics: $\mathcal{EL}$, $\mathcal{EL}^+$, and $\mathcal{ALC}$.

Let $\mathsf{N}_C = \{A, B, \ldots\}$ be an infinite set of concept names, and let $\mathsf{N}_R = \{r, s, t, \ldots\}$ be an infinite set of role names. We assume that N_C and N_R are disjoint. The syntax and semantics of $\mathcal{L}$ -concepts are given below. Note that $\mathcal{EL}$-concepts and $\mathcal{EL}^+$-concepts are of the same form. The difference between $\mathcal{EL}$ and $\mathcal{EL}^+$ ontologies is on the axiom part, which will be introduced later.

Definition 1 (Syntax of concepts) *The set of $\mathcal{EL}$ and $\mathcal{EL}^+$ concepts C are built according to the following grammar rule:*

$$C ::= \top \mid A \mid C \sqcap C \mid \exists r.C$$

where $A \in \mathsf{N}_C, r \in \mathsf{N}_R$. The set of $\mathcal{ALC}$-concepts C are built by:

$$C ::= \top \mid A \mid C \sqcap C \mid \exists r.C \mid \bot \mid \neg C \mid C \sqcup C \mid \forall r.C,$$

where $A \in \mathsf{N}_C, r \in \mathsf{N}_R$.

We distinguish the concept names in N_C from other concepts generated by the above grammar rule by calling $\exists r.C,\ \forall r.E,\ C \sqcap D, \ldots$ that contain symbols $\exists, \forall, \sqcup, \sqcap$, or $\neg$: *complex concept*.

Example 2 *For example, let us consider the following concept names related to the school:*

concept names : `Teacher`
`Student`
`Person`
`Course`

role names : `attend`
`attend`

Then we can build complex concepts as follows:

1. $\texttt{Student} \sqcap \texttt{Teacher}$: *it describes individuals that are both student and teacher;*
2. $\texttt{Student} \sqcup \texttt{Teacher}$: *it describes individuals that are either student or teacher;*
3. $\exists \texttt{attend.Course}$: *it describes individuals that attend some course;*
4. $\forall \texttt{attend.Course}$: *it describes individuals that do not attend anything other than a course;*
5. $\neg \texttt{Person}$: *it describes individuals that are not Person;*

Definition 3 (Semantics of concepts and role names) *For $\mathcal{L} \in \{\mathcal{EL}, \mathcal{EL}^+, \mathcal{ALC}\}$, an **interpretation** $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{L}$-concepts and role names consists of*

1. *a non-empty set $\Delta^{\mathcal{I}}$, and*
2. *a mapping $\cdot^{\mathcal{I}}$ that assigns each $\mathcal{L}$-concept C to a subset $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and each role name $r \in N_R$ to a subset $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ such that:*

$$
\begin{aligned}
&(\top)^{\mathcal{I}} = \Delta^{\mathcal{I}},\\
&(\bot)^{\mathcal{I}} = \emptyset,\\
&(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}\\
&(C \sqcap E)^{\mathcal{I}} = C^{\mathcal{I}} \cap E^{\mathcal{I}},\\
&(C \sqcup E)^{\mathcal{I}} = C^{\mathcal{I}} \cup E^{\mathcal{I}},\\
&(\exists r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \text{ there is a } b \in C^{\mathcal{I}} \text{ such that } (a,b) \in r^{\mathcal{I}}\}\\
&(\forall r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \text{ for all } b \in \Delta^{\mathcal{I}}, \text{ if } (a,b) \in r^{\mathcal{I}}, \text{ then } b \in C^{\mathcal{I}}\}
\end{aligned}
$$

We call $C^{\mathcal{I}}$ the extension of C in $\mathcal{I}$.

Example 4 (Example 2 cont'd) *For example, we may have an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ as follows. The universal set $\Delta^{\mathcal{I}}$ is*

$$\Delta^{\mathcal{I}} = \{\texttt{Tom}, \texttt{Alice}, \texttt{Peter}, \texttt{math7}, \texttt{art3}, \texttt{club1}\}.$$

The extension $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ of each concept name A is:

$$
\begin{aligned}
\texttt{Student}^{\mathcal{I}} &= \{\texttt{Tom}, \texttt{Alice}\}\\
\texttt{Teacher}^{\mathcal{I}} &= \{\texttt{Alice}, \texttt{Peter}\}\\
\texttt{Person}^{\mathcal{I}} &= \{\texttt{Tom}, \texttt{Alice}, \texttt{Peter}\}\\
\texttt{Person}^{\mathcal{I}} &= \{\texttt{math7}, \texttt{art3}\}
\end{aligned}
$$

And the extension $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ of each role name r is:

$$
\begin{aligned}
\texttt{teach}^{\mathcal{I}} &= \{(\texttt{Peter}, \texttt{math7})\}\\
\texttt{attend}^{\mathcal{I}} &= \{(\texttt{Tom}, \texttt{math7}), (\texttt{Alice}, \texttt{art3}), (\texttt{Tom}, \texttt{club1})\}
\end{aligned}
$$

By definition, the extensions of the complex concepts under $\mathcal{I}$ are:

1. $(\texttt{Student} \sqcap \texttt{Teacher})^{\mathcal{I}} = \{\texttt{Alice}\}$.

   There is only one element $\texttt{Alice}$ *that is both student and teacher according to* $\mathcal{I}$*;*

2. $(\texttt{Student} \sqcup \texttt{Teacher})^{\mathcal{I}} = \{\texttt{Tom}, \texttt{Alice}, \texttt{Peter}\}$. *There are three elements* $\texttt{Tom}$, $\texttt{Alice}$ *and* $\texttt{Peter}$ *that are either student or teacher according to* $\mathcal{I}$*;*

3. $(\exists \texttt{attend.Course})^{\mathcal{I}} = \{\texttt{Tom}, \texttt{Alice}\}$. *There are two elements* $\texttt{Tom}$ *and* $\texttt{Alice}$ *that attend some course according to* $\mathcal{I}$*;*

4. $(\forall \texttt{attend.Course})^{\mathcal{I}} = \{\texttt{Alice}\}$. *There is only one element* $\texttt{Alice}$ *that do not attend anything other than the course according to* $\mathcal{I}$*;*

5. $\neg \texttt{Person} = \{\texttt{math7}, \texttt{art3}, \texttt{club1}\}$. *There are three elements* $\texttt{math7}$, $\texttt{art3}$, $\texttt{club1}$ *that are not person according to* $\mathcal{I}$*.*

Definition 5 (Axioms and ontologies) *Let* C, C' *be* $\mathcal{EL}$*-concepts, let* E, E' *be* $\mathcal{EL}^+$*-concepts, and let* F, F' *be* $\mathcal{ALC}$*-concepts. Let* $r_1 \ldots, r_n, r, t \in \mathsf{N}_R$ *be role names. Then the* $\mathcal{EL}$*-axiom* α*, the* $\mathcal{EL}^+$*-axiom* β*, and the* $\mathcal{ALC}$*-axiom* γ *are defined by the following grammar rules:*

$$\begin{aligned}
\alpha &::= C \sqsubseteq C' \mid C \equiv C' \\
\beta &::= E \sqsubseteq E' \mid E \equiv E' \mid r \sqsubseteq t \mid r_1 \circ \ldots \circ r_n \sqsubseteq t \\
\gamma &::= F \sqsubseteq F' \mid F \equiv F'
\end{aligned}$$

We call axioms of the form

1. $r \sqsubseteq t$*: a* role inclusion*;*

2. $r_1 \circ \ldots \circ r_n \sqsubseteq t$*: a* complex role inclusion*.*

An $\mathcal{L}$-ontology *is defined as a finite set of* $\mathcal{L}$*-axioms.*

Example 6 (Example 4 cont'd) *For example, an axiom could be:*

$$\alpha_1 : \texttt{Teacher} \equiv \texttt{Person} \sqcap \exists \texttt{teach.Course}.$$

This axiom means that a teacher is a person who teaches some courses.

$$\alpha_2 : \texttt{Teacher} \sqsubseteq \neg \texttt{Student}$$

is another axiom meaning that a teacher is not a student. Moreover, we could have a role inclusion of the form:

$$\alpha_3 : \texttt{teach} \sqsubseteq \texttt{attend}$$

which means that if x *teaches* y*, then* x *also attends* y*, where* x*,* y *are arbitrary individuals. We have*

1. $\mathcal{O}_1 = \{\alpha_1\}$ *is an* $\mathcal{EL}$*-ontology;*

2. $\mathcal{O}_2 = \{\alpha_1, \alpha_3\}$ *and* $\mathcal{O}_3 = \{\alpha_3\}$ *are* $\mathcal{EL}^+$*-ontologies;*

3. $\mathcal{O}_4 = \{\alpha_1, \alpha_2\}$ *is an* $\mathcal{ALC}$*-ontology.*

Note that $\mathcal{O}_5 = \{\alpha_1, \alpha_2, \alpha_3\}$ *is neither an* $\mathcal{EL}$*-ontology, nor an* $\mathcal{EL}^+$*-ontology nor an* $\mathcal{ALC}$*-ontology since* $\mathcal{O}_5$ *contains* α_2 *(an axiom with negation) and* α_3 *(an role inclusion axiom) simultaneously.*

Definition 7 *For* $\mathcal{L} = \mathcal{EL}$, $\mathcal{EL}^+$ *or* $\mathcal{ALC}$, *an interpretation* $\mathcal{I}$ *is a* ***model*** *of a* $\mathcal{L}$*-ontology* $\mathcal{O}$ *if it is compatible with all axioms in* $\mathcal{O}$. *That is, for all axioms* $C \sqsubseteq C'$ *(resp.* $C \equiv C'$, $r_1 \circ \ldots \circ r_n \sqsubseteq s \in \mathcal{O}$*), we have* $C^{\mathcal{I}} \subseteq (C')^{\mathcal{I}}$ *(resp.* $C^{\mathcal{I}} = (C')^{\mathcal{I}}$, $(r_1 \circ \ldots \circ r_n)^{\mathcal{I}} \subseteq s^{\mathcal{I}}$*). Note that* $(r_1 \circ \ldots \circ r_n)^{\mathcal{I}}$ *is defined by*

$$(r_1 \circ \ldots \circ r_n)^{\mathcal{I}} = \{(a_1, a_{n+1}) \in \Delta^{\mathcal{I}} \mid \text{there exists } a_2, \ldots, a_n \in \Delta^{\mathcal{I}} \text{ such that } (a_i, a_{i+1}) \in r^{\mathcal{I}} \text{ for } 1 \leq i \leq n\}$$

For an $\mathcal{L}$*-axiom* α, *we say* $\mathcal{O} \models \alpha$ *iff any model of* $\mathcal{O}$ *is compatible with* α.

Example 8 (Example 6 cont'd) *For the* $\mathcal{ALC}$*-ontology* $\mathcal{O}_4$, *the interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *given in Example 4 is not a model. The reason is that*

$$\mathtt{Teacher} \sqsubseteq \neg \mathtt{Student} \in \mathcal{O}_4,$$

but $\mathtt{Teacher}^{\mathcal{I}} \not\subseteq (\neg \mathtt{Student})^{\mathcal{I}}$ *since*

$$\mathtt{Teacher}^{\mathcal{I}} = \{\mathtt{Alice}, \mathtt{Peter}\},$$
$$(\neg \mathtt{Student})^{\mathcal{I}} = \{\mathtt{Peter}, \mathtt{math7}, \mathtt{art3}, \mathtt{club1}\}.$$

However, if we delete $\mathtt{Alice}$ *from the set* $\mathtt{Teacher}^{\mathcal{I}}$ *and fix the extensions of other concept names, then* $\mathcal{I}$ *is compatible with* $\mathcal{O}_4$. *Then, the new interpretation is a model of* $\mathcal{O}_4$.

We have $\mathcal{O}_4 \models \mathtt{Teacher} \sqsubseteq \mathtt{Person}$ *since*

$$\mathtt{Teacher} \equiv \mathtt{Person} \sqcap \exists \mathtt{teach.Course} \in \mathcal{O}_4.$$

Therefore, for any model $\mathcal{I}_1$ *for* $\mathcal{O}_4$, *we have :*

$$\begin{aligned} \mathtt{Teacher}^{\mathcal{I}_1} &= \mathtt{Person}^{\mathcal{I}_1} \cap (\exists \mathtt{teach.Course})^{\mathcal{I}_1} \\ &\subseteq \mathtt{Person}^{\mathcal{I}_1} \end{aligned}$$

Definition 9 (length of axioms and ontologies) *For* $\mathcal{L} = \mathcal{EL}$, $\mathcal{EL}^+$ *or* $\mathcal{ALC}$, *The* length of $\mathcal{L}$-concepts and $\mathcal{L}$-axioms *is defined inductively by:*

1. $l(A) = 1, A \in N_C$;

2. $l(C \bowtie D) = l(C) + l(D)$, *where* $\bowtie \in \{\sqcap, \sqcup, \sqsubseteq, \equiv\}$;

3. $l(Qr.C) = l(C) + 1$, *where* $Q \in \{\forall, \exists\}$;

4. $l(\neg C) = l(C)$;

5. $l(r_1 \circ \ldots \circ r_n \sqsubseteq t) = n + 1, \quad r_1, \ldots, r_n, t \in N_R$;

Then the length of an $\mathcal{L}$-ontology, *denoted* $\|\mathcal{O}\|$, *is defined by* $\|\mathcal{O}\| = \sum_{\alpha \in \mathcal{O}} l(\alpha)$.

Intuitively, the length of a concept or axioms counts the number of concept and role names. It should be noted that there are also other defintions of length of axioms and ontologies that count logical connector, such as $\sqcap$, $\sqcup$, $\sqsubseteq$, $\equiv$.

Example 10 (Example 8 cont'd) *Recall* α_1 : $\texttt{Teacher} \equiv \texttt{Person} \sqcap \exists \texttt{teach.Course}$. *We have*

$$l(\alpha_1) = l(\texttt{Teacher}) + l(\texttt{Person} \sqcap \exists \texttt{teach.Course}) = 1 + 3 = 4.$$

Therefore, $\|\mathcal{O}_1\| = l(\alpha_1) = 4$. *Similarly, we have* $l(\alpha_2) = l(\alpha_3) = 2$ *and thus* $\|\mathcal{O}_2\| = l(\alpha_1) + l(\alpha_3) = 6$.

Definition 11 (Signature) *For* $\mathcal{L} = \mathcal{EL}$, $\mathcal{EL}^+$ *or* $\mathcal{ALC}$, *let* $\mathcal{O}$ *be a* $\mathcal{L}$*-ontology, and let* C *be a* $\mathcal{L}$*-concept.*

- *We denote by sig*$(\mathcal{O})$ *(resp. sig*(C) *) the set of concept names and role names occurring in* $\mathcal{O}$ *(resp.* C*);*
- *We use sig*$_C(*)$ *(resp. sig*$_R(*)$ *) to refer to the concept (resp. role) names in sig*$(*)$*;*
- *A* signature Σ *is a set of concept and role names. Thus,* $\Sigma \subseteq N_C \cup N_R$.

2.2 . Ontology Normalization

Here, we introduced *the normalized form* of $\mathcal{ALC}$-ontologies, $\mathcal{EL}$-ontologies, and $\mathcal{EL}^+$-ontologies. This form can be used in different reasoning tasks such as *the classification of ontologies*.

Definition 12 (normalized $\mathcal{ALC}$-ontology [70]) *An* $\mathcal{ALC}$*-ontology* $\mathcal{O}$ *is* normalized *if all its axioms have one of the following forms:*

$$A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq B_1 \sqcup B_2 \sqcup \ldots \sqcup B_m$$
$$\exists r.B \sqsubseteq A$$
$$A \sqsubseteq \exists r.B$$
$$A \sqsubseteq \forall r.B$$

where $A, A_1, A_2, \ldots, A_n, B, B_1, \ldots, B_m \in N_C \cup \{\top, \bot\}, r \in N_R, n, m \geq 1$.

Similarly, normalized $\mathcal{EL}$ and $\mathcal{EL}^+$-ontologies are defined as follows.

Definition 13 (normalized $\mathcal{EL}$-ontology) *An $\mathcal{EL}$-ontology $\mathcal{O}$ is normalized if all its axioms have one of the form:*

$$\begin{aligned} A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n &\sqsubseteq B \\ \exists r.B &\sqsubseteq A \\ A &\sqsubseteq \exists r.B \end{aligned}$$

where $A, A_1, A_2, \ldots, A_n, B \in \mathsf{N}_C, r \in \mathsf{N}_R, n \geq 1$.

Definition 14 (normalized $\mathcal{EL}^+$-ontology) *An $\mathcal{EL}^+$-ontology $\mathcal{O}$ is normalized if all its axioms have one of the form:*

$$\begin{aligned} A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n &\sqsubseteq B \\ \exists r.B &\sqsubseteq A \\ A &\sqsubseteq \exists r.B \\ r &\sqsubseteq t \\ r \circ s &\sqsubseteq t \end{aligned}$$

where $A, A_1, A_2, \ldots, A_n, B \in \mathsf{N}_C, r, s, t \in \mathsf{N}_R, n \geq 1$.

By the normalization rules in Figure 2.1, we can translate every $\mathcal{L}$-ontology to its normalized form in polynomial time.

Theorem 15 ([9, 46]) *For $\mathcal{L} = \mathcal{EL}$, $\mathcal{EL}^+$ or $\mathcal{ALC}$ and every $\mathcal{L}$-ontology $\mathcal{O}$, applying the rules in Figure 2.1 yields $\mathcal{O}_1$ that satisfies the following properties:*

1. *$\mathcal{O}_1$ is normalized $\mathcal{L}$-ontology;*
2. *$sig(\mathcal{O}) \subseteq sig(\mathcal{O}_1)$;*
3. *For every axiom α with $sig(\alpha) \subseteq sig(\mathcal{O})$, $\mathcal{O} \models \alpha$ iff $\mathcal{O}_1 \models \alpha$.*

Example 16 (Example 6 cont'd) *To normalize the axiom*

$$\alpha_1 : \mathtt{Teacher} \equiv \mathtt{Person} \sqcap \exists \mathtt{teach.Course},$$

we first split it into two axioms using the first rule in Figure 2.1:

$$\begin{aligned} \alpha_1' &: \mathtt{Teacher} \sqsubseteq \mathtt{Person} \sqcap \exists \mathtt{teach.Course} \\ \alpha_1'' &: \mathtt{Person} \sqcap \exists \mathtt{teach.Course} \sqsubseteq \mathtt{Teacher} \end{aligned}$$

Then α_1', α_1'' are normalized by introducing the new concept names A, A_1:

$$\begin{aligned} \alpha_1' &\rightarrow \ \mathtt{Teacher} \sqsubseteq \mathtt{Person} \sqcap A, \quad A \sqsubseteq \exists \mathtt{teach.Course} \\ \alpha_1'' &\rightarrow \ \exists \mathtt{teach.Course} \sqsubseteq A_1, \quad \mathtt{Person} \sqcap A_1 \sqsubseteq \mathtt{Teacher} \end{aligned}$$

$$C \equiv D \rightarrow C \sqsubseteq D, D \sqsubseteq C,$$

$$\begin{aligned} C \sqsubseteq D \sqcap D' &\rightarrow C \sqsubseteq D, C \sqsubseteq D' \\ C \sqcup C' \sqsubseteq D &\rightarrow C \sqsubseteq D, C' \sqsubseteq D \end{aligned} \quad \text{(*)}$$

$$C \sqcap \forall r.C' \sqsubseteq D \rightarrow C \sqsubseteq D \sqcup \exists r.(\neg C') \quad \text{(*)}$$

$$\begin{aligned} C \sqcap \hat{C} \sqsubseteq D &\rightarrow \hat{C} \sqsubseteq \mathbf{A},\ \mathbf{A} \sqcap C \sqsubseteq D \\ C \sqsubseteq D \sqcup \hat{D} &\rightarrow C \sqsubseteq \mathbf{A} \sqcup D,\ \mathbf{A} \sqsubseteq \hat{D} \end{aligned} \quad \text{(*)}$$

$$\begin{aligned} \exists r.\hat{C} \sqsubseteq B &\rightarrow \hat{C} \sqsubseteq \mathbf{A},\ \exists r.\mathbf{A} \sqsubseteq D \\ B \sqsubseteq \exists r.\hat{C} &\rightarrow B \sqsubseteq \exists r.\mathbf{A},\ \mathbf{A} \sqsubseteq \hat{C} \\ B \sqsubseteq \forall r.\hat{C} &\rightarrow B \sqsubseteq \forall r.\mathbf{A},\ \mathbf{A} \sqsubseteq \hat{C} \end{aligned} \quad \text{(*)}$$

$$\begin{aligned} C \sqcap \neg A \sqsubseteq D &\rightarrow C \sqsubseteq D \sqcup A && (C \text{ could be } \top) \\ C \sqsubseteq D \sqcup \neg A &\rightarrow C \sqcap A \sqsubseteq D && (D \text{ could be } \bot) \end{aligned}$$

$$r_1 \circ \ldots \circ r_n \sqsubseteq t \rightarrow r_1 \circ \ldots \circ r_{n-1} \sqsubseteq \mathbf{s},\ \ \mathbf{s} \circ r_n \sqsubseteq t$$

where

- C, C', D, D' are arbitrary $\mathcal{L}$-concepts;
- $\hat{C}, \hat{D}$ are complex $\mathcal{L}$-concepts;
- B is an concept name;
- **A** is a **new introduced** concept name.
- **s** is a **new introduced** role name.

All rules are applied independently of the order of the concepts within conjunctions or disjunctions. The rules with label (*) are only useful for $\mathcal{ALC}$-ontologies.

Figure 2.1: Normalization rules

Finally, α_1 is translated to the following five normalized axioms:

$$\{\texttt{Teacher} \sqsubseteq \texttt{Person},$$
$$\texttt{Teacher} \sqsubseteq \sqcap A,$$
$$A \sqsubseteq \exists \texttt{teach.Course},$$
$$\exists \texttt{teach.Course} \sqsubseteq A_1,$$
$$\texttt{Person} \sqcap A_1 \sqsubseteq \texttt{Teacher}\}$$

These five axioms rewrite α_1 by introducing new concept names.

Definition 17 ($\mathcal{EL}$-terminology) *Given a normalized $\mathcal{EL}$-ontology $\mathcal{O}$. Let $L_A = \{C \mid C \sqsubseteq A \in \mathcal{O}\}, R_A = \{D \mid A \sqsubseteq D \in \mathcal{O}\}$.*

- *We say a concept name A is* primitive *iff (i) $L_A = \emptyset$ **or** (ii) $L_A = R_A$ and $|L_A| = 1$;*
- *We say $\mathcal{O}$ is a* terminology *iff all the concept names in $\mathcal{O}$ are primitive.*

Moreover, for a terminology $\mathcal{O}$, let $R \subseteq sig(\mathcal{O}) \times sig(\mathcal{O})$ be the relation defined by $(A, B) \in R$ iff $A \sqsubseteq C \in \mathcal{O}$ and $B \in sig(C)$. We say a terminology $\mathcal{O}$ is acyclic *iff R has no cycle, which is a sequence of elements $(A_1, A_2), (A_2, A_3), \ldots, (A_n, A_1) \subseteq R$.*

The notion of terminology defined here is the same as the one introduced in [45]. We state the definition in a different way because we use a different form of normalized ontologies.

Example 18 *The ontology $\mathcal{O}$ shown below is a normalized $\mathcal{EL}$-ontology.*

$$\mathcal{O}=\{\ \alpha_1 : A \sqsubseteq D,$$
$$\alpha_2 : D \sqsubseteq \exists r.E,$$
$$\alpha_3 : E \sqsubseteq F,$$
$$\alpha_4 : \exists t.F \sqsubseteq B,$$
$$\alpha_5 : r \sqsubseteq t,$$
$$\alpha_6 : G \sqsubseteq C,$$
$$\alpha_7 : C \sqsubseteq A$$
$$\alpha_8 : C \sqcap B \sqsubseteq G\ \ \}.$$

However, $\mathcal{O}$ is not an $\mathcal{EL}$-terminology because A is not primitive since $L_A = \{C\}, R_A = \{D\}$.

2.3 . Ontology Classification

Given a $\mathcal{L}$-ontology $\mathcal{O}$, a standard reasoning task over $\mathcal{O}$ is *classification*, which aims at finding all *subsumptions*. That is, conclusions of the form $\mathcal{O} \models A \sqsubseteq B$, where $A,\ B$ are concept names occurring in $\mathcal{O}$. Next, we introduce two methods for classification over $\mathcal{EL}^+$-ontologies and $\mathcal{ALC}$-ontologies using *inferences*.

Definition 19 (Inference and Derivation) *An **inference** ρ is a pair $\langle \rho_{pre}, \rho_{con} \rangle$ whose premise set ρ_{pre} consists of $\mathcal{L}$-axioms and conclusion ρ_{con} is a single $\mathcal{L}$-axiom. We say a sequence of inferences $\rho^1, \ldots, \rho^n$ is a **derivation** of an axiom α from $\mathcal{O}$ iff*

1. $\rho^n_{con} = \alpha$;
2. *for any $\beta \in \rho^i_{pre}, 1 \leq i \leq n$, we have $\beta \in \mathcal{O}$ or $\beta = \rho^j_{con}$ for some $j < i$.*

Denoted by $\mathcal{O} \vdash \alpha$ if there is a derivation from $\mathcal{O}$ to the axiom α.

As usual, *inference rules* are used to generate inferences. Next, we will show two different sets of inference rules for classification over $\mathcal{EL}^+$-ontologies and $\mathcal{ALC}$-ontologies.

Classification over $\mathcal{EL}^+$-ontologies

First, Table 2.1 [2, 6] is a set of inference rules over normalized $\mathcal{EL}^+$-ontologies.

$$\mathcal{R}_1: \frac{A \sqsubseteq A_1, \ldots, A \sqsubseteq A_n, \;\; A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq B}{A \sqsubseteq B}$$

$$\mathcal{R}_2: \frac{A \sqsubseteq A_1, \;\; A_1 \sqsubseteq \exists r.B}{A \sqsubseteq \exists r.B}$$

$$\mathcal{R}_3: \frac{A \sqsubseteq \exists r.B_1, \;\; B_1 \sqsubseteq B_2, \;\; \exists r.B_2 \sqsubseteq B}{A \sqsubseteq B}$$

$$\mathcal{R}_4: \frac{A_0 \sqsubseteq \exists r_1.A_1, \;\; \ldots, A_{n-1} \sqsubseteq \exists r_n.A_n, \;\; r_1 \circ \ldots \circ r_n \sqsubseteq r}{A_0 \sqsubseteq \exists r.A_n}$$

Table 2.1: Inference rules over $\mathcal{EL}^+$.

Example 20 (Example 18 cont'd) *According to the inference rules of Table 2.1, we can generate three inferences*

$$\rho = \langle \{A \sqsubseteq D, D \sqsubseteq \exists r.E\}, A \sqsubseteq \exists r.E \rangle$$
$$\rho' = \langle \{A \sqsubseteq \exists r.E, r \sqsubseteq t\}, A \sqsubseteq \exists t.E \rangle$$
$$\rho'' = \langle \{A \sqsubseteq \exists t.E, E \sqsubseteq F, \exists t.F \sqsubseteq B\}, A \sqsubseteq B \rangle$$

by inference rules $\mathcal{R}_2$, $\mathcal{R}_4$ and $\mathcal{R}_3$ respectively. Then we have

$$\mathcal{O} \vdash A \sqsubseteq B$$

since $A \sqsubseteq B$ is derivable from $\mathcal{O}$ by the sequence ρ, ρ', ρ''.

The set of inference rules in Table 2.1 is *sound* and *complete*, as shown in the following result.

Theorem 21 ([6]) *Denoted by $\mathcal{O} \vdash A \sqsubseteq B$ iff there is a derivation from $\mathcal{O}$ to the axiom α using inferences generated by rules in Table 2.1. For any normalized $\mathcal{EL}^+$ ontology $\mathcal{O}$, we have*

$$\mathcal{O} \vdash A \sqsubseteq B \text{ iff } \mathcal{O} \models A \sqsubseteq B$$

for any concept names $A, B \in \mathsf{N}_C$.

Therefore, the classification task over an $\mathcal{EL}^+$ ontology $\mathcal{O}$ can be solved by checking whether $\mathcal{O} \vdash A \sqsubseteq B$ for all $A, B \in \mathit{sig}(\mathcal{O})$ using Table 2.1. Since $\mathcal{EL}$ ontologies is a subclass of $\mathcal{EL}^+$ ontologies, the classification for $\mathcal{EL}$ ontologies can also be solved by Table 2.1.

Classification over $\mathcal{ALC}$-ontologies

[70] provides a set of inference rules for classifying normalized $\mathcal{ALC}$-ontologies.

Let H, K denote (possibly empty) conjunctions of concept names or their negations, and M, N (possibly empty) disjunctions of concept names or their negations. That is,

$$H, K := A_1 \sqcap \ldots \sqcap A_n \sqcap \neg B_1 \sqcap \ldots \sqcap \neg B_m$$
$$M, N := A_1 \sqcup \ldots \sqcup A_n \sqcup \neg B_1 \sqcup \ldots \sqcup \neg B_m$$

where $A_i, B_j \in \mathsf{N}_C \cup \{\top, \bot\}$ for $0 \leq n, 0 \leq m$, $1 \leq i \leq n, 1 \leq j \leq m$.

Then, for the rules defined in Table 2.2, the following result holds:

Theorem 22 ([70]) *Denoted by $\mathcal{O} \vdash A \sqsubseteq B$ iff there is a derivation from $\mathcal{O}$ to the axiom α using inferences generated by rules in Table 2.2. For any normalized $\mathcal{ALC}$-ontology $\mathcal{O}$, we have*

$$\mathcal{O} \vdash H \sqsubseteq \bot \text{ iff } \mathcal{O} \models H \sqsubseteq \bot$$

for any conjunction H.

$\mathcal{O} \models A \sqsubseteq B$ is equivalent to $\mathcal{O} \models A \sqcap \neg B \sqsubseteq \bot$. Therefore, the classification for an $\mathcal{ALC}$-ontology $\mathcal{O}$ can be solved by checking whether $\mathcal{O} \vdash H \sqsubseteq \bot$ for all H of the form $A \sqcap \neg B$, where $A, B \in \mathit{sig}(\mathcal{O})$.

2.4 . Justification

Definition 23 (Support and Justification) *Given an ontology $\mathcal{O}$ such that $\mathcal{O} \models A \sqsubseteq B$.*

1. *a **support** of $A \sqsubseteq B$ over $\mathcal{O}$ is a sub-ontology $\mathcal{O}' \subseteq \mathcal{O}$ such that $\mathcal{O}' \models A \sqsubseteq B$.*

$$\mathbf{R}^{+}_{\mathbf{A}}\ \frac{}{H \sqsubseteq A} : A \in H \qquad \mathbf{R}^{-}_{\mathbf{A}}\ \frac{H \sqsubseteq N \sqcup A}{H \sqsubseteq N} : \neg A \in H$$

$$\mathbf{R}^{\mathbf{n}}_{\sqcap}\ \frac{\{H \sqsubseteq N_i \sqcup A_i\}_{i=1}^{n}}{H \sqsubseteq \sqcup_{i=1}^{n} N_i \sqcup M} : \sqcap_{i=1}^{n} A_i \sqsubseteq M \in \mathcal{O}$$

$$\mathbf{R}^{+}_{\exists}\ \frac{H \sqsubseteq N \sqcup A}{H \sqsubseteq N \sqcup \exists r.B} : A \sqsubseteq \exists r.B \in \mathcal{O}$$

$$\mathbf{R}^{-}_{\exists}\ \frac{H \sqsubseteq M \sqcup \exists r.K, K \sqsubseteq N \sqcup A}{H \sqsubseteq M \sqcup B \sqcup \exists r.(K \sqcap \neg A)} : \exists r.A \sqsubseteq B \in \mathcal{O}$$

$$\mathbf{R}^{\perp}_{\exists}\ \frac{H \sqsubseteq M \sqcup \exists r.K, K \sqsubseteq \perp}{H \sqsubseteq M}$$

$$\mathbf{R}_{\forall}\ \frac{H \sqsubseteq M \sqcup \exists r.K, H \sqsubseteq N \sqcup A}{H \sqsubseteq M \sqcup N \sqcup \exists r.(K \sqcap B)} : A \sqsubseteq \forall r.B \in \mathcal{O}$$

Table 2.2: Inference rules for $\mathcal{ALC}$. Each rule is applicable iff the side condition holds.

2. *a **justification** of $A \sqsubseteq B$ is a minimal (under set-inclusion) support of $A \sqsubseteq B$ over $\mathcal{O}$.*

Example 24 (Example 20 cont'd) *$\mathcal{O}' = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$ is a support for $A \sqsubseteq B$, and thus, any super-set of $\mathcal{O}'$ is a support of $A \sqsubseteq B$. $\mathcal{O}'$ is also one of the justifications for $A \sqsubseteq B$ as for any $\mathcal{O}'' \subset \mathcal{O}'$, we have $\mathcal{O}'' \not\models A \sqsubseteq B$.*

Definition 25 (Complete Set) *We say a set of inferences S is a **complete set** for $A \sqsubseteq B$ if for any justification $\mathcal{J}$ of $A \sqsubseteq B$, we can derive $A \sqsubseteq B$ from $\mathcal{J}$ using only the inferences in S.*

Intuitively, a complete set traces all minimal proofs of a subsumption $\mathcal{O} \models A \sqsubseteq B$.

Example 26 (Example 20 cont'd) *The three inferences ρ, ρ', ρ'' provide a complete set for $A \sqsubseteq B$. Because the only justification of $A \sqsubseteq B$ is $\mathcal{O}'$ and ρ, ρ', ρ'' provide a derivation from $\mathcal{O}'$ to $A \sqsubseteq B$.*

2.5 . General Module

Given two ontologies $\mathcal{O}_1, \mathcal{O}_2$ and a *signature* $\Sigma \subseteq \mathsf{N}_C \cup \mathsf{N}_R$ of concept and role names, the *logical difference* [39] between $\mathcal{O}_1$ and $\mathcal{O}_2$ with respect to Σ is defined as the set of axioms inferred by $\mathcal{O}_1$ but not inferred by $\mathcal{O}_2$:

$$\texttt{cDiff}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_2) = \{\alpha \mid \mathit{sig}(\alpha) \subseteq \Sigma,\ \mathcal{O}_i \models \alpha,\ \mathcal{O}_j \not\models \alpha,\ \{i,j\} = \{1,2\}\ \}.$$

Next, we say $\mathcal{O}_1 \equiv_{\Sigma} \mathcal{O}_2$ iff $\texttt{cDiff}_{\Sigma}(\mathcal{O}_1, \mathcal{O}_2) = \emptyset$.

Definition 27 (General module [58]) *Given an ontology $\mathcal{O}$ and a signature Σ, an ontology $\mathcal{M}$ is a* general module *for $\mathcal{O}$ and Σ iff (i) $\mathcal{O} \equiv_\Sigma \mathcal{M}$ and (ii) $\mathcal{O} \models \mathcal{M}$.*

Two extreme cases of general modules are uniform interpolants and deductive modules.

Definition 28 (Uniform interpolant & deductive module) *Let $\mathcal{O}$ be an ontology, Σ a signature, and $\mathcal{M}$ a general module for $\mathcal{O}$ and Σ. Then, $\mathcal{M}$ is*

1. *a* uniform interpolant *for $\mathcal{O}$ and Σ if sig$(\mathcal{M}) \subseteq \Sigma$, and*
2. *a* deductive module *for $\mathcal{O}$ and Σ if $\mathcal{M} \subseteq \mathcal{O}$.*

In particular, we call a deductive module that is minimal under set-inclusion as follows.

Definition 29 (Minimal module) *A* minimal module *for an ontology $\mathcal{O}$ and a signature Σ is a sub-ontology $\mathcal{M}$ of $\mathcal{O}$ such that $\mathcal{M}$ is a minimal (under inclusion) deductive module for $\mathcal{O}$ and Σ.*

Example 30 *Let us consider the following ontology*

$$\begin{aligned}\mathcal{O} = \{\ & \alpha_1 : A \sqsubseteq \exists r.B_1,\\ & \alpha_2 : B_1 \sqsubseteq A_1 \sqcap A_2,\\ & \alpha_3 : A \sqsubseteq \exists r.A_1\\ & \alpha_4 : \exists r.B_2 \sqsubseteq A_2,\\ & \alpha_5 : A_3 \sqcap A_4 \sqsubseteq B_2\ \}.\end{aligned}$$

and the signature $\Sigma = \{A,\ A_1,\ A_2,\ A_3,\ A_4,\ r\}$. Then, a uniform interpolant for $\mathcal{O}$ and Σ could be

$$\begin{aligned}\mathcal{U}_\Sigma^{\mathcal{O}} = \{\ & \beta_1 : A \sqsubseteq \exists r.(A_1 \sqcap A_2),\\ & \beta_3 : \exists r.(A_3 \sqcap A_4) \sqsubseteq A_2\ \}.\end{aligned}$$

Also, $\mathcal{O} \setminus \{\alpha_3\}$ is a deductive module for $\mathcal{O}$ and Σ, it is also a minimal module because no proper subset of $\mathcal{O} \setminus \{\alpha_3\}$ is a deductive module. The above uniform interpolant and deductive module are both general modules.

3 - Computing Ontology Justifications

As stated in the introduction, justifications play an essential role in the study of Description Logics. They have been widely explored for different tasks like debugging ontologies [2, 3, 4, 37, 57, 34] and computing ontology modules [16, 13, 14]. In this chapter, based on a graph representation of $\mathcal{EL}^+$-ontologies, we propose a method for computing all justifications for a given conclusion by introducing a set of inference rules (called *H-rules*). The advantage of our setting is that most of the time, it reduces the number of *inferences* (wrt H-rules) required to derive a given conclusion. This accelerates the computation of justifications relying on these inferences.

This chapter is organized as follows. First, we discuss related works in Section 3.1 and the motivation of our work in Section 3.2. Then, in Section 3.3, we introduce our inference rules *H-rule* based on the hypergraph representation of $\mathcal{EL}^+$-ontologies. In Section 3.4, we present the main algorithm `minH` for computing all justifications based on our inference rules. Finally, we evaluate our method with real-world ontologies in Section 3.5. The proofs of all results can be found in Section 3.6.

3.1 . Related Work

In this section, we discuss some existing works whose purpose is to compute justifications for a given conclusion. In this chapter, we always assume that $\mathcal{O}$ is an $\mathcal{EL}^+$-ontology.

3.1.1 . Computing one justification

Let us start with the methods that compute one justification for a given conclusion α. An easy way to compute one justification is to remove redundant axioms one by one. Here, an axiom $\beta \in \mathcal{O}$ is redundant if the given conclusion α can still be derived from $\mathcal{O}$ without β (i.e., $\mathcal{O} \setminus \{\beta\} \models \alpha$). The details are shown in Algorithm 1.

In Algorithm 1, whether $J \setminus \{\beta\} \models \beta$ in Line 3 can be determined by efficient reasoners such as ELK[36] and HermiT[25]. The complexity of Algorithm 1 depends on the complexity of this entailment checking (i.e., whether $J \setminus \{\beta\} \models \beta$). For $\mathcal{EL}^+$-ontologies, Algorithm 1 takes polynomial time as it is known that entailment checking over $\mathcal{EL}^+$-ontologies takes polynomial time with respect to the size of input ontology [6].

Note that Algorithm 1 is non-deterministic in the sense that we may obtain different justifications of α depending on the enumeration order of axioms at Line 2 of Algorithm 1. For example:

Algorithm 1 : Computing one justification

input : a conclusion α, an ontology $\mathcal{O}$
output : a justification J of α

```
1 J ← O;
2 for each axiom β ∈ O do
3 |   if J \ {β} ⊨ α then
4 |   |   J ← J \ {β}
5 |   end
6 end
7 return J
```

Example 31 *Considering the ontology*

$$\mathcal{O} = \{ \begin{array}{l} \beta_1 : A \sqsubseteq B, \\ \beta_2 : A \sqsubseteq C, \\ \beta_3 : C \sqsubseteq B \end{array} \},$$

and the conclusion

$$\alpha = A \sqsubseteq B.$$

Then the following two different computations are possible:

1. *If we enumerate axioms in $\mathcal{O}$ in the order:*

   $$\beta_1, \beta_2, \beta_3$$

   at Line 2 of Algorithm 1, then the resulting justification of α output by Algorithm 1 is $\{\beta_2, \beta_3\}$;

2. *If we enumerate axioms in $\mathcal{O}$ in the order:*

   $$\beta_2, \beta_3, \beta_1$$

   at Line 2 of Algorithm 1, then the resulting justification of α output by Algorithm 1 is $\{\beta_1\}$.

Therefore, different computation could leads to different resulting justifications.

3.1.2 . Computing all justifications

Compared to computing one justification, computing all justifications is much more complex and reveals itself to be a challenging problem [63]. In the worst case, there might be exponentially many different justifications with respect to the size of given ontology for a given conclusion α. Here is an example.

Example 32 *[62] Consider $\mathcal{O}$ to be the $\mathcal{EL}^+$-ontology consisting of the axioms defined below:*

$$\begin{aligned} A_0 \sqsubseteq A_1,\ & A_0 \sqsubseteq A_2, \\ A_{2k-1} \sqsubseteq A_{2k+1},\ & A_{2k-1} \sqsubseteq A_{2k+2}, \quad (\text{ for } 1 \leq k \leq n-1) \\ A_{2k} \sqsubseteq A_{2k+1},\ & A_{2k} \sqsubseteq A_{2k+2}, \quad (\text{ for } 1 \leq k \leq n-1) \\ A_{2n-1} \sqsubseteq A_{2n+1},\ & A_{2n} \sqsubseteq A_{2n+1}. \end{aligned}$$

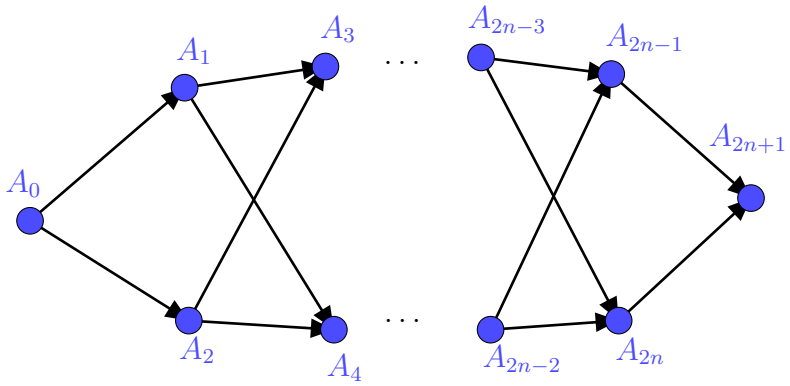


Figure 3.1: Graph representation of $\mathcal{O}$

$\mathcal{O}$ can be represented by a graph as shown in Figure 3.1. Then $\mathcal{O}$ contains $4n$ axioms, and there are 2^n different justifications of the conclusion $A_0 \sqsubseteq A_{2n+1}$.

There are mainly two approaches for computing all justifications for a given conclusion, the *black-box* approach and the *glass-box* approach.

Black-box approach The simple idea shared by *black-box* methods [34, 33, 10] is to compute all the justifications by going through all possible sub-ontologies of $\mathcal{O}$ that derive the given conclusion α. From each generated sub-ontology, they extract a (new) justification using Algorithm 1. Since Algorithm 1 only relies on some existing reasoner, black-box approaches can be adapted easily to any existing Description Logics as long as a reasoner is available.

For example, a naive black-box method could go through all subsets $\mathcal{O}$ and filter out those sub-ontologies that could not derive α. However, this requires doing exponentially many checks since there are exponentially many different subsets of $\mathcal{O}$. Therefore, such a process could be highly time-consuming. Some search strategies could be applied to help one accelerate this process. The main idea of these strategies is to filter out some redundant subsets according to the known results. For example, if a justification J has already been computed for the given conclusion α, then no subset $\mathcal{O}' \subset J$ can be a justification of α. Thus these subsets

$\mathcal{O}'$ should be ignored in future computations. Similarly, if it is known already that $\mathcal{M} \not\models \alpha$ (i.e., $\mathcal{M}$ does not derive α) for a subset $\mathcal{M} \subset \mathcal{O}$, then obviously, any subset of $\mathcal{M}$ should be ignored since they can not derive α. These optimizations can significantly reduce the number of subsets that need to be checked through the computation process of all justifications using the naive method. There are also other strategies for searching for new justifications, such as the one proposed in [34] based on Reiter's *Hitting Set Tree*. Details are referred to [30].

Glass-box Approach The glass-box approaches achieve better performance than *black-box* approaches by taking advantage of the reasoning process. The main drawback of glass-box approaches is that they are limited to less expressive ontology languages such as $\mathcal{EL}^+$-ontologies. For expressive ontologies such as $\mathcal{ALC}$, the entire reasoning process could grow exponentially and make it hard to tracking the reasoning process.

Among different glass-box approaches, SAT-based methods [2, 3, 4, 37, 57] achieve the best performance. These SAT-based methods are based on an algorithm that generates a new justification different from known ones using SAT tools. To be capable of using SAT tools, they compute a *complete set* (recall Definition 25 on page 21) for the given conclusion, then encode this complete set as *Horn-clauses*. Below, we propose a simple example that illustrates how those methods proceeds.

Example 33 *Let $\mathcal{O}$ be the $\mathcal{EL}^+$-ontology showed below:*

$$\begin{aligned}\mathcal{O} = \{\ & \alpha_1 : A \sqsubseteq D,\\ & \alpha_2 : D \sqsubseteq \exists r.E,\\ & \alpha_3 : E \sqsubseteq F,\\ & \alpha_4 : \exists t.F \sqsubseteq B,\\ & \alpha_5 : r \sqsubseteq t,\\ & \alpha_6 : G \sqsubseteq C,\\ & \alpha_7 : C \sqsubseteq A\\ & \alpha_8 : C \sqcap B \sqsubseteq G\ \ \}.\end{aligned}$$

Then, $\mathcal{O} \models G \sqsubseteq D$ as for all models $\mathcal{I}$ of $\mathcal{O}$, we have

- $G^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ *by the axiom* α_6*;*
- $C^{\mathcal{I}} \subseteq A^{\mathcal{I}}$ *by the axiom* α_7*;*
- $A^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ *by the axiom* α_1*.*

Let us show now how the SAT-based methods compute all justifications for the conclusion $G \sqsubseteq D$.

Note that one could derive $G \sqsubseteq D$ by different proof systems, which leads to different kinds of complete sets of $G \sqsubseteq D$. Here, we use the proof system defined by Table 2.1. Therefore, the complete set of $G \sqsubseteq D$ consists of inferences generated by inference rules in Table 2.1.

The computation of justifications of $G \sqsubseteq D$ consists of 5 steps as follows.

Step 1 *Compute the complete set of $G \sqsubseteq D$ with respect to Table 2.1. This complete set consists of the following two inferences:*

$$\rho_1 = \langle \{\alpha_6, \alpha_7\}, G \sqsubseteq A \rangle,$$
$$\rho_2 = \langle \{G \sqsubseteq A, \alpha_1\}, G \sqsubseteq D \rangle,$$

Step 2 *Encode inferences ρ_1, ρ_2 by a set of Horn-clauses as follows. Map each axiom occurring in ρ_1, ρ_2 to a* (propositional) variables *through the map f defined below:*

$$f(\alpha_6) = \boldsymbol{p}_1,$$
$$f(\alpha_7) = \boldsymbol{p}_2,$$
$$f(\alpha_1) = \boldsymbol{p}_3,$$
$$f(G \sqsubseteq A) = p_4,$$
$$f(G \sqsubseteq D) = p_5.$$

The variables $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3$ corresponding to axioms $\alpha_6, \alpha_7, \alpha_1 \in \mathcal{O}$ are called ***answer variables****. To distinguish answer variables from other literals, answers literals $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3$ are written in bold. The literals based on answer variables are called* ***answer literals*** *(e.g., $\boldsymbol{p}_1$, $\neg \boldsymbol{p}_1$).*

Now, the inferences ρ_1, ρ_2 can be encoded by the following set of Horn-clauses:

$$\mathcal{C} = \{\neg \boldsymbol{p}_1 \vee \neg \boldsymbol{p}_2 \vee p_4,\ \neg p_4 \vee \neg \boldsymbol{p}_3 \vee p_5\}.$$

Step 3 *Find all justifications of $G \sqsubseteq D$ using SAT tools as follows. Let*

$$\mathcal{C}_0 = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3\}, \qquad \mathcal{C}_1 = \mathcal{C} \cup \{\neg p_5\}.$$

Here, p_5 is the variable corresponding to the conclusion $G \sqsubseteq D$. Then, all justifications of $G \sqsubseteq D$ are equivalent to minimal unsatisfiable subsets *$S \subseteq \mathcal{C}_0$. That is, S is a minimal subset in $\mathcal{C}_0$ that satisfies (i) $S \cup \mathcal{C}_1$ is unsatisfiable; and (ii) for any proper subset $S' \subset S$, $S' \cup \mathcal{C}_1$ is satisfiable.*

Specifically, it holds that $S \subseteq \mathcal{C}_0$ is a minimal unsatisfiable subset of $\mathcal{C}_0, \mathcal{C}_1$ iff

$$\mathcal{M}_S := \{g^{-1}(\boldsymbol{p}) \mid \boldsymbol{p} \in S\}$$

is a justification of $G \sqsubseteq D$. Therefore, all justifications of $G \sqsubseteq D$ can be computed by enumerating all minimal unsatisfiable subsets in $S \subseteq \mathcal{C}_0$ using SAT tools.

In this simple example, the unique minimal unsatisfiable subset is $S_0 = \mathcal{C}_0$, and thus $\mathcal{M}_{S_0} = \{\alpha_1, \alpha_6, \alpha_7\}$ is the unique justification of $G \sqsubseteq D$.

One could derive the given conclusion using different proof systems (e.g., [35], [25]), which leads to different complete sets for the same conclusion. To accelerate the process of searching new justifications (i.e., Steps 3 and 4 in the above example), many different SAT-based methods have been explored (e.g., Beacon [2], EL2MCS [3], SATPin [57]) using different SAT tools.

Compared to the Black-box approach, the SAT-based approach has mainly two advantages:

1. They record the reasoning process as Horn-clauses.

2. Their computation could benefit from the highly optimized SAT tools.

However, it should be noting that most SAT tools employ a *hitting-set* procedure [51] to enumerate minimal unsatisfiable subsets, which involves computing a byproduct known as the *minimal correction subset* [51]. This introduces some redundancy in the process. To get rid of such redundancy, the state-of-the-art method PULi proposed a method that generates justifications using resolution. This method allows for a more efficient enumeration of all justifications. We illustrate how PULi [37] proceeds by the following example.

Example 34 (Example 33 cont'd) *Next, we compute all justifications of the conclusion $G \sqsubseteq D$ as in PULi. First, we compute a complete set and encode it by the Horn-clause set $\mathcal{C}$ given in Example 33. Then, PULi enumerates all justifications of $G \sqsubseteq D$ by resolution over $\mathcal{C}$ as follows. Recall that p_5 is the propositional variable corresponding to the given conclusion $G \sqsubseteq D$.*

Step 1 We do resolution exhaustively over

$$\mathcal{C} \cup \{\neg p_5\}$$

by applying the following resolution rule exhaustively for all variable p that occurring in $\mathcal{C} \cup \{\neg p_5\}$:

$$\frac{\neg p \vee c_1, \quad p \vee c_2}{c_1 \vee c_2},$$

Denote the resulting Horn-clauses set as $\mathcal{C}^*$. In this example, we have

$$\begin{aligned}\mathcal{C}^* = \mathcal{C} \cup \{ \ & \neg p_5, \\ & \boldsymbol{p}_1 \vee \neg \boldsymbol{p}_2 \vee \neg \boldsymbol{p}_3 \vee p_5, \\ & \neg \boldsymbol{p}_3 \vee \neg p_4, \\ & \neg \boldsymbol{p}_1 \vee \neg \boldsymbol{p}_2 \vee \neg \boldsymbol{p}_3 \ \}.\end{aligned}$$

Step 2 According to [37], every justification of $G \sqsubseteq D$ corresponds to a clause $c \in \mathcal{C}^*$ such that:

(a) c contains only answer literals;

(b) c *is minimal. I.e., there is no other clause* $c' \in \mathcal{C}^*$ *such that* $c \neq c'$ *and all the literals in* c' *are also in* c.

In this simple example, there is only one Horn-clause $\neg \boldsymbol{p}_1 \vee \neg \boldsymbol{p}_2 \vee \neg \boldsymbol{p}_3$ *in* $\mathcal{C}^*$ *that satisfies those two requirements. Therefore, the set of all justifications of* $G \sqsubseteq D$ *is*

$$\{\{\alpha_1,\ \alpha_6,\ \alpha_7\}\},$$

where $\alpha_6,\ \alpha_7,\ \alpha_1$ *are axioms encoded by the answer variables* $\boldsymbol{p}_1,\ \boldsymbol{p}_2,\ \boldsymbol{p}_3$ *respectively.*

3.2 . Motivation of Our Method

In the real world, ontologies are always huge. For instance, the *Snomed CT* ontology contains more than 300,000 axioms. As a result, the complete set of a given conclusion could be extremely large, making it challenging to extract all the justifications over them even with efficient SAT tools or resolution. Therefore, one may consider to accelerate the computation of justifications by reducing the size of the *complete set*. One way to do that is to identify, for a given conclusion, a particular part of the ontology relevant to the given conclusion. For example, one can extract a small sub-ontology (e.g., a *locality-based module*[27]) that preserves all the justifications for a given conclusion. Also, one could filter out redundant inferences that do not contribute to the derivation of the given conclusion (e.g. using a *goal-directed tracing algorithm* [35]). Another way is to develop a new proof system that produces more concise complete sets for the given conclusion. For example, the state-of-the-art algorithm, PULi [37] uses the proof system proposed in [36]. Their experiments on real-world ontologies show that their proof system generates more concise complete sets than others.

However, for the simple ontology

$$\mathcal{O} = \{A_1 \sqsubseteq A_2,\ A_2 \sqsubseteq A_3,\ \ldots,\ A_{n-1} \sqsubseteq A_n\}$$

and the conclusion $A_0 \sqsubseteq A_n$, the complete set of $A_0 \sqsubseteq A_n$ computed by proof systems defined in Table 2.1 contains at least $n-1$ inferences. This complete set can not be reduced further, even with the previously mentioned optimizations. From this observation, we decided to explore a new method to handle such situation better.

Our motivation comes from the following observations. Let us look carefully at the ontology $\mathcal{O}$ above. If we regard each A_i as a graph node N_{A_i}, then we can associate to $\mathcal{O}$ a directed graph $\mathcal{G}_{\mathcal{O}}$, which consists of edges of the form

$$N_{A_i} \rightarrow N_{A_{i+1}}.$$

It turns out that each justification for the conclusion $A_0 \sqsubseteq A_n$ is equivalent to a path from N_{A_0} to N_{A_n} in $\mathcal{G}_{\mathcal{O}}$, and here we have only one such path. Thus, we can

compute all justifications of $A_0 \sqsubseteq A_n$ by an efficient algorithm that outputs paths. We can easily extend this idea to general $\mathcal{EL}^+$-ontologies because most of the $\mathcal{EL}^+$-axioms can be interpreted as direct edges. However, there is one case deserves to introuduce hyperedges. That is, for axioms of the form $B_1 \sqcap \ldots \sqcap B_n \sqsubseteq A$ (see Definition 37). This example inspired us to explore a new proof system based on a hypergraph representation of $\mathcal{EL}^+$-ontologies. Roughly, our proof system consists of inferences rules that are built from elementary paths (i.e., directed paths consist of directed edges) of the hypergraph. Based on these inference rules, we reformulate justifications to particular paths called *H-paths*. Then, computing all the justifications for a given conclusion is made using such H-paths.

Our new proof system create a more concise complete set for a given conclusion. For example, the *complete set* of the previous ontology $\mathcal{O}$ and the conclusion $A_0 \sqsubseteq A_n$ is now reduced to two inferences (no matter the value of n):

1. The first inference states that all the justifications of $A_0 \sqsubseteq A_n$ are elementary paths from node N_{A_0} to node N_{A_n} in the corresponding hypergraph of $\mathcal{O}$;

2. The second inference represents the unique elementary path from N_{A_0} to N_{A_n}.

By using this new proof system, we develop for $\mathcal{EL}^+$ a new SAT-based glass-box method for computing all justifications of a given conclusion. The source of the improvement provided by our method is twofold. On the one hand, it comes from the fact that the elementary paths are pre-computed while extracting the inferences and that the existing algorithms like depth-first search can efficiently compute such paths. On the other hand, yet as a consequence, decreasing the size of the *complete sets* of inferences leads to smaller inputs for SAT tools. Therefore, we can accelerate the extraction of justifications from the *complete set*.

3.3 . Hypergraph-based Inference Rules

3.3.1 . Hypergraph

A (directed) hypergraph [5] $\mathcal{H} = \{\mathcal{V}, \mathcal{E}\}$ consists of a finite node set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ and an finite edge set $\mathcal{E} = \{e_1, e_2 \ldots, e_m\}, e_i = (T(e_i), f(e_i))$, where $T(e_i) \subseteq \mathcal{V}$ is a subset and $f(e_i) \in \mathcal{V}$ is a node.

Definition 35 *Given a hypergraph $\mathcal{H} = \{\mathcal{V}, \mathcal{E}\}$, assume $S \subseteq \mathcal{V}$ and $v \in \mathcal{V}$. A hyper-path from S to v is a sequence $h = [e_1, e_2, \ldots, e_n]$ of hyperedges satisfying:*

1. $f(e_n) = \{v\}$;
2. *for* $i = 1, \ldots, n$, $T(e_i) \subseteq S \cup \{f(e_1; \ldots, f(e_{i-1}))\}$;
3. *for* $i = 1, \ldots, n$, $f(e_i) \in \bigcup_{i<j\leq n} T(e_j)$.

If $v \in T(e_1)$, we say h is a loop. If h does not contain any loop, we say h is loop-free.

Example 36 *An example of hypergraph is shown in Figure 3.2. The sequence $h = [e_1, e_2, e_3, e_4]$ is a hyperpath from $S = \{N_1, N_3\}$ to node N_6.*

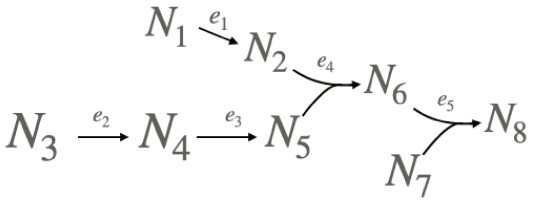


Figure 3.2: An example of hypergraph.

3.3.2 . H-inferences

Definition 37 *The hypergraph $\mathcal{G}_\mathcal{O}$ associated to the $\mathcal{EL}^+$-ontology $\mathcal{O}$ is defined by $\mathcal{G}_\mathcal{O} = (\mathcal{V}_\mathcal{O}, \mathcal{E}_\mathcal{O})$ where*

1. *the set of nodes $\mathcal{V}_\mathcal{O} = \{N_A, N_r, N_{\exists r.A} \mid A \in N_C, r \in N_R\}$;*
2. *the set of edges $\mathcal{E}_\mathcal{O}$ is defined by $g(\mathcal{O})$ where g is the multi-valued mapping defined in Table 3.1.*

Given a hyperedge e of $\mathcal{E}_\mathcal{O}$, the inverse image of e, $g^{-1}(e)$, is defined in the obvious manner. For a set E of hyperedges, $g^{-1}(E) = \cup_{e \in E}\ g^{-1}(e)$.

	α	$g(\alpha)$
1.	$A \sqsubseteq B$	$(\{N_A\}, \{N_B\})$,
2.	$B_1 \sqcap \ldots \sqcap B_m \sqsubseteq A$	$(\{N_{B_1}, \ldots, N_{B_m}\}, \{N_A\})$
3.	$A \sqsubseteq \exists r.B$	$(\{N_A\}, \{N_{\exists r.B}\})$
4.	$\exists r.B \sqsubseteq A$	$(\{N_{\exists r.B}\}, \{N_A\})$,
5.	$r \sqsubseteq s$	$(\{N_r\}, \{N_s\})$, $(\{N_{\exists r.A}\}, \{N_{\exists s.A}\})$ for all $A \in sig(\mathcal{O})$
6.	$r \circ s \sqsubseteq t$	$(\{N_r, N_s\}, \{N_s, N_t\})$

Table 3.1: The multivalued map f from axioms to hyperedges

Figure 3.3 shows how the hyperedges defined in Table 3.1 are represented. It should be noted that the hyperedge in Case 6 is constructed for the purpose of allowing the reconstruction of the input axioms $r \circ s \sqsubseteq t$, which do not have much geometric meaning.

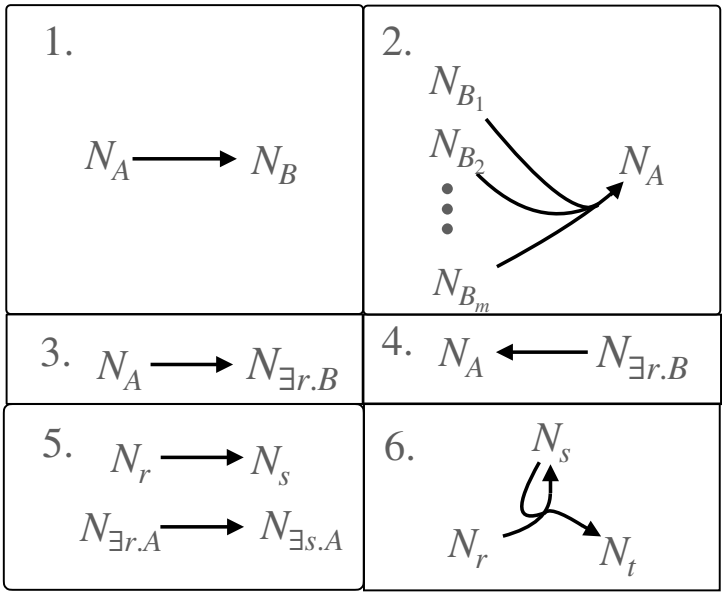


Figure 3.3: Illustration of hyperedges

Example 38 *Consider the $\mathcal{EL}^+$-ontology $\mathcal{O}$ defined by*

$$\begin{aligned}\mathcal{O}=\{\ & \alpha_1 : A \sqsubseteq D,\\ & \alpha_2 : D \sqsubseteq \exists r.E,\\ & \alpha_3 : E \sqsubseteq F,\\ & \alpha_4 : \exists t.F \sqsubseteq B,\\ & \alpha_5 : r \sqsubseteq t,\\ & \alpha_6 : G \sqsubseteq C,\\ & \alpha_7 : C \sqsubseteq A\\ & \alpha_8 : C \sqcap B \sqsubseteq G\ \ \}.\end{aligned}$$

Then, the hypergraph $\mathcal{G}_\mathcal{O}$ associated to $\mathcal{O}$ is shown in Figure 3.4, where

$$\begin{aligned}e_0 &= (\{N_C\}, \{N_A\}),\\ e_1 &= (\{N_A\}, \{N_D\}),\\ e_2 &= (\{N_D\}, \{N_{\exists r.E}\}),\end{aligned}$$

and so on. We also have

$$\begin{aligned}g^{-1}(e_0) &= C \sqsubseteq A,\\ g^{-1}(e_1) &= A \sqsubseteq D,\\ g^{-1}(e_2) &= D \sqsubseteq \exists r.E,\end{aligned}$$

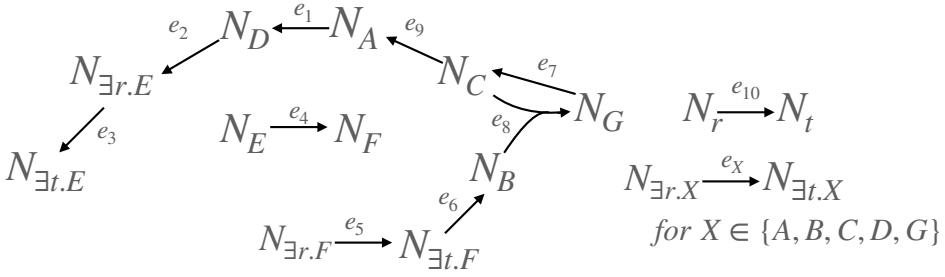


Figure 3.4: The hypergraph associated with the ontology $\mathcal{O}$.

As for standard graphs, a path (next called **regular path**) from nodes N_1 to N_2 in a hypergraph is a sequence of edges:

$$e_0 = (S_1^0, S_2^0), e_1 = (S_1^1, S_2^1), \ldots, e_n = (S_1^n, S_2^n) \tag{3.1}$$

where $S_1^0 = \{N_1\}, S_2^n = \{N_2\}$ and $S_2^{i-1} = S_1^i, 1 \leq i \leq n$. Next, the **existence of a regular path** from N_X to N_Y in a hypergraph $\mathcal{G}_\mathcal{O}$ is denoted

$$N_X \rightsquigarrow N_Y.$$

For regular paths, we have the following result:

Proposition 39 *Given $\mathcal{O}$ and its associated hyper-graph $\mathcal{H}_\mathcal{O}$, if there exists a regular path from N_X to N_Y then $\mathcal{O} \models X \sqsubseteq Y$.*

Proof. The proof is provided on page 50. □

Now, we introduce hypergraph-based inferences which are based on the existence of regular paths as follows.

Definition 40 *Given a hypergraph $\mathcal{G}_\mathcal{O}$, Table 3.2 gives a set of inference rules called **H-rules**. Inferences based on H-rules are called **H-inferences**. Next, we denote by*

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y \text{ (or simply } N_X \overset{h}{\rightsquigarrow} N_Y)$$

the fact that $N_X \overset{h}{\rightsquigarrow} N_Y$ can be derived from $\mathcal{G}_\mathcal{O}$ using the H-inferences.

Example 41 (Example 38 cont'd) *As shown in Figure 3.4, we have*

$$N_A \rightsquigarrow N_{\exists r.E},$$
$$N_E \rightsquigarrow N_F,$$
$$N_{\exists r.F} \rightsquigarrow N_B$$

$$\mathcal{H}_0: \frac{N_X \rightsquigarrow N_Y}{N_X \overset{h}{\rightsquigarrow} N_Y}$$

$$\mathcal{H}_1: \frac{N_X \overset{h}{\rightsquigarrow} N_{B_1}, \ldots, N_X \overset{h}{\rightsquigarrow} N_{B_m},\ \ N_A \rightsquigarrow N_Y,\ \ (\{N_{B_1}, \ldots, N_{B_m}\}, \{N_A\})}{N_X \overset{h}{\rightsquigarrow} N_Y}$$

$$\mathcal{H}_2: \frac{N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1},\ \ N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2},\ \ N_{\exists r.B_2} \rightsquigarrow N_Y}{N_X \overset{h}{\rightsquigarrow} N_Y}$$

$$\mathcal{H}_3: \frac{N_X \overset{h}{\rightsquigarrow} N_{\exists r.A_1},\ \ N_{A_1} \overset{h}{\rightsquigarrow} N_{\exists s.A_2},\ \ N_{\exists t.A_2} \rightsquigarrow N_Y,\ \ (\{N_r, N_s\}, \{N_s, N_t\})}{N_X \overset{h}{\rightsquigarrow} N_Y}$$

Table 3.2: H-rules over $\mathcal{G}_\mathcal{O} = (\mathcal{V}_\mathcal{O}, \mathcal{E}_\mathcal{O})$.

from the existence of regular paths. Then we can derive $N_A \overset{h}{\rightsquigarrow} N_B$ from $\mathcal{G}_\mathcal{O}$ by the H-rules $\mathcal{H}_0$, $\mathcal{H}_0$ and $\mathcal{H}_2$ which generate the H-inferences ρ^1, ρ^2, ρ^3, where

$$\rho^1 = \langle \{N_A \rightsquigarrow N_{\exists r.E}\}, N_A \overset{h}{\rightsquigarrow} N_{\exists r.E} \rangle \qquad (\mathcal{H}_0)$$

$$\rho^2 = \langle \{N_E \rightsquigarrow N_F\}, N_E \overset{h}{\rightsquigarrow} N_F \rangle \qquad (\mathcal{H}_0)$$

$$\rho^3 = \langle \{N_A \overset{h}{\rightsquigarrow} N_{\exists r.E}, N_E \overset{h}{\rightsquigarrow} N_F, N_{\exists r.F} \rightsquigarrow N_B\}, N_A \overset{h}{\rightsquigarrow} N_B \rangle \qquad (\mathcal{H}_2)$$

Note that the first rule $\mathcal{H}_0$, the initialization rule, makes regular paths the elementary components of H-rules. Moreover, Proposition 42 formally states that our H-inference system does not need the transitive inference rule:

$$\frac{N_X \overset{h}{\rightsquigarrow} N_Z,\ \ N_Z \overset{h}{\rightsquigarrow} N_Y}{N_X \overset{h}{\rightsquigarrow} N_Y}.$$

Proposition 42 *If $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Z$ and $\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_Y$ then $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.*

Proof. The proof is provided on page 50. □

3.3.3 . Completeness and soundness of H-inferences

The following result is the main result of this section. It states the equivalence of $N_X \overset{h}{\rightsquigarrow} N_Y$ derivation (by Table 3.2) and ontology entailment for $X \sqsubseteq Y$, and thus states that our H-rules are sound and complete for $\mathcal{EL}^+$-ontology.

Theorem 43 *If $\mathcal{O}$ is an $\mathcal{EL}^+$-ontology, then*

$$\mathcal{O} \models X \sqsubseteq Y \text{ iff } \mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y,$$

where X, Y are concepts of either form A or $\exists r.B$.

Proof. The proof is provided on page 52. □

3.3.4 . Extracting justifications from $\mathcal{G}_\mathcal{O}$

Now, we formally define H-path as a hypergraph representation of classical derivations based on H-rules. The reader should pay attention to the fact that H-paths are not classical hyperpaths [22]. Next, for the sake of homogeneity, we consider a regular path from N_X to N_Y as the set of its edges and denote it as $P_{X,Y}$, $P'_{X,Y}$, ...

Definition 44 (H-paths) *In the hypergraph $\mathcal{G}_\mathcal{O}$, an H-path $H_{X,Y}$ from N_X to N_Y is a set of edges recursively generated by the following composition rules:*

0. *A regular path $P_{X,Y}$ is an H-path from N_X to N_Y;*

1. *If the following three conditions holds,*

   - $e = (\{N_{B_1}, \ldots, N_{B_m}\}, \{N_A\}) \in \mathcal{V}_\mathcal{O}$;
   - H_{X,B_i} *are H-paths for* $i = 1..m$;
   - $P_{A,Y}$ *is a regular path,*

   then

   $$H_{X,B_1} \cup \ldots \cup H_{X,B_m} \cup P_{A,Y} \cup \{e\}$$

   is an H-path from N_X to N_Y;

2. *If the following two conditions holds,*

   - $H_{X,\exists r.B_1}, H_{B_1,B_2}$ *are H-paths;*
   - $P_{\exists r.B_2,Y}$ *is a regular path,*

   then

   $$H_{X,\exists r.B_1} \cup H_{B_1,B_2} \cup P_{\exists r.B_2,Y}$$

   is an H-path from N_X to N_Y;

3. *If the following three conditions holds,*

   - $e = (\{N_r, N_s\}, \{N_s, N_t\}) \in \mathcal{V}_\mathcal{O}$;
   - $H_{X,\exists r.A_1}, H_{A_1,\exists s.A_2}$ *are H-paths;*
   - $P_{\exists t.A_2,B}$ *is a regular path,*

   then

   $$H_{X,\exists r.A_1} \cup H_{A_1,\exists s.A_2} \cup P_{\exists t.A_2,B} \cup \{e\}$$

   is an H-path from N_X to N_Y.

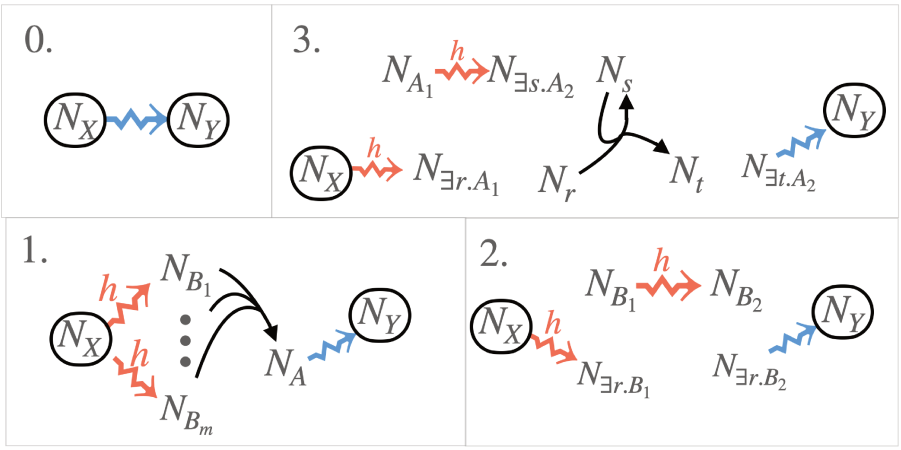


Figure 3.5: Structure of H-paths from N_X to N_Y

Figure 3.5 gives an illustration of H-paths: the blue arrows $\rightsquigarrow$ correspond to regular paths, the red arrows $\overset{h}{\rightsquigarrow}$ correspond to H-paths. It is straightforward to compare composition rules building H-paths with H-rules building derivations in Table 3.2. One may also consider H-paths as deviation-trees with leaves corresponding to the edges in $\mathcal{G}_\mathcal{O}$. However, our approach provides a more direct characterization of justifications as shown by Theorem 45.

We say that an H-path $H_{X,Y}$ is **minimal** if there is no H-path $H'_{X,Y}$ such that $H'_{X,Y} \subset H_{X,Y}$.

Now, we are ready to explain how H-paths and justifications are related. We can compute justifications from minimal H-paths as stated below:

Theorem 45 *Let X, Y be of either form A or $\exists r.B$. Let*

$$\mathcal{S} = \{g^{-1}(H_{X,Y}) \mid H_{X,Y} \text{ is a minimal H-path from } N_X \text{ to } N_Y\}.$$

Then we have

$$\mathcal{J}_\mathcal{O}(X \sqsubseteq Y) = \{s \in \mathcal{S} \mid s' \not\subset s, \forall s' \in \mathcal{S}\}.$$

That is, all justifications for $X \sqsubseteq Y$ are the minimal subsets in $\mathcal{S}$.

Proof. The proof is provided on page 55. □

Example 46 (Example 38 cont'd) *Now, we show how to compute justifications for $A \sqsubseteq B$ by finding H-paths from N_A to N_B. The regular paths from N_A to $N_{\exists r.E}$ and from N_E to N_F produce two H-paths*

$$H_{A,\exists tE} = \{e_1, e_2, e_3\},$$
$$H_{E,F} = \{e_4\}.$$

Then, applying the third composition rule with $H_{A,\exists t.E}, H_{E,F}$ and $P_{\exists t.F,B} = \{e_6\}$, we get

$$H_{A,B} = \{e_1, e_2, e_3, e_4, e_6\},$$

which is the unique H-path from N_A to N_B. Thus, by Theorem 45, we have

$$\{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\}$$

is the only justification for $A \sqsubseteq B$.

3.4 . Implementation: Computing justifications

3.4.1 . Main algorithm

In this section, given an ontology $\mathcal{O}$ and its associated hypergraph $\mathcal{G}_{\mathcal{O}}$, we present `minH` (Algorithm 2) that computes all justifications for $X_0 \sqsubseteq Y_0$ using the minimal H-paths from N_{X_0} to N_{Y_0} in $\mathcal{G}_{\mathcal{O}}$. The algorithm `minH` proceeds in two steps described below.

Algorithm 2 : `minH`

input : $X \sqsubseteq Y$
output : J: $\mathcal{J}_{\mathcal{O}}(X \sqsubseteq Y)$.

```
1 J ← ∅;
2 U ← CompleteH(N_X ⇝^h N_Y);
3 min_hpaths ← resolution(clauses(U));
4 for h ∈ min_hpaths do
5 |   if g^{-1}(h') ⊄ g^{-1}(h) for any h' ∈ min_hpaths then
6 |   |   J.add(g^{-1}(h))
7 |   end
8 end
```

Step 1. First, at Line 2 of Algorithm 3, `minH` computes a *complete set* of inferences $\mathcal{U}$ for $N_{X_0} \overset{h}{\rightsquigarrow} N_{Y_0}$ using `CompleteH` (i.e., Algorithm 3). Here, $\mathcal{U}$ is complete set in the sense that for any H-path $H_{X,Y}$, we can derive $N_X \overset{h}{\rightsquigarrow} N_Y$ using inferences in $\mathcal{U}$ from the edge set $H_{X,Y}$.

The set $\mathcal{U}$ of inferences output by `CompleteH` consists of the following two parts:

1. **Line 3-12 of Algorithm 3:** In this part, we first compute all H-inferences whose conclusion is $N_{X_0} \overset{h}{\rightsquigarrow} N_{Y_0}$ using `trace_one_turn` (i.e., Algorithm 4), where H-inferences generated by H-rules $\mathcal{H}_1$, $\mathcal{H}_2$, $\mathcal{H}_3$ are captured at line 2-7, line 8-13, line 14-17 of Algorithm 4, respectively.

   We apply this process recursively until there are no new terms that emerge. That is, for all new terms of the form $N_{X_1} \overset{h}{\rightsquigarrow} N_{Y_1}$ in the result of previous computation, we apply of `trace_one_turn` with $N_{X_1} \overset{h}{\rightsquigarrow} N_{Y_1}$ as input and

collect all the obtained H-inferences in $\mathcal{U}$.

2. **Line 13-17 of Algorithm 3:** In this part, we add to $\mathcal{U}$ all new inferences of the form

$$\langle \{e_1, e_2, \ldots, e_n\}, N_{X_2} \rightsquigarrow N_{Y_2} \rangle,$$

where $\{e_1, e_2, \ldots, e_n\}$ is a regular path from N_{X_2} to N_{Y_2} over $\mathcal{G}_\mathcal{O}$ such that $N_{X_2} \rightsquigarrow N_{Y_2}$ appeared in $\mathcal{U}$. Those new inferences capture all the possible regular paths indicated by $N_{X_2} \rightsquigarrow N_{Y_2}$. Here, the algorithm `getPath` in line 14 is the depth-first search algorithm that computes all regular paths from N_X to N_Y in $\mathcal{G}_\mathcal{O}$ with input (N_X, N_Y).

Algorithm 3 : `CompleteH`

input : $N_X \overset{h}{\rightsquigarrow} N_Y$
output : $\mathcal{U}$: a complete set of inferences for $N_X \overset{h}{\rightsquigarrow} N_Y$.

1 $\mathcal{U}$, history, Q $\leftarrow \emptyset$; `// Q is a queue`
2 Q.*add*$(N_X \overset{h}{\rightsquigarrow} N_Y)$;
3 **while** Q $\neq \emptyset$ **do**
4 $\quad N_{X_1} \overset{h}{\rightsquigarrow} N_{Y_1} \leftarrow$ Q.*takeNext*();
5 $\quad$ history.*add*$(N_{X_1} \overset{h}{\rightsquigarrow} N_{Y_1})$;
6 $\quad \mathcal{U} \leftarrow \mathcal{U} \bigcup$ `trace_one_turn`$(N_{X_1} \overset{h}{\rightsquigarrow} N_{Y_1})$;
7 $\quad$ **for** $N_{X_2} \overset{h}{\rightsquigarrow} N_{Y_2}$ *appearing in* `trace_one_turn`$(N_{X_1} \overset{h}{\rightsquigarrow} N_{Y_1})$ **do**
8 $\quad\quad$ **if** $N_{X_2} \overset{h}{\rightsquigarrow} N_{Y_2} \notin$ history *and* $N_{X_2} \overset{h}{\rightsquigarrow} N_{Y_2} \notin$ Q **then**
9 $\quad\quad\quad$ Q.add$(N_{X_2} \overset{h}{\rightsquigarrow} N_{Y_2})$
10 $\quad\quad$ **end**
11 $\quad$ **end**
12 **end**
13 **for** $N_{X_2} \rightsquigarrow N_{Y_2}$ *appearing in* $\mathcal{U}$ **do**
14 $\quad$ **for** $p = \{e_1, e_2, \ldots, e_n\} \in$`getPath`$(N_{X_2}, N_{Y_2})$ **do**
15 $\quad\quad \mathcal{U}$.*add*$(\langle \{e_1, e_2, \ldots, e_n\}, N_{X_2} \rightsquigarrow N_{Y_2} \rangle)$;
16 $\quad$ **end**
17 **end**

Step 2. Then Algorithm `minH` computes all justifications for $X_0 \sqsubseteq Y_0$ as follows:

1. **Line 3 of Algorithm 2:** `minH` computes all minimal H-paths from N_{X_0} to N_{Y_0} using resolution, which is developed by PULi (available at `https:`

Algorithm 4 : `trace_one_turn`

input : $N_X \overset{h}{\rightsquigarrow} N_Y$

output : all H-inferences whose conclusion is $N_X \overset{h}{\rightsquigarrow} N_Y$.

result $\leftarrow \emptyset$;
$\mathcal{P}_1(X,Y) \leftarrow \{(\{N_{B_1},\ldots,N_{B_m}\},\{N_A\}) \in \mathcal{E}^{\mathcal{O}} \mid \mathcal{O} \models X \sqsubseteq A \sqsubseteq Y\}$;
for $(\{N_{B_1},\ldots,N_{B_m}\},\{N_A\}) \in \mathcal{P}_1(X,Y)$ **do**
  if `path`$(N_A, N_Y) \neq \emptyset$ *or* $Y = A$ **then**
    result.*add*$(\langle \Big\{ N_X \overset{h}{\rightsquigarrow} N_{B_1}, \ldots, N_X \overset{h}{\rightsquigarrow} N_{B_m}, N_A \rightsquigarrow N_Y,$
    $\big(\{N_{B_1},\ldots,N_{B_m}\},\{N_A\}\big) \Big\}, N_X \overset{h}{\rightsquigarrow} N_Y \rangle)$;
  end
end

$\mathcal{P}_2(X,Y) \leftarrow \{(r,B_1,B_2) \mid \mathcal{O} \models X \sqsubseteq \exists r.B_1,\ B_1 \sqsubseteq B_2, \exists r.B_2 \sqsubseteq Y\}$;
for $(r,B_1,B_2) \in \mathcal{P}_2(X,Y)$ **do**
  if `path`$(N_{\exists r.B_2}, N_Y) \neq \emptyset$ *or* $Y = \exists r.B_2$ **then**
    result.*add*$(\langle \{N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1}, N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2}, N_{\exists r.B_2} \rightsquigarrow N_Y\}, N_X \overset{h}{\rightsquigarrow} N_Y \rangle)$;
  end
end

$\mathcal{P}_3(X,Y) \leftarrow \{(r,s,t,A_1,A_2) \mid r \circ s \sqsubseteq t \in \mathcal{O},$
$\mathcal{O} \models X \sqsubseteq \exists r.A_1, A_1 \sqsubseteq \exists s.A_2, \exists t.A_2 \sqsubseteq Y\}$; **for** $(r,s,t,A_1,A_2) \in \mathcal{P}_3(X,Y)$ **do**
  if `path`$(N_{\exists t.A_2}, N_Y) \neq \emptyset$ *or* $Y = \exists t.A_2$ **then**
    result.*add*$(\langle \Big\{ N_X \overset{h}{\rightsquigarrow} N_{\exists r.A_1}, N_{A_1} \overset{h}{\rightsquigarrow} N_{\exists s.A_2}, N_{\exists t.A_2} \rightsquigarrow N_Y,$
    $\big(\{N_r, N_t\},\{N_s, N_t\}\big) \Big\}, \ N_X \overset{h}{\rightsquigarrow} N_Y \rangle)$;
  end
end

//github.com/liveontologies/pinpointing-experiments), over the clauses generated from $\mathcal{U}$ as illustrated in Example 34. Here, a literal p is associated with each edge e, each $N_X \overset{h}{\rightsquigarrow} N_Y$, and each $N_X \rightsquigarrow N_Y$ appeared in $\mathcal{U}$. The answer variables are those associated with edges.

2. **Line 4-8 of Algorithm 2:** `minH` computes justifications by mapping back all the minimal H-paths and select the subset-minimal sets as stated in Theorem 45.

By examining Definition 44 and `CompleteH` (i.e., Algorithm 3), we can establish a direct mapping between the composition rules in Definition 44 and the steps of `CompleteH`. Specifically, rules 1, 2, and 3 of Definition 44 correspond respectively to lines 2-7, 8-13, and 14-17 of `trace_one_turn` (i.e., Algorithm 4), which is exacurated in lines 6 of `CompleteH`. Rule 0, on the other hand, corresponds to lines 15 of `CompleteH`. Therefore, we capture all minimal H-paths from N_X to N_Y by applying `CompleteH` and the `resolution` function in lines 2 and 3 of `minH` (i.e., Algorithm 2), respectively.

As a direct consequence of Theorem 45, we obtain the following result:

Corollary 47 *The `minH` algorithm computes all justifications for any subsumption $\mathcal{O} \models X \sqsubseteq Y$ over $\mathcal{O}$.*

Example 48 (Example 38 cont'd) *Assume $X_0 = G$ and $Y_0 = D$ are the inputs of `minH`. Then at line 2 of `minH`, we have $\mathcal{U} = \{\rho^1, \rho^2\}$, where*

1. *$\rho^1 = \langle \{N_G \rightsquigarrow N_D\}, N_G \overset{h}{\rightsquigarrow} N_D \rangle$ is the H-inference obtained by `CompleteH` (line 3-12);*
2. *$\rho^2 = \langle \{e_0, e_1, e_8\}, N_G \rightsquigarrow N_D \rangle$ is produced from the regular paths obtained by `CompleteH` (line 13-17).*

Let us associate to each term of inferences in $\mathcal{U}$ the propositional variables

$$\boldsymbol{p}_0 : e_0,\ \boldsymbol{p}_1 : e_1,\ \boldsymbol{p}_2 : e_8,\ p_3 : N_G \rightsquigarrow N_D,$$

where $\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_2$ are answer variables. $p_4 : N_G \overset{h}{\rightsquigarrow} N_D$. The clause encoding the inferences in $\mathcal{U}$ are

$$\texttt{clauses}(\mathcal{U}) = \{\neg p_3 \vee p_4,\ \neg \boldsymbol{p}_0 \vee \neg \boldsymbol{p}_1 \vee \neg \boldsymbol{p}_2 \vee p_3\}.$$

By resolution over $\texttt{clauses}(\mathcal{U})$, we obtain

$$\mathit{min\_hpaths} = \{\{e_0, e_1, e_8\}\}$$

at line 3 of `minH`. Then the output of `minH` is

$$J = \{\{\alpha_1, \alpha_6, \alpha_7\}\},$$

which is the set of all justifications (here, a unique justification) for $G \sqsubseteq D$.

3.4.2 . Optimization

Below we present two optimizations that have been implemented in order to accelerate the computation of all justifications.

1. In Algorithm 4, for the H-inference added at Line 5, we require that there exists at least one regular path from N_A to N_Y that does not contain an edge

$$e_i = (\{N_A\}, \{N_{B_i}\}) \text{ for some } 1 \leq i \leq m.$$

Otherwise, as shown in Figure 3.6, H-paths corresponding to this H-inference are not minimal, as they all contain one H-path from N_X to N_Y of the form

$$H_{X,B_i} \cup (P_{A,Y} - \{e_i\}).$$

In the same spirit, we require that the H-path from N_X to N_{B_i} does not pass by N_A.

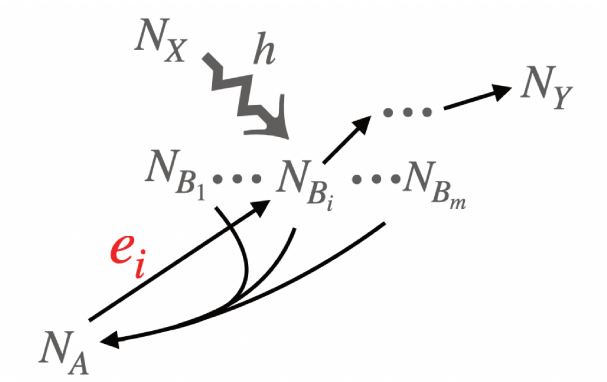


Figure 3.6: illustration of optimization 1

2. If we have an H-path

$$H_{A,B} = H_{A,\exists r.B_1} \cup H_{B_1,B_2} \cup P_{\exists r.B_2,B},$$

where

$$H_{A,\exists r.B_1} = H_{A,\exists r.C} \cup H_{C,B_1}. \tag{3.2}$$

then $H_{C,B_2} = H_{C,B_1} \cup H_{B_1,B_2}$ is also an H-path and we have

$$H_{A,B} = H_{A,\exists r.C} \cup H_{C,B_2} \cup P_{\exists r.B_2,B}.$$

The two different ways to decompose $H_{A,B}$ above are already considered in Line 8 when executing Algorithm 4 with the input $N_A \overset{h}{\rightsquigarrow} N_B$. It means that the decomposition (3.2) is redundant. We can avoid such redundancy by requiring $\exists r.B_2 \neq Y$ at Line 11.

3.5 . Experiments

To evaluate and validate our approach, we compare `minH` (a prototype available at https://gitlab.lisn.upsaclay.fr/yang/minH) with PULi [37], the state-of-the-art algorithm for computing justifications at this moment. Both methods compute all justifications based on resolution but with different inference rules generated in different ways. PULi uses a complete set (next denoted by *elk*) generated by the ELK reasoner [36], whose inference rules are variant of those in Table 2.1 and applicable for non-normalized $\mathcal{EL}^+$ ontologies. Our method uses the complete set $\mathcal{U}$ generated by Step 1 of `minH`, described in Section 3.4.1. To analyze the performance of our setting, we use the following two measures:

1. We compare the size of *elk* with that of $\mathcal{U}$;
2. We compare the time cost of PULi with that of `minH`.

All the experiments were conducted on a machine with an INTEL Xeon 2.6 GHz and 128 GiB of RAM.

The experiments were processed with four different ontologies (available at https://osf.io/9sj8n/, https://www.snomed.org/):

1. go-plus;
2. galen7;
3. SnomedCT (version Jan. 2015), denoted as snt2015;
4. SnomedCT (version Jan. 2021), denoted as snt2021.

All the non-$\mathcal{EL}^+$ axioms were deleted from those ontologies. Here, go-plus, galen7 are the same ontologies used in [37]. The number of axioms, concepts, relations, and queries for each ontology are shown in Table 3.3.

Next a **query** refers to a *direct subsumption* $A \sqsubseteq B$. We say $A \sqsubseteq B$ is a direct subsumption iff $\mathcal{O} \models A \sqsubseteq B$ and there is no concept name $A' \neq A, B$ such that $\mathcal{O} \models A \sqsubseteq A', A' \sqsubseteq B$. Direct subsumptions are computed by a reasoner supporting ontology classification. In our experiments, for each of the four ontologies, the set of all justifications $J_{\mathcal{O}}(A \sqsubseteq B)$ is computed for each query $A \sqsubseteq B$. A query $A \sqsubseteq B$ is called **trivial** iff all minimal H-paths from N_A to N_B are regular paths, otherwise, the query is called **non-trivial**.

Comparing complete sets: $\mathcal{U}$ vs. *elk*

We summarize our results in Table 3.4 and Figure 3.7. Table 3.4 shows that on all four ontologies, $\mathcal{U}$ is much smaller than *elk* on average. Especially on galen7, the difference between *elk* and $\mathcal{U}$ is even up to 50 times. The gap is even more significant for the median value since a large part of the queries is trivial. However,

	go-plus	galen7	snt2015	snt2021
#axioms	105557	44475	311466	362638
#concepts	57173	28482	311480	361226
#roles	157	964	58	132
#queries	90443	91332	461854	566797

Table 3.3: Summary of sizes of the input ontologies.

		go-plus	galen7	snt2015	snt2021
elk	average	166.9	3602.0	114.7	67.3
	median	43	3648	10	31
	max	7919	81501	2357	2226
$\mathcal{U}$	average	34.2	74.6	29.4	19.4
	median	4	5	1	3
	max	7772	24103	2002	6452
#non-trivial query		50272	62470	195082	304321

Table 3.4: Summary of sizes of *elk*, $\mathcal{U}$.

the gap is much smaller for the maximal size. On snt2021, the largest $\mathcal{U}$ is three times larger in size than that of *elk*.

In Figure 3.7, for a given query, if the complete set *elk* contains fewer inference rules than $\mathcal{U}$, the corresponding blue point is below the red line. The percentage of such cases are:

- 0.34% for go-plus;
- 0.066% for galen7;
- 0.79% for snt2015;
- 1.01% for snt2021.

It means that for most of the queries, the corresponding $\mathcal{U}$ is smaller than *elk*.

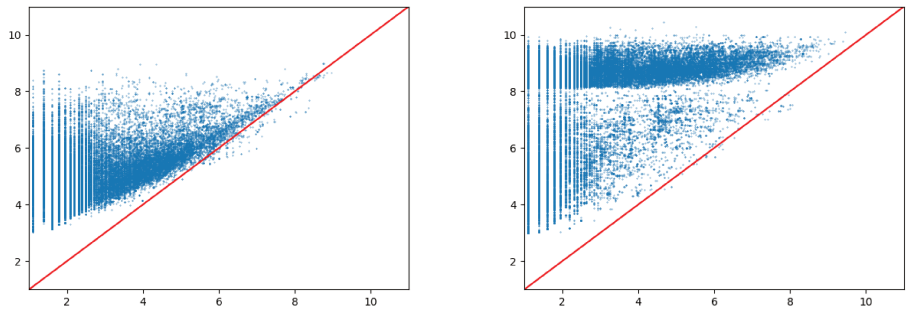
As shown in Table 3.4 and in Figure 3.7, sometimes `minH` generates bigger complete sets $\mathcal{U}$ than PULi. It may happen when, for example, there might be exponentially many different regular paths resulting from the computation process of `minH`. Also, $\mathcal{U}$ can be bigger than *elk* when the regular paths involved are simple. For example, if most regular paths contain only one edge, then the complete set $\mathcal{U}$ includes many clauses of the form

$$\neg p_e \vee p_{N_A \rightsquigarrow N_B},$$

which happens because H-rules use regular paths. Indeed, the clause

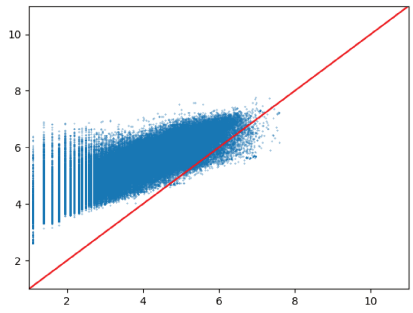
$$\neg p_e \vee p_{N_A \rightsquigarrow N_B}$$

is redundant since we can omit this clause by replacing $p_{N_A \rightsquigarrow N_B}$ by p_e. For *elk*, this does not happen.

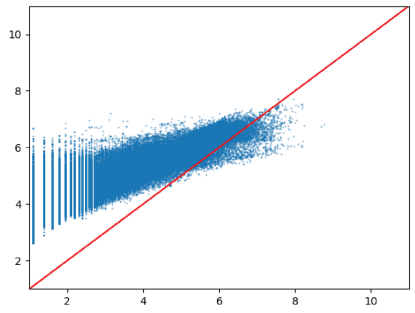


(a) go-plus

(b) galen7



(c) snt2015



(d) snt2021

Figure 3.7: The x-axis (resp. y-axis) is $\#|\mathcal{U}|$ (resp. $\#|elk|$) in logarithmic scale; the red line is the diagonal.

Comparing time cost: `minH` vs. PULi

In the following, we only compare the time cost on non-trivial queries. For trivial queries, all H-path are regular paths. Thus all the justifications have already been enumerated by `path` in `minH`. It is also easy for PULi to compute all the justifications for trivial queries.

We set a time limit of 60s for each query. The timed-out queries contribute 60s to the total time cost. To compare `minH` with PULi, we test all three different strategies, *threshold, top down, bottom up* of the resolution algorithm proposed

		threshold	***top down***	***bottom up***
total times(s) (PULi/`minH`)	go-plus	8482.7/**7350.3**	16352.3/**8935.6**	73629.1/**17950.9**
	galen7	10796.2/**3681.4**	43372.9/**10607.9**	36300.9/**3156.3**
	snt2015	1956.8/**973.5**	13650.7/**1107.6**	15058.3/**11392.2**
	snt2021	**2116.1**/2222.6	11573.9/**2361.6**	19402.1/**17154.9**
timed-out queries (PULi/`minH`/both)	go-plus	116/**103** /93	202/**117**/114	935/**223**/223
	galen7	48/**43**/43	370/**123**/120	228/**38**/38
	snt2015	**0**/3/0	49/**3**/3	96/**88**/83
	snt2021	**2**/8/1	39/**9**/9	144/**133**/128

Table 3.5: Total time cost and number of timed-out queries.

in [37]. We summarize in Table 3.5 the total time cost (top) and the timed-out queries (bottom). Figure 3.8 gives the comparisons over queries that are successful both for `minH` and PULi.

As shown in Table 3.5, when using the threshold strategy, `minH` is more time consuming in total (+5%) on snt2021, and `minH` has more timed-out queries than PULi on snt2015 and snt2021. This is in part due to the fact that $\mathcal{U}$ is larger than *elk* for relatively many queries on snt2015 and snt2021 as shown in Figure 3.7. For the remaining 11 cases, `minH` performs better than PULi in terms of total time cost and the number of timed-out queries. Especially on galen7, the gap between the two methods is even up to ten times for the total time cost. We can see from Table 3.5 that the threshold strategy performs the best for PULi on all four ontologies. This strategy is also the best strategy for `minH` except for galen7, for which the *bottom up* strategy is the best for `minH`.

For each strategy detailed in Figure 3.8, the black curve (the ordered time costs of `minH` on successful queries) is always below the red curve (the ordered time costs of PULi on successful queries) for all the ontologies. This suggests that `minH` spends less time over successful queries. Also, most of the green points are below the red lines, which suggests that `minH` performs better than PULi most of the time for a given query. In some cases, we can see that PULi is more efficient than `minH`. One of the reasons might be as follows. Note that when computing justifications by resolution, we have to compare two different clauses and delete the redundant one (i.e., the non-minimal one). If too many long regular paths occur, `minH` might be time-consuming because of these comparisons.

Figure 3.9 shows the relation between the size of the complete set and time cost for computing justifications. In every sub-figure of Figure 3.9, each blue point represents a test signature with

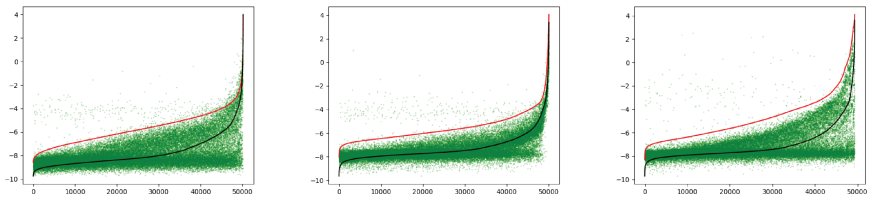
1. the log value of the size of the complete set $\mathcal{U}$ generated by `minH` (**resp.** *elk* generated by `PULi`) as the x-coordinate;

2. the log value of the time-cost of `minH` (**resp.** `PULi`), using *threshold* strategy, as the y-coordinate.

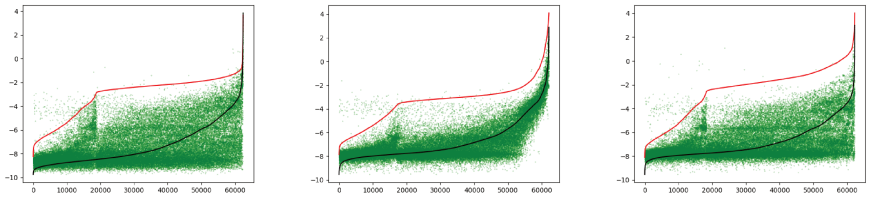
The red line represents the linear regression result for those blue points. When the complete set is not so big, we can see that the log value of the time cost grows linear with respect to the log value of the size of the generated complete set for both `minH` and `PULi`. Therefore, the time cost growth polynomially with respect to the size of complete set since we have:

$$\log t = k \cdot \log s + b \Rightarrow t = e^b \cdot s^k.$$

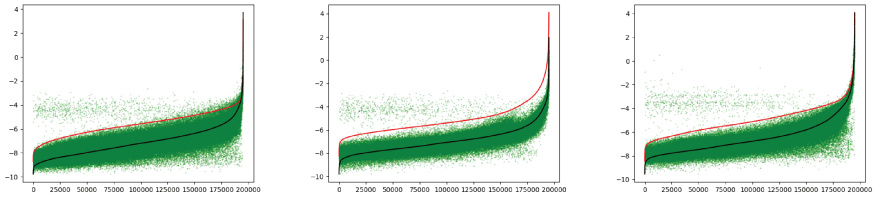
However, in the extreme case when the complete set is big enough, the log value of the time cost could grow exponentially with respect to the log value size of the generated complete set. Therefore, when the complete set is big enough, the



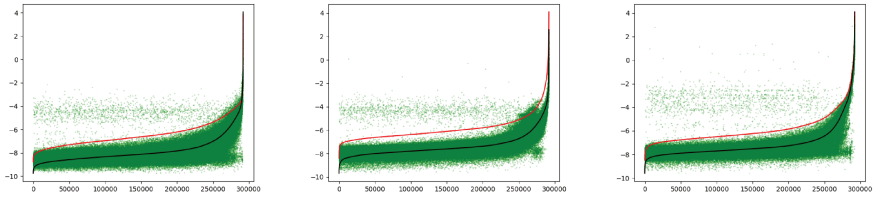
(a) go-plus



(b) galen7



(c) snt2015

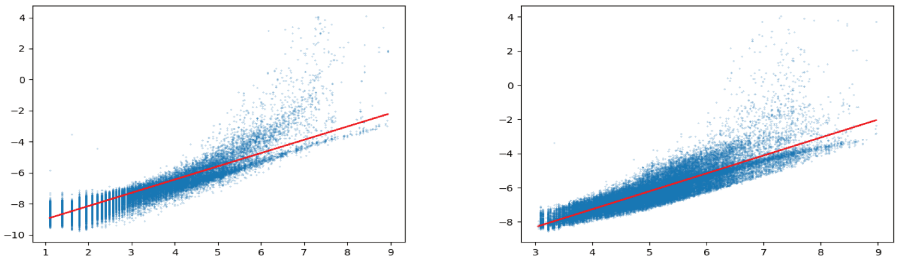


(d) snt2021

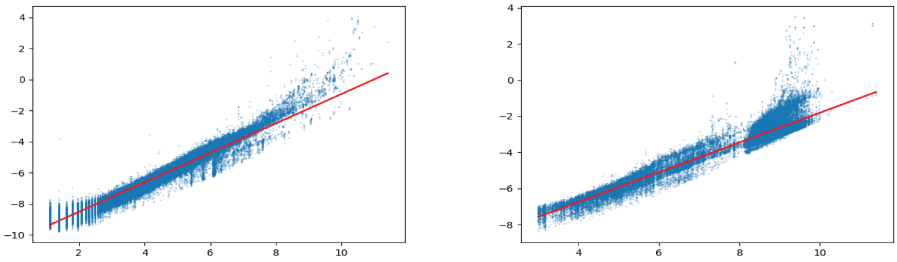
Figure 3.8: For each line, the left, middle, and right charts correspond to *threshold, top down, bottom up* strategies, respectively. The y-axis is the log value of time (s). The ascending ordered time cost of PULi (resp. `minH`) is shown by the red (resp. black) curve. The green points (`minH`) follow the exact same query ordering as the red line.

time cost growth exponentially with respect to of the size of complete set since:

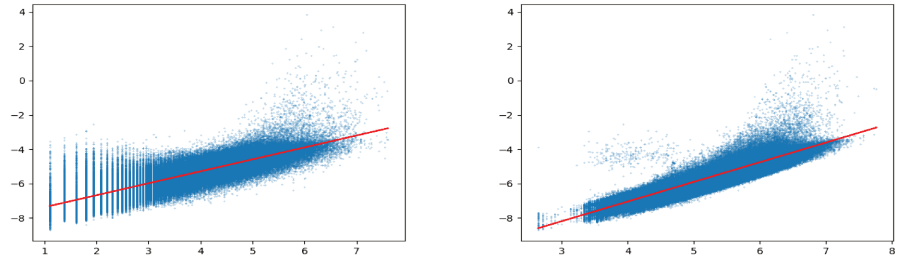
$$\log t = e^{\log s} \Rightarrow t = e^s.$$



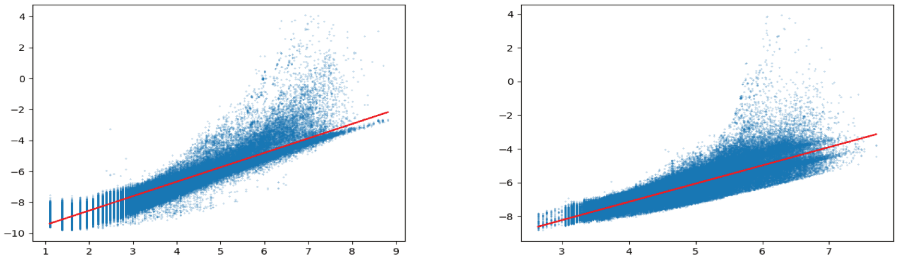
(a) go-plus (left: `minH`, right: `PULi`)



(b) galen7 (left: `minH`, right: `PULi`)



(c) snt2015 (left: `minH`, right: `PULi`)



(d) snt2021 (left: `minH`, right: `PULi`)

Figure 3.9: Relation between the size of the complete set (x-axis) and time cost for computing justifications (y-axis) in logarithmic scale.

3.6 . Proofs

Proposition 49 *Given $\mathcal{O}$ and its associated hyper-graph $\mathcal{H}_\mathcal{O}$, if there exists a regular path from N_X to N_Y then $\mathcal{O} \models X \sqsubseteq Y$.*

Proof. Assume we have a regular path like Equation 3.1 (page 33) from N_X to N_Y, then we can find a sequence $\{X_i\}_{i=0}^n$ such that

$$X_0 = X, X_n = Y \text{ and } N_{X_i} \in S_2^{i-1} \cap S_1^i, 1 \leq i \leq n.$$

Then $N_{X_{i-1}} \in S_1^i, N_{X_{i+1}} \in S_2^i$ for all $1 \leq i \leq n$. By the construction of hyper-edges, we know that for any model $\mathcal{I}$ of $\mathcal{O}$, $X_{i-1}^\mathcal{I} \subseteq X_i^\mathcal{I}$. Therefore,

$$X_0^\mathcal{I} \subseteq X_n^\mathcal{I}$$

for all models $\mathcal{I}$ of $\mathcal{O}$, so $\mathcal{O} \models X_0 \sqsubseteq X_n$, i.e., $\mathcal{O} \models X \sqsubseteq Y$. □

Proposition 50 *If $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Z$ and $\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_Y$ then $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.*

In the following, when $\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_Y$, we denote by $d_H(Z, Y)$ the length of the smallest derivation(s) of $N_Z \overset{h}{\rightsquigarrow} N_Y$ using the H-rules of Table 3.2 (page 34). We have $d_H(Z, Y) \geq 1$.

Proof. The proof of Proposition 42 is done by induction on $d_H(Z, Y)$.

1. **Assume** $d_H(Z, Y) = 1$. Then $N_Z \overset{h}{\rightsquigarrow} N_Y$ is inferred through $\mathcal{H}_0$, and therefore $N_Z \rightsquigarrow N_Y$.

   Assume $N_X \overset{h}{\rightsquigarrow} N_Z$ is derived by a sequence of inferences $\rho^1, \rho^2, \ldots, \rho^n$. Then ρ^n is of the form

   $$\langle \{\ldots, N_{X'} \rightsquigarrow N_Z\}, N_X \rightsquigarrow N_Z \rangle.$$

   We have $N_{X'} \rightsquigarrow N_Y$ because $N_Z \rightsquigarrow N_Y$. Therefore, we can obtain a new inference ρ^{new} by replacing Z by Y in $N_{X'} \rightsquigarrow N_Z, N_X \rightsquigarrow N_Z$. Then ρ^{new} has the following form

   $$\langle \{\ldots, N_{X'} \rightsquigarrow N_Y\}, N_X \rightsquigarrow N_Y \rangle.$$

   Therefore, we have $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$ because $N_X \overset{h}{\rightsquigarrow} N_Y$ is derived by the inference sequence

   $$\rho^1, \rho^2, \ldots, \rho^{n-1}, \rho^{new}.$$

   Proposition 42 is proved when $d_H(Z, Y) = 1$.

2. **Assume that Proposition 42 holds for $d_H(Z, Y) < k$. Let us prove it holds for $d_H(Z, Y) = k$.** Assume again that

   $$\rho^1, \rho^2, \ldots, \rho^n$$

   is one shortest inference sequence that derive $N_Z \overset{h}{\rightsquigarrow} N_Y$.

   If ρ^n is of type $\mathcal{H}_0$, then $N_Z \rightsquigarrow N_Y$ and it has been proved.

   If not, there are 3 different cases:

(a) ρ^n **is of type** $\mathcal{H}_1$**.** Then

$$\rho^n = \langle \{N_Z \overset{h}{\rightsquigarrow} N_{A_1},\ \ldots,\ N_Z \overset{h}{\rightsquigarrow} N_{A_m}, N_{A_0} \rightsquigarrow N_Y, \\ (\{N_{A_1}, N_{A_2}, \ldots, N_{A_m}\}, \{N_{A_0}\})\},\ N_Z \rightsquigarrow N_Y \rangle$$

$N_Z \overset{h}{\rightsquigarrow} N_{A_i}$ can be derived using inferences $\rho^1, \rho^2, \ldots, \rho^{n-1}$, and thus we have $d_H(Z, A_i) < k$. By assumption, we have $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{A_i}$ because

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Z,$$
$$\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_{A_i} \\ \text{and } d_H(Z, A_i) < k.$$

Then we can derive $N_X \overset{h}{\rightsquigarrow} N_Y$ by first deriving $N_X \overset{h}{\rightsquigarrow} N_{A_i}$, then applying the following inference

$$\rho^{new} = \langle \{N_X \overset{h}{\rightsquigarrow} N_{A_1}, \ldots, N_X \overset{h}{\rightsquigarrow} N_{A_m}, N_{A_0} \rightsquigarrow N_Y, \\ (\{N_{A_1}, N_{A_2}, \ldots, N_{A_m}\}, \{N_{A_0}\})\},\ N_X \rightsquigarrow N_Y \rangle$$

Therefore, we have $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.

(b) ρ^n **is of type** $\mathcal{H}_2$**.** Then

$$\rho^n = \langle \{N_Z \overset{h}{\rightsquigarrow} N_{\exists r.B_1},\ N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2},\ N_{\exists r.B_2} \rightsquigarrow N_Y\},\ N_Z \rightsquigarrow N_Y \rangle$$

$N_Z \overset{h}{\rightsquigarrow} N_{\exists r.B_1}$ can be derived using inferences $\rho^1, \rho^2, \ldots, \rho^{n-1}$, and thus we have $d_H(Z, \exists r.B_1) < k$.

By assumption, we know $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1}$ because

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Z,$$
$$\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_{\exists r.B_1} \\ \text{and } d_H(Z, \exists r.B_1) < k.$$

Then $N_X \overset{h}{\rightsquigarrow} N_Y$ can be derived by

- first deriving $N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1},\ N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2}$;
- then applying the following inference

$$\rho^{new} = \langle \{N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1},\ N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2},\ N_{\exists r.B_2} \rightsquigarrow N_Y\},\ N_X \rightsquigarrow N_Y \rangle$$

Therefore, we have $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.

(c) ρ^n **is of type** $\mathcal{H}_3$**.** Then

$$\rho^n = \langle \{N_Z \overset{h}{\rightsquigarrow} N_{\exists r.A_1},\ N_{A_1} \overset{h}{\rightsquigarrow} N_{\exists s.A_2},\ N_{\exists t.A_2} \rightsquigarrow N_Y, \\ (\{N_r, N_s\}, \{N_s, N_t\})\},\ N_Z \rightsquigarrow N_Y \rangle$$

$N_Z \overset{h}{\rightsquigarrow} N_{\exists r.A_1}$ can be derived using inferences $\rho^1, \rho^2, \ldots, \rho^{n-1}$, and thus $d_H(Z, \exists r.A_1) < k$. By assumption, we have

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{\exists r.A_1}$$

because $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Z$, $\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_{\exists r.A_1}$ and $d_H(Z, \exists r.A_1) < k$. Then we can derive $N_X \overset{h}{\rightsquigarrow} N_Y$ by:

- first deriving $N_X \overset{h}{\rightsquigarrow} N_{\exists r.A_1}, N_{A_1} \overset{h}{\rightsquigarrow} N_{\exists s.A_2}$;
- then applying the following inference

$$\rho^n = \langle \{N_X \overset{h}{\rightsquigarrow} N_{A_1}, \ldots, N_X \overset{h}{\rightsquigarrow} N_{A_m}, N_{A_0} \rightsquigarrow N_Y, \\ (\{N_{A_1}, N_{A_2}, \ldots, N_{A_m}\}, \{N_{A_0}\})\},\ N_X \rightsquigarrow N_Y \rangle$$

Therefore, we have $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.

The proposition is proved by induction. □

Theorem 51 *If $\mathcal{O}$ is an $\mathcal{EL}^+$-ontology, then*

$$\mathcal{O} \models X \sqsubseteq Y \text{ iff } \mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y,$$

where X, Y are concepts of either form A or $\exists r.B$.

Proof. "⇐" is obvious by induction over Table 3.2 and the fact that $N_X \rightsquigarrow N_Y$ implies $\mathcal{O} \models X \sqsubseteq Y$, so we only need to prove the direction "⇒".

Next, $\mathcal{O} \vdash X \sqsubseteq Y$ means $X \sqsubseteq Y$ can be derived from $\mathcal{O}$ using Table 2.1 (page 19). Recall that we have $\mathcal{O} \models X \sqsubseteq Y$ iff $\mathcal{O} \vdash X \sqsubseteq Y$ when X and Y are concept names.

Assume that $\mathcal{O} \models X \sqsubseteq Y$. We consider two cases:

Case 1. We assume $\mathcal{O} \vdash X \sqsubseteq Y$ Let $d(X, Y)$ be the length of one shortest derivation of $X \sqsubseteq Y$ from $\mathcal{O}$ using Table 2.1. We prove "⇒" by induction on $d(X, Y)$.

1. **Assume $d(X, Y) = 0$.** In this case $\mathcal{O}$ must contain axioms of the form

   $$X \sqsubseteq Y.$$

   We have $N_X \rightsquigarrow N_Y$ thus $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.

2. **Assuming "⇒" holds when $d(X, Y) < k$, we prove "⇒" also holds when $d(X, Y) = k$.**

   Suppose ρ^{last} is the last inference in one shortest derivation of $X \sqsubseteq Y$ using Table 2.1. Four cases arise:

   (a) ρ^{last} **is generated by** $\mathcal{R}_1$ $(n > 1)$. Then

   $$\rho^{last} = \langle \{X \sqsubseteq A_1, \ldots,\ X \sqsubseteq A_n,\ A_1 \sqcap \ldots \sqcap A_n \sqsubseteq Y\}, \\ X \sqsubseteq Y \rangle.$$

   We have $d(X, A_i) < k, i = 1, 2, \ldots, n$ because their corresponding subsumptions can be derived without ρ^{last}.

   By the assumption, we have

   $$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{A_i}, i = 1, 2, \ldots, n.$$

   Then we can derive $N_X \overset{h}{\rightsquigarrow} N_Y$ by first deriving

   $$N_X \overset{h}{\rightsquigarrow} N_{A_i}, i = 1, 2, \ldots, n,$$

then applying the H-inference:

$$\rho^n = \langle \{N_X \overset{h}{\rightsquigarrow} N_{A_1},\ \ldots,\ N_X \overset{h}{\rightsquigarrow} N_{A_m}, N_Y \rightsquigarrow N_Y, \\ (\{N_{A_1}, N_{A_1}, \ldots, N_{A_m}\}, \{N_Y\})\}, \\ N_X \rightsquigarrow N_Y \rangle$$

Then $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$.

(b) ρ^{last} **is generated by** $\mathcal{R}_1(n = 1)$ **or** $\mathcal{R}_2$. Then, in each case, we have ρ^{last} has the form

$$\langle \{X \sqsubseteq Z, Z \sqsubseteq Y\}, X \sqsubseteq Y \rangle.$$

Then, we have $d(X, Z), d(Z, Y) < k$. By the assumption, we have

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Z,$$
$$\mathcal{G}_\mathcal{O} \vdash_h N_Z \overset{h}{\rightsquigarrow} N_Y.$$

Then $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$ by Proposition 42.

(c) ρ^{last} **is generated by** $\mathcal{R}_3$. Then

$$\rho^{last} = \langle \{X \sqsubseteq \exists r.B_1, B_1 \sqsubseteq B_2, \exists r.B_2 \sqsubseteq Y\}, X \sqsubseteq Y \rangle.$$

We have

$$d(X, \exists r.B_1),\ d(B_1, B_2),\ d(\exists r.B_2, Y) < k$$

because their corresponding subsumptions can be derived without ρ^{last}. By the assumption, we have

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1},$$
$$\mathcal{G}_\mathcal{O} \vdash_h N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2},$$
$$\mathcal{G}_\mathcal{O} \vdash_h N_{\exists r.B_2} \overset{h}{\rightsquigarrow} N_Y.$$

Then we can derive $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_2}$ by:

i. first deriving $N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1}, N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2}$;
ii. then applying H-inference:

$$\rho^{new} = \langle \{N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_1}, N_{B_1} \overset{h}{\rightsquigarrow} N_{B_2}, N_{\exists r.B_2} \rightsquigarrow N_{\exists r.B_2}\}, \\ N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_2} \rangle.$$

Then $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$ by Proposition 42 because

$$\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_{\exists r.B_2},$$
$$\mathcal{G}_\mathcal{O} \vdash_h N_{\exists r.B_2} \overset{h}{\rightsquigarrow} N_B.$$

(d) ρ^{last} **is generated by** $\mathcal{R}_4$. Then we have ρ^{last} has the form

$$\langle \{X \sqsubseteq \exists r_1.A_1,\ \ldots,\ A_{n-1} \sqsubseteq Y,\ r_1 \circ \ldots \circ r_n \sqsubseteq r\}, X \sqsubseteq Y \rangle,$$

where $X = A_0, Y = \exists r_n.A_n$. Then, we have $d(A_0 \sqsubseteq \exists r_1.A_1), \ldots, d(A_{n-1} \sqsubseteq \exists r_n.A_n) < k$. By the assumption, we have

$$\mathcal{G}_\mathcal{O} \vdash_h N_{A_i} \overset{h}{\rightsquigarrow} N_{\exists r_i.A_{i+1}},\ \forall 1 \leq i < n.$$

Note that we have $n \in \{1, 2\}$ since $\mathcal{O}$ is normalized. We can derive $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$ as follows. First, we derive $N_{A_i} \overset{h}{\rightsquigarrow} N_{\exists r_i.A_{i+1}},\ \forall 1 \leq i < n$. Then, two cases arise:

i. if $n = 1$, we have $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$ by Proposition 42 and the fact that $\mathcal{G}_\mathcal{O} \vdash_h N_{\exists r_1.A_1} \rightsquigarrow N_{\exists r.A_1}$ since $r_1 \sqsubseteq r$.

ii. if $n = 2$, we derive $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$ by applying the H-inference:

$$\rho^{new} = \langle \{N_X \overset{h}{\rightsquigarrow} N_{\exists r_1.A_1},\ N_{A_1} \overset{h}{\rightsquigarrow} N_{\exists r_2.A_2},\ N_{\exists r_2.A_2} \rightsquigarrow N_Y, \\ (\{N_{r_1}, N_{r_2}\}, \{N_{r_2}, N_{r_1}\})\},\ N_X \rightsquigarrow N_Y \rangle,$$

where $Y = \exists r_2.A_2$.

Case 2. If $\mathcal{O} \vdash X \sqsubseteq Y$ does not hold, then X or Y is not concept name. In this case, we introduce new axioms

$$A \sqsubseteq X,$$
$$X \sqsubseteq A,$$
$$B \sqsubseteq Y,$$
$$Y \sqsubseteq B$$

with new concept names A, B and denote the extended ontology by $\mathcal{O}'$. Then, $\mathcal{O}' \models A \sqsubseteq B$ and thus $\mathcal{O}' \vdash A \sqsubseteq B$ since Table 2.1 is sound and complete. Therefore, we have $\mathcal{G}_{\mathcal{O}'} \vdash_h N_A \overset{h}{\rightsquigarrow} N_B$ by the same arguments as above. Now, notice that $\mathcal{G}_{\mathcal{O}'}$ is obtained from $\mathcal{G}_\mathcal{O}$ by adding 4 edges:

$$(\{N_A\}, \{N_X\}),$$
$$(\{N_X\}, \{N_A\}),$$
$$(\{N_B\}, \{N_Y\}),$$
$$(\{N_Y\}, \{N_B\}).$$

Thus, we have $\mathcal{G}_{\mathcal{O}'} \vdash_h N_A \overset{h}{\rightsquigarrow} N_B$ iff $\mathcal{G}_\mathcal{O} \vdash_h N_X \overset{h}{\rightsquigarrow} N_Y$. □

Theorem 52 *Let X, Y be of either form A or $\exists r.B$. Let*

$$\mathcal{S} = \{g^{-1}(H_{X,Y}) \mid H_{X,Y} \text{ is a minimal H-path from } N_X \text{ to } N_Y\}.$$

Then we have

$$\mathcal{J}_\mathcal{O}(X \sqsubseteq Y) = \{s \in \mathcal{S} \mid s' \not\subset s, \forall s' \in \mathcal{S}\}.$$

That is, all justifications for $X \sqsubseteq Y$ are the minimal subsets in $\mathcal{S}$.

Proof. For any justification $\mathcal{O}'$ of $X \sqsubseteq Y$, there exists a minimal H-path $H_{X,Y}$ such that $\mathcal{O}' = g^{-1}(H_{X,Y})$. The reason is that, since $\mathcal{O}' \models X \sqsubseteq Y$, there exists an H-path $H_{X,Y}$ from N_X to N_Y on $\mathcal{G}_{\mathcal{O}'}$ by Theorem 43. Without loss of generality, we can assume $H_{X,Y}$ is minimal on $\mathcal{G}_{\mathcal{O}'}$, then it is also minimal on $\mathcal{G}_{\mathcal{O}}$ since $\mathcal{G}_{\mathcal{O}'}$ is a sub-graph of $\mathcal{G}_{\mathcal{O}}$. We have

$$\mathcal{O}' = g^{-1}(H_{X,Y})$$

because otherwise there exists $\mathcal{O}'' \subsetneq \mathcal{O}'$ such that $\mathcal{O}'' = g^{-1}(H_{X,Y})$, and thus $\mathcal{O}'' \models X \sqsubseteq Y$ by Theorem 43 again. Therefore, $\mathcal{O}'$ is not a justification. Contradiction.

Now, we know $\mathcal{S}$ contains all justifications for $X \sqsubseteq Y$. Moreover, we have

$$g^{-1}(H_{X,Y}) \models X \sqsubseteq Y$$

for any H-path $H_{X,Y}$. Therefore, we have

$$\mathcal{J}_{\mathcal{O}}(X \sqsubseteq Y) = \{s \in \mathcal{S} \mid s' \not\subset s, \forall s' \in \mathcal{S}\}$$

by the definition of justifications.

□

4 - Computing Deductive Modules

As stated in the introduction, deductive modules provide users with a concise view of an ontology and has been used for various areas, like ontology debugging [2], re-use [32], and forgetting [49]. However, existing deductive module proposals are either inefficient from a computing point of view, or unsatisfactory from a quality point of view since the modules extracted are not concise enough. For example, *minimal modules* [16] guarantee the most concise results, but their computation suffer from high complexity and are time-consuming in practice. In contrast, $\top\bot^*$-*modules* [27] are easy to compute, but usually, they contain many redundant items.

To overcome the computation cost and the lack of quality, we propose to calculate two different kinds of deductive modules, called *pseudo-minimal modules* and *complete modules*, for $\mathcal{EL}$-ontologies. Complete modules are deductive modules that containing all minimal modules (recall Defintion 75 on page 73). In contrast, the definition of pseudo-minimal module relies on associating a *forest* (i.e., collection of trees) for a given ontology and a given vocabulary (i.e., signature). Pseudo-minimal modules are indeed minimal modules when there are finitely many entailments over a given vocabulary. Complete modules are less concise than pseudo-minimal modules, but easier to compute.

To compute these deductive modules, we propose a SAT-based method, called `ForMod`, based on Horn-clause encoding. For our experiments on real-world ontologies, the pseudo-minimal modules are indeed minimal in almost all cases (98.9%), and computing pseudo-minimal modules are more efficient (99.79 times faster on average) than `Zoom`, which is the state-of-the-art method for computing all minimal modules. This implies that our pseudo-minimal modules are of good quality and can be computed more efficiently. Note that our proposal applies to $\mathcal{EL}$-ontologies while `Zoom` only works for *acyclic $\mathcal{EL}$-terminologies*, which are specific $\mathcal{EL}$-ontologies (recall Definition 17 on page 18). Moreover, our complete modules are more compact than $\top\bot^*$-modules, although their computation time remains comparable.

This chapter is organized as follows. First, in Section 4.1, we discuss some works related to computing deductive modules. Then, in Section 4.2, we introduce our notion of *forest*, which is the core of definition of *pseudo-minimal modules* in Section 4.3. In Section 4.4, we present our method `ForMod`, which computes all pseudo-minimal modules and one complete module based on Horn-clause encoding for the construction of the forest. Then, in Section 4.5, we validate our method with real-world ontologies. Finally, the proofs of all theoretical results are provided in the last section of this chapter.

4.1 . Related work

We can distinguish two classes of deductive modules. The first class is the *minimal modules* [16, 48]. Minimal modules are of good quality as they do not contain any redundant terms. However, they suffer from high complexity (2-EXPTIME [28]) and are indeed time-consuming in practice [16, 48]. The second class is *syntactical locality-based modules* such as $\top\bot^*$-modules [69], *CEL*[71], *MEX*[41] and *AMEX* [24]. Their main idea is to compute a deductive module based on local syntactic structures efficiently (e.g., PTIME for $\top\bot^*$-module [69]). However, they usually contain many redundant axioms and thus are of bad quality. In this chapter, we are focusing on computing small deductive modules. Therefore, next, we only present details of related works focusing on computing minimal modules.

Recall that a deductive module for an ontology $\mathcal{O}$ and a signature Σ is a sub-ontology $\mathcal{M} \subseteq \mathcal{O}$ such that $\texttt{cDiff}_\Sigma(\mathcal{O}, \mathcal{M}) = \emptyset$. In the literature, there are mainly two different methods developed in order to compute minimal modules for a given signature Σ.

Minimal modules for $\mathcal{EL}$-ontologies The first method is the recursive algorithm Zoom[16], which computes all minimal modules for acyclic $\mathcal{EL}$-terminologies. The main idea is based on *the primitive witnesses theorem* stated as below.

Theorem 53 (primitive witnesses theorem [40]) *Given two $\mathcal{EL}$-terminologies $\mathcal{O}_1, \mathcal{O}_2$ and a signature Σ, if there exists an axiom $C \sqsubseteq D \in \texttt{cDiff}_\Sigma(\mathcal{O}_1, \mathcal{O}_2)$, then there exists an concept name $A \in sig(C \sqsubseteq D)$ such that one of the following properties hold:*

1. $C \sqsubseteq A \in \texttt{cDiff}_\Sigma(\mathcal{O}_1, \mathcal{O}_2)$;
2. $A \sqsubseteq D \in \texttt{cDiff}_\Sigma(\mathcal{O}_1, \mathcal{O}_2)$.

As a result of the primitive witnesses theorem, to compute deductive modules, one needs only to focus on preserving the derivability of axioms of the form $C \sqsubseteq A$ and $A \sqsubseteq D$. In other words, if $\mathcal{M} \subseteq \mathcal{O}$ is a minimal sub-ontology such that

1. $\mathcal{M} \models C \sqsubseteq A$ if $\mathcal{O} \models C \sqsubseteq A$,
2. $\mathcal{M} \models A \sqsubseteq D$ if $\mathcal{O} \models A \sqsubseteq D$,

for any concept name $A \in \Sigma$, $\mathcal{EL}$-concepts C, D such that $sig(C) \cup sig(D) \subseteq \Sigma$, then $\texttt{cDiff}_\Sigma(\mathcal{O}, \mathcal{M})$ is an empty set by the primitive witnesses theorem. Therefore, such a sub-ontology $\mathcal{M}$ is a minimal module for $\mathcal{O}$ and Σ.

To capture all axioms $C \sqsubseteq A$, $A \sqsubseteq D$ that are inferred by $\mathcal{O}$, Zoom introduces two different kinds of modules, called *subsumee justification* and *subsumer justification*. Each subsumee (or subsumer) justification is associated with a concept $A \in \mathsf{N}_C$ and a signature $\Sigma \subseteq \mathsf{N}_C \cup \mathsf{N}_R$ as follows:

- a $\langle A, \Sigma \rangle$-subsumee justification for $\mathcal{O}$ is a minimal sub-ontology $\mathcal{M} \subseteq \mathcal{O}$ such that for any axiom $\alpha := C \sqsubseteq A$, $sig(C) \subseteq \Sigma$, we have $\mathcal{M} \models \alpha$ if $\mathcal{O} \models \alpha$.

- a $\langle A, \Sigma\rangle$-subsumer justification for $\mathcal{O}$ is a minimal sub-ontology $\mathcal{M} \subseteq \mathcal{O}$ such that for any axiom $\alpha := A \sqsubseteq D,\ \mathit{sig}(D) \subseteq \Sigma$, we have $\mathcal{M} \models \alpha$ if $\mathcal{O} \models \alpha$.

To compute all subsumee and subsumer justifications, Zoom adapts a *logical difference algorithm* proposed in [53] that computes logical difference $\mathtt{cDiff}_\Sigma(\mathcal{O}_1, \mathcal{O}_2)$ (recall Section 2.5 on page 21) between two $\mathcal{EL}$-terminologies $\mathcal{O}_1, \mathcal{O}_2$. Then, Zoom computes all minimal modules by combining all $\langle A, \Sigma\rangle$-subsumee and $\langle A, \Sigma\rangle$-subsumer justifications for all $A \in \Sigma$. In details, let $\mathcal{J}^{\leftarrow}_{\mathcal{O}}(A, \Sigma)$, $\mathcal{J}^{\rightarrow}_{\mathcal{O}}(A, \Sigma)$ be the collection of all $\langle A, \Sigma\rangle$-subsumee, $\langle A, \Sigma\rangle$-subsumer justifications for $\mathcal{O}$ and Σ respectively. Then all minimal modules for the ontology $\mathcal{O}$ and the signature Σ are minimal sub-ontologies in the following collection:

$$\bigotimes_{A\in\Sigma}\left(\mathcal{J}^{\leftarrow}_{\mathcal{O}}(A, \Sigma) \bigotimes \mathcal{J}^{\rightarrow}_{\mathcal{O}}(A, \Sigma)\right),$$

where $\bigotimes$ is the product of two collections of sets defined by:

$$\mathcal{S} \bigotimes \mathcal{S}' = \{S \cup S' \mid S \in \mathcal{S},\ S' \in \mathcal{S}'\}.$$

Example 54 *Let us consider the ontology $\mathcal{O}$ below:*

$$\begin{aligned}\mathcal{O} = \{\ \ & \alpha_1 : Y \sqcap Z \sqsubseteq A,\\ & \alpha_2 : A \sqsubseteq Y,\\ & \alpha_3 : Y \sqsubseteq B,\\ & \alpha_4 : Z_1 \sqcap Z_2 \sqsubseteq Z,\\ & \alpha_5 : A_1 \sqsubseteq Y,\\ & \alpha_6 : A_2 \sqsubseteq Z,\\ & \alpha_7 : A_2 \sqsubseteq Z_1,\\ & \alpha_8 : A_2 \sqsubseteq Z_2\ \ \},\end{aligned}$$

and the signature $\Sigma = \{A_1, A_2, B\}$. Note that $\mathcal{O} \models A_1 \sqcap A_2 \sqsubseteq A$ is a conclusion of the form $\mathcal{O} \models C \sqsubseteq A$ with $C = A_1 \sqcap A_2$ and $\mathit{sig}(C) \subseteq \Sigma$.

In this example, a sub-ontology $\mathcal{M}$ is a $\langle A, \Sigma\rangle$-subsumee justification iff it is a minimal sub-ontology such that $\mathcal{M} \models A_1 \sqcap A_2 \sqsubseteq A$. There are two such $\langle A, \Sigma\rangle$-subsumee justifications:

$$\begin{aligned}\mathcal{M}_1 &= \{\alpha_1,\ \alpha_4,\ \alpha_5,\ \alpha_7,\ \alpha_8\},\\ \mathcal{M}_2 &= \{\alpha_1,\ \alpha_5,\ \alpha_6\}.\end{aligned}$$

Similarly, it holds that $\mathcal{O} \models A \sqsubseteq B$, which is the unique conclusion that must be preserved by a $\langle A, \Sigma\rangle$-subsumer justification. There is only one $\langle A, \Sigma\rangle$-subsumer justification: $\mathcal{M}_3 = \{\alpha_2,\ \alpha_3\}$.

This leads to consider the following collection for extracting minimal modules

$$\bigotimes_{X\in\Sigma}\left(\mathcal{J}^{\leftarrow}_{\mathcal{O}}(X,\Sigma)\bigotimes\mathcal{J}^{\rightarrow}_{\mathcal{O}}(X,\Sigma)\right)=\mathcal{J}^{\leftarrow}_{\mathcal{O}}(A,\Sigma)\bigotimes\mathcal{J}^{\rightarrow}_{\mathcal{O}}(A,\Sigma)$$
$$=\{\mathcal{M}_1\cup\mathcal{M}_3,\ \mathcal{M}_2\cup\mathcal{M}_3\}.$$

In the above collection, $\mathcal{M}_1\cup\mathcal{M}_3$, $\mathcal{M}_2\cup\mathcal{M}_3$ are minimal sub-ontologies. Therefore, the two minimal modules for $\mathcal{O}$ and Σ are $\mathcal{M}_1\cup\mathcal{M}_3$, $\mathcal{M}_2\cup\mathcal{M}_3$.

One of the main constraints of `Zoom` is that it does not work for $\mathcal{EL}$-ontologies because the logical difference algorithm does not apply to $\mathcal{EL}$-ontologies. In contrast, the witnesses theorem has been extended to $\mathcal{EL}$-ontologies [20].

Minimal modules of $\mathcal{ALC}$-ontologies The second method proposed in [48] computes minimal deductive modules with respect to $\mathcal{ALCH}$-ontologies. $\mathcal{ALCH}$-ontologies are more expressive than $\mathcal{EL}$-ontologies because they allow

1. disjunction (i.e., $A\sqcup B$);
2. negation (i.e., $\neg A$);
3. value restriction (e.g., $\forall r.C$).
4. role hierarchy (e.g., $r\sqsubseteq s$)

The main idea developed in [48] is represented by Algorithm 5. This algorithm is based on the same idea as the algorithm 1 presented in Section 3.1.1 which computes a single justification for a given conclusion. For a given ontology $\mathcal{O}$ and a signature Σ, Algorithm 5 outputs one minimal module for $\mathcal{O}$ and Σ by eliminating redundant axioms one by one. Redundancy is determined by reasoners using a uniform interpolant for each sub-ontology (i.e., $U(\mathcal{M}\setminus\{\beta\},\Sigma)\models U(\mathcal{O},\Sigma)$ in Line 5).

However, it is known that computing uniform interpolant could be extremely time-consuming [28, 59]. Therefore, computing uniform interpolant for each sub-ontology $\mathcal{M}\setminus\{\beta\}$ (Algorithm 5, Line 4) is not reasonable. To overcome this difficulty, the authors of [48] introduced *annotation form*. Briefly, for each given $\mathcal{ALCH}$-ontology $\mathcal{O}$ and signature Σ, the *annotation form* $\mathcal{O}_a$ for $\mathcal{O}$ is constructed by integrating a fresh concept name $\mathtt{A}_\alpha$ for each axiom $\alpha\in\mathcal{O}$:

$$\mathcal{O}_a:=\{\mathtt{A}_{C\sqsubseteq D}\sqcap C\sqsubseteq D\mid C\sqsubseteq D\in\mathcal{O}\}.$$

Let $\Sigma_a:=\Sigma\cup\{\mathtt{A}_\alpha\mid\alpha\in\mathcal{O}\}$ be an extended signature of Σ, and let $U(\mathcal{O}_a,\Sigma_a)$ be a uniform interpolant for the annotation form $\mathcal{O}_a$ and Σ_a. The authors of [48] show that a uniform interpolant for a sub-ontology $\mathcal{M}\subseteq\mathcal{O}$ and a signature Σ could be directly generated by applying a substitution procedure over $U(\mathcal{O}_a,\Sigma_a)$. The substitution takes two steps:

Algorithm 5 : Computing one minimal module

input : a signature Σ, an ontology $\mathcal{O}$
output : a minimal module $\mathcal{M}$ for $\mathcal{O}$ and Σ

$U(\mathcal{O}, \Sigma) \leftarrow$ *a uniform interpolant for* $\mathcal{O}$ *and* Σ;
$\mathcal{M} \leftarrow \mathcal{O}$;
for *each axiom* $\beta \in \mathcal{O}$ **do**
  $U(\mathcal{M} \setminus \{\beta\}, \Sigma) \leftarrow$ *a uniform interpolant for* $\mathcal{M} \setminus \{\beta\}$ *and* Σ;
  if $U(\mathcal{M} \setminus \{\beta\}, \Sigma) \models U(\mathcal{O}, \Sigma)$ **then**
    $\mathcal{M} \leftarrow \mathcal{M} \setminus \{\beta\}$
  end
end
return $\mathcal{M}$

1. Replacing each label $\mathtt{A}_\alpha$ in $U(\mathcal{O}_a, \Sigma_a)$, $\alpha \notin \mathcal{M}$, to $\bot$;
2. Replacing each label $\mathtt{A}_\alpha$ in $U(\mathcal{O}_a, \Sigma_a)$, $\alpha \in \mathcal{M}$, to $\top$.

As a result, one only needs to compute one uniform interpolant $U(\mathcal{O}_a, \Sigma_a)$. This is done by adapting LETHE [47], which is a tool that computes uniform interpolants over $\mathcal{ALCH}$ or more expressive ontologies.

Example 55 *Consider the signature* $\Sigma = \{A, r\}$ *and the ontology*

$$\mathcal{O} = \{\ \alpha_1 : \exists r.\top \sqsubseteq A \sqcup B,\\ \alpha_2 : \exists r.A \sqsubseteq B,\\ \alpha_3 : \exists r.B \sqsubseteq A\ \}.$$

Then the annotated ontology of $\mathcal{O}$ *is*

$$\mathcal{O}_a = \{\ \mathtt{A}_{\alpha_1} \sqcap \exists r.\top \sqsubseteq A \sqcup B,\\ \mathtt{A}_{\alpha_2} \sqcap \exists r.A \sqsubseteq B,\\ \mathtt{A}_{\alpha_3} \sqcap \exists r.B \sqsubseteq A\ \},$$

and the extended signature is $\Sigma_a = \{A, r, \mathtt{A}_{\alpha_1}, \mathtt{A}_{\alpha_2}, \mathtt{A}_{\alpha_3}\}$. *One possible uniform interpolant for* $\mathcal{O}_a$ *and* Σ_a *is*

$$U(\mathcal{O}_a, \Sigma_a) = \{\ \mathtt{A}_{\alpha_3} \sqcap \exists r.(\mathtt{A}_{\alpha_2} \sqcap \exists r.A) \sqsubseteq A,\\ \mathtt{A}_{\alpha_3} \sqcap \exists r.(\mathtt{A}_{\alpha_1} \sqcap \exists r.\top \sqcap \neg A) \sqsubseteq A\ \}.$$

Then, by replacing all labels $\mathtt{A}_{\alpha_i}, i = 1, 2, 3$ *to* $\top$, *one obtain a uniform interpolant for* $\mathcal{O}$ *and* Σ:

$$U(\mathcal{O}, \Sigma) = \{\ \exists r.(\top \sqcap \exists r.A) \sqsubseteq A,\\ \exists r.(\top \sqcap \exists r.\top \sqcap \neg A) \sqsubseteq A\ \},$$

Finally, since $U(\mathcal{M}, \Sigma) \not\models U(\mathcal{O}, \Sigma)$ *for any sub-ontology* $\mathcal{M} \subset \mathcal{O}$, *the unique minimal module for* $\mathcal{O}$ *and* Σ *(obtained by Algorithm 5) is* $\mathcal{O}$ *itself.*

It is important to recall here that a uniform interpolant for a given ontology may not exist. Therefore, the method above does not always work. The authors of [48] have also proposed another deductive module obtained by collecting all axioms with labels appearing in $U(\mathcal{O}_a, \Sigma_a)$. This deductive module is no longer minimal but much less expensive to compute than minimal ones.

Note that even for the same ontology and signature, their corresponding minimal modules could be different if different languages are used as the underlying semantics. Therefore, we can not compare different methods of computing deductive modules under different languages directly (e.g., `Zoom` for $\mathcal{EL}$ and [48] for $\mathcal{ALC}$). Here is a simple example:

Example 56 *Consider the ontology*

$$\begin{aligned}\mathcal{O} = \{\ \ & \alpha_1 : \exists r.A \sqsubseteq A_1,\\ & \alpha_2 : \exists r.A_2 \sqsubseteq A_1,\\ & \alpha_3 : \exists r.B \sqsubseteq A_1,\\ & \alpha_4 : A \sqsubseteq B,\\ & \alpha_5 : A_2 \sqsubseteq B\ \ \},\end{aligned}$$

and the signature $\Sigma = \{r, A, A_1, A_2\}$. *Then* $\mathcal{O}$ *is an* $\mathcal{EL}$*-ontology and there are two different minimal modules for* $\mathcal{O}$ *and* Σ *under* $\mathcal{EL}$*-semantics:*

$$\begin{aligned}\mathcal{M}_1 &= \{\ \alpha_1,\ \alpha_2\ \}\\ \mathcal{M}_2 &= \{\ \alpha_3,\ \alpha_4,\ \alpha_5\ \}\end{aligned}$$

Now, if we regard $\mathcal{O}$ *as an* $\mathcal{ALC}$*-ontology, then there is only one minimal module* $\mathcal{M}_2$ *for* $\mathcal{O}$ *and* Σ *under the* $\mathcal{ALC}$*-semantics. The reason is that we have*

$$\begin{aligned}\mathcal{M}_1 &\not\models \exists r.(A \sqcup A_2) \sqsubseteq A_1\\ \mathcal{M}_2 &\models \exists r.(A \sqcup A_2) \sqsubseteq A_1.\end{aligned}$$

Note that $\exists r.(A \sqcup A_2) \sqsubseteq A_1$ *is an* $\mathcal{ALC}$*-axiom but it is not an* $\mathcal{EL}$*-axiom.*

4.2 . Introducing Forest $F_\Sigma^\mathcal{O}$

Now, we introduce our method for computing deductive modules. First, we analyze possible ways to compute deductive models and then introduce our notion of forest $F_\Sigma^\mathcal{O}$.

4.2.1 . Motivation

Assuming $\mathcal{U}_\Sigma^\mathcal{O}$ is a uniform interpolant for an ontology $\mathcal{O}$ and a signature Σ, we can compute deductive modules using the justifications of each axiom $\beta \in \mathcal{U}_\Sigma^\mathcal{O}$ (see Figure 4.1). More precisely, consider the following collection of sub-ontologies defined as the union of justifications:

$$\mathcal{S} = \{\ \bigcup_{\alpha \in \mathcal{U}_\Sigma^\mathcal{O}} J_\alpha \mid J_\alpha \text{ is a justification of } \alpha\}.$$

Each element $\mathcal{M}$ in $\mathcal{S}$ is a sub-ontology of $\mathcal{O}$ and contains exactly one justification for each element in $\mathcal{U}_\Sigma^\mathcal{O}$. Thus it is a deductive module for $\mathcal{O}$ and Σ. We can conclude that the minimal modules for $\mathcal{O}$ and Σ are identical to the subset-minimal sub-ontologies in $\mathcal{S}$. For example:

Example 57 *Consider the ontology*

$$\begin{aligned}\mathcal{O} = \{\ &\alpha_1 : A \sqsubseteq \exists r.B_1,\\ &\alpha_2 : B_1 \sqsubseteq A_1 \sqcap A_2,\\ &\alpha_3 : A \sqsubseteq \exists r.A_1\\ &\alpha_4 : \exists r.B_2 \sqsubseteq A_2,\\ &\alpha_5 : A_3 \sqcap A_4 \sqsubseteq B_2\ \}.\end{aligned}$$

and the signature $\Sigma = \{A,\ A_1,\ A_2,\ A_3,\ A_4,\ r\}$. *Then,*

$$\begin{aligned}\mathcal{U}_\Sigma^\mathcal{O} = \{&\beta_1 : A \sqsubseteq \exists r.(A_1 \sqcap A_2),\\ &\beta_2 : A \sqsubseteq \exists r.A_1,\\ &\beta_3 : \exists r.(A_3 \sqcap A_4) \sqsubseteq A_2\}\end{aligned}$$

is a uniform Interpolant for $\mathcal{O}$ *and* Σ. *We know that there are*

1. *one justification for* β_1*:*
$$J_{\beta_1} = \{\alpha_1, \alpha_2\};$$

2. *two justifications for* β_2*:*
$$\begin{aligned}J^1_{\beta_2} &= \{\alpha_1,\ \alpha_2\},\\ J^2_{\beta_2} &= \{\alpha_3\};\end{aligned}$$

3. *one justification for* β_3*:*
$$J_{\beta_3} = \{\alpha_4, \alpha_5\}.$$

Therefore, the collection $\mathcal{S}$ *is* $\{\mathcal{O},\ \mathcal{O}\backslash\{\alpha_3\}\}$. $\mathcal{S}$ *contains two different deductive modules for* $\mathcal{O}$ *and* Σ. *Moreover,* $\mathcal{O}\backslash\{\alpha_3\}$ *is the unique minimal module for* $\mathcal{O}$ *and* Σ.

In Figure 4.1, computing deductive modules through uniform interpolant and justifications is shown by the black bold arrows. As already discussed, two main difficulties arise when implementing such a simple idea:

1. Computing uniform interpolants is hard;

2. A uniform interpolant does not always exist.

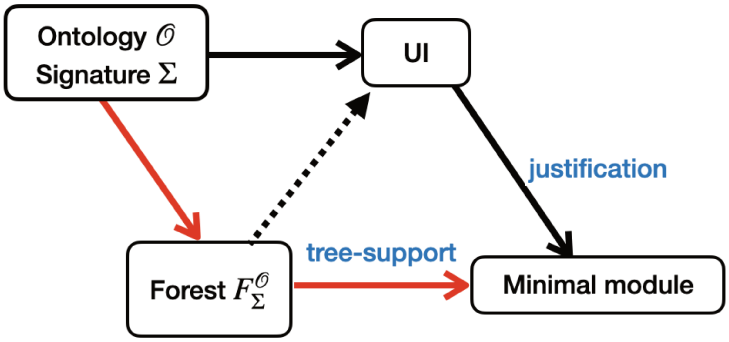


Figure 4.1: The main schema

Our contribution to overcome these difficulties is to proceed to the computation of deductive modules through a new object $F_\Sigma^{\mathcal{O}}$, called a *forest*, associated with the ontology $\mathcal{O}$ and the signature Σ. This alternative computation process is depicted by the red arrows in Figure 4.1.

The notion of forest $F_\Sigma^{\mathcal{O}}$, formally introduced hereafter, is inspired by the *regular tree grammar* developed in [59]. The forest $F_\Sigma^{\mathcal{O}}$ and the regular tree grammar can be regarded respectively as a set of *derivation trees* and *derivation rules* that generate candidate axioms of a uniform interpolant. Our new method for computing deductive modules relies on the structure of the trees in $F_\Sigma^{\mathcal{O}}$ instead of using justifications, and has mainly three benefits:

1. The computation of $F_\Sigma^{\mathcal{O}}$ is efficient;
2. Our method works no matter whether a uniform interpolant exists or not;
3. Our method can be easily encoded as a SAT-problem and thus solved by efficient SAT-tools.

Next, we develop the presentation of forests $F_\Sigma^{\mathcal{O}}$ and show the relation between $F_\Sigma^{\mathcal{O}}$ and uniform interpolant (i.e., the dotted arrow in Figure 4.1). In the following sections, we introduce *pseudo-minimal modules* using $F_\Sigma^{\mathcal{O}}$ and describe `ForMod`, our SAT-based algorithm `ForMod` to compute them.

4.2.2 . Definition of forest $F_\Sigma^{\mathcal{O}}$

Next, we consider *labeled trees*. Each labeled tree $\mathtt{t}$ is associated with a *label map* "`lab`" that maps

1. node $\mathtt{n}$ in $\mathtt{t}$ to an concept name;
2. edge e in $\mathtt{t}$ to a condition of the form $\alpha \in \mathcal{O}$ or $\mathcal{O} \models A \sqsubseteq B$.

For simplicity, we use $\mathtt{root(t)}$ to denote the root of $\mathtt{t}$, and $\mathtt{A_t}$ the label of the root (i.e., $\mathtt{A_t} = \mathtt{lab(root(t))}$).

The forest $F_\Sigma^{\mathcal{O}}$ consists of two kinds of trees, forward trees, and backward trees, defined as follows:

Definition 58 *A labeled tree* $\mathtt{t}^+$ *is a* ***forward tree*** *(hereafter f-tree) from* A *to* Σ *over* $\mathcal{O}$ *iff:*

1. *the label of the root of* $\mathtt{t}^+$ *is* A *(i.e.,* $\mathtt{A}_{\mathtt{t}^+} = A$*);*
2. *for node* $\mathtt{n} \in \mathtt{t}^+$ *distinct from the root,* $\mathtt{lab}(\mathtt{n}) \in \Sigma$ *iff* $\mathtt{n}$ *is a leaf of* $\mathtt{t}^+$*,*
3. *if the child set of a node* $\mathtt{n}_0 \in \mathtt{t}^+$ *is* $\{\mathtt{n}_1, \ldots, \mathtt{n}_m\}$ *and*

$$B_i = \mathtt{lab}(\mathtt{n}_i), 0 \leq i \leq m,$$

then for each $0 \leq i \leq m$*, one of the following conditions holds:*

(a) $B_0 \sqsubseteq \exists r.B_i \in \mathcal{O}$ *and* $r \in \Sigma$*,*

(b) $\mathcal{O} \models B_0 \sqsubseteq B_i$*, where* B_i *is not primitive or* $B_i \in \Sigma$*.*

The edge e *from* $\mathtt{n}_0$ *to* $\mathtt{n}_i$ *is labeled by the condition that generates* e*. That is, the* $\mathtt{lab}(e)$ *is* $B_0 \sqsubseteq \exists r.B_i \in \mathcal{O}$ *for the case 3(a);* $\mathtt{lab}(e)$ *is* $\mathcal{O} \models B_0 \sqsubseteq B_i$ *for the case 3(b).*

For the sake of the presentation, edges labeled by $B_0 \sqsubseteq \exists r.B_i \in \mathcal{O}$ (i.e., case 3(a)) are called $\exists r$-edges.

Example 59 (Example 57 cont'd) *For simplicity and without ambiguity, in the figures, we represent a node* $\mathtt{n}$ *by its label* $\mathtt{lab}(\mathtt{n})$ *and mark* $\exists r$*-edges by a red "*$\exists r$*". In Figure 4.2,* $\mathtt{t}_1^+$*,* $\mathtt{t}_2^+$ *are two f-trees from* A *to* Σ *over* $\mathcal{O}$*, and* $\mathtt{t}_2^+$ *is a sub-tree of* $\mathtt{t}_1^+$*. There are still 5 other f-trees from* A *to* Σ *over* $\mathcal{O}$ *that are proper sub-trees of* $\mathtt{t}_1^+$*.*

For $\Sigma' = \Sigma \cup \{B_1\}$*, the trees* $\mathtt{t}_1^+, \mathtt{t}_2^+$ *are no longer f-trees from* A *to* Σ' *over* $\mathcal{O}$*, because* $\mathtt{t}_1^+, \mathtt{t}_2^+$ *both contain an internal node labeled by* $B_1 \in \Sigma'$ *violating the requirement 2 in Definition 58.*

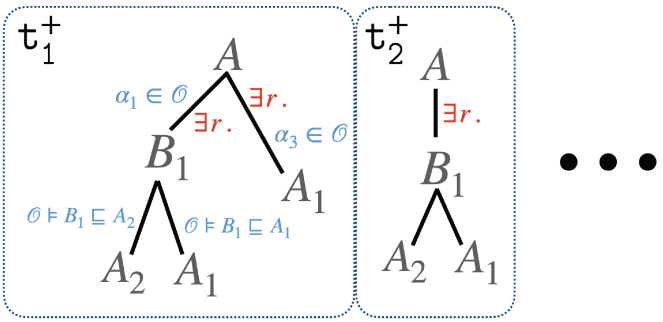


Figure 4.2: f-trees (blue words are the label of edges).

Definition 60 *A labeled tree* $\mathtt{t}^-$ *is a* ***backward tree*** *(hereafter b-tree) from* A *to* Σ *over* $\mathcal{O}$ *iff:*

1. *the label of the root of* $\mathtt{t}^-$ *is* A *(i.e.,* $\mathtt{A}_{\mathtt{t}^-} = A$*);*
2. *for node* $\mathtt{n} \in \mathtt{t}^-$ *distinct from the root,* $\mathtt{lab}(\mathtt{n}) \in \Sigma$ *iff* $\mathtt{n}$ *is a leaf of* $\mathtt{t}^-$*;*
3. *if the child set of a node* $\mathtt{n}_0 \in \mathtt{t}^-$ *is* $\{\mathtt{n}_1, \ldots, \mathtt{n}_m\}$ *and*

$$B_i = \mathtt{lab}(\mathtt{n}_i), 0 \leq i \leq m,$$

then one of the following conditions holds:

   (a) $m = 1$, $\exists r.B_1 \sqsubseteq B_0 \in \mathcal{O}, r \in \Sigma$*;*

   (b) $m = 1$, $\mathcal{O} \models B_1 \sqsubseteq B_0$*, where* B_1 *is not primitive;*

   (c) $m > 1$, $B_1 \sqcap \ldots \sqcap B_m \sqsubseteq B_0 \in \mathcal{O}$*.*

Again, here, the edge from $\mathtt{n}_0$ *to* $\mathtt{n}_i$ *is labeled by the condition generating* e*.*

Similarly, edges labeled with $\exists r.B_1 \sqsubseteq B_0 \in \mathcal{O}$ are called $\exists r$-edges. Note that for b-trees, all edges from a node to its children are generated by the same condition.

Example 61 (Example 57 cont'd) *In Figure 4.3,* $\mathtt{t}_1^-$ *is the only b-tree from* A_2 *to* Σ *over* $\mathcal{O}$*. Note that the tree on the right at Figure 4.3 is not a b-tree from* A_2 *to* Σ *over* $\mathcal{O}$ *because we do not have* $A_3 \sqsubseteq B_2 \in \mathcal{O}$ *nor* $\mathcal{O} \models A_3 \sqsubseteq B_2$ *which violates the requirement 3 at Definition 60.*

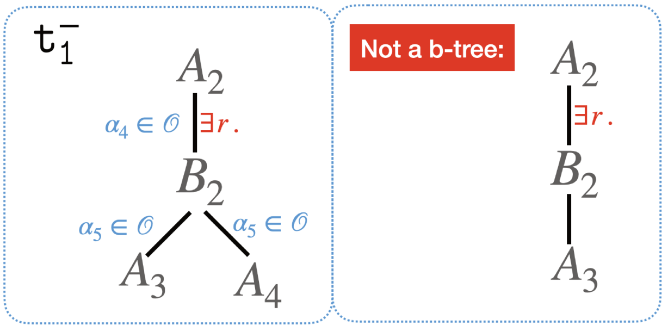


Figure 4.3: b-tree (an example and a counter-example)

The forest $F_\Sigma^\mathcal{O}$ associated with an ontology $\mathcal{O}$ and a signature Σ is defined as follows:

Definition 62 *The forest* $F_\Sigma^\mathcal{O}$ *is the collection of all f-trees and b-trees from* A *to* Σ *over* $\mathcal{O}$*, where* $A \in sig(\mathcal{O})$*.*

Notice that $F^{\mathcal{O}}_{\Sigma}$ could be an infinite set of trees. For example:

Example 63 *Let* $\mathcal{O}_1=\{A \sqsubseteq \exists r.B,\ B \sqsubseteq \exists r.B,\ B \sqsubseteq \exists r.A\}$ *and* $\Sigma_1=\{A, r\}$. *Then* $F^{\mathcal{O}_1}_{\Sigma_1}$ *contains infinitely many f-trees* $\mathtt{t}^+_1, \mathtt{t}^+_2, \ldots$ *from* A *to* Σ_1 *over* $\mathcal{O}_1$ *as shown in Figure 4.4.*

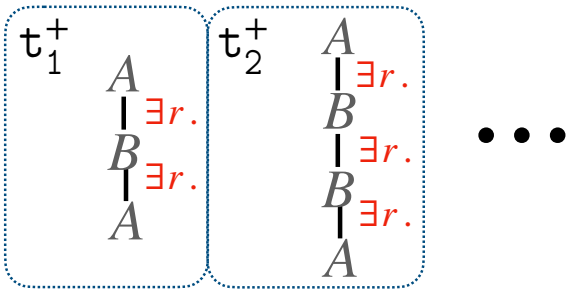


Figure 4.4: Some trees of $F^{\mathcal{O}_1}_{\Sigma_1}$

The f-trees in Figure 4.4 are generated by the loop starting at the node labeled by A. More precisely, we say that a tree $\mathtt{t} \in F^{\mathcal{O}}_{\Sigma}$ contains a **loop** if there is a path p from root $\mathtt{root}(\mathtt{t})$ to a leaf of $\mathtt{t}$ with two different nodes having the same label. For the same reason, there could be infinitely many b-trees in a forest $F^{\mathcal{O}}_{\Sigma}$. We have that $F^{\mathcal{O}}_{\Sigma}$ is infinite iff some $\mathtt{t} \in F^{\mathcal{O}}_{\Sigma}$ contains a loop.

4.2.3 . Generating uniform interpolant from forest $F^{\mathcal{O}}_{\Sigma}$

Although our computation of deductive modules does not require computing UI, we investigate here how to generate a uniform interpolant from a forest. These discussions (definitions and results) are necessary for presenting our method.

First, we associate each f-tree or b-tree $\mathtt{t}$ with an $\mathcal{EL}$-concept $\mathtt{C_t}$ defined as follows:

Definition 64 *For a f-tree or b-tree* $\mathtt{t}$, *the corresponding* $\mathcal{EL}$*-concept* $\mathtt{C_t}$ *is defined inductively by:*

1. *if the child set of the root* $\mathtt{root}(\mathtt{t})$ *is empty, then* $\mathtt{C_t} = \mathtt{A_t}$;
2. *if the child set of the root* $\mathtt{root}(\mathtt{t})$ *is* $\{\mathtt{n}_1, \ldots, \mathtt{n}_m\}$, *and there exists* $0 \leq k \leq m$ *such that the edge from* $\mathtt{root}(\mathtt{t})$ *to* $\mathtt{n}_i$ *is a* $\exists r_i$*-edge for* $i \leq k$. *Then:*

$$\mathtt{C_t} = \exists r_1.\mathtt{C_{t_1}} \sqcap \ldots \sqcap \exists r_k.\mathtt{C_{t_k}} \sqcap \mathtt{C_{t_{k+1}}} \sqcap \ldots \sqcap \mathtt{C_{t_m}},$$

where $\mathtt{t}_i$ *is the maximal sub-tree of* $\mathtt{t}$ *rooted at* $\mathtt{n}_i$.

For instance, for $\mathtt{t}^+_1$ of Example 59, we have

$$\mathtt{C_{t^+_1}} = \exists r.(A_2 \sqcap A_1) \sqcap \exists r.A_1;$$

For $\mathtt{t}_1^-$ in Example 61, we have

$$\mathtt{C}_{\mathtt{t}_1^-}{=}\exists r.(A_3 \sqcap A_4).$$

Now, the uniform interpolant candidate generated from $F_\Sigma^{\mathcal{O}}$ is defined as follows.

Definition 65 *Given a forest F of b-trees and f-trees, let us associate with F a set of axioms, denoted $\mathcal{U}_\Sigma(F)$, and defined as the union of the three axiom sets below:*

$$\mathcal{U}_\Sigma^1(F){=}\{\mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{A}_{\mathtt{t}^-} \mid \mathtt{t}^- \in F, \quad \mathtt{A}_{\mathtt{t}^-} \in \Sigma\}$$

$$\mathcal{U}_\Sigma^2(F){=}\{\mathtt{A}_{\mathtt{t}^+} \sqsubseteq \mathtt{C}_{\mathtt{t}^+} \mid \mathtt{t}^+ \in F, \quad \mathtt{A}_{\mathtt{t}^+} \in \Sigma\}$$

$$\mathcal{U}_\Sigma^3(F){=}\{\mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{C}_{\mathtt{t}^+} \mid \mathtt{t}^-, \mathtt{t}^+ \in F, \quad \mathtt{A}_{\mathtt{t}^-}{=}\mathtt{A}_{\mathtt{t}^+} \notin \Sigma\}.$$

Next, when there is no ambiguity, we omit Σ in $\mathcal{U}_\Sigma(F)$ and $\mathcal{U}_\Sigma^i(F), i \in \{1,2,3\}$. We say $\mathcal{U}_\Sigma(F)$ is a uniform interpolant candidate because of the following result:

Theorem 66 *For any axiom α with sig$(\alpha) \subseteq \Sigma$, we have*

$$\mathcal{O} \models \alpha \text{ iff } \mathcal{U}(F_\Sigma^{\mathcal{O}}) \models \alpha.$$

Therefore, $\mathcal{U}(F_\Sigma^{\mathcal{O}})$ is a uniform interpolant for $\mathcal{O}$ and Σ if $F_\Sigma^{\mathcal{O}}$ is finite.

Proof. The proof is provided on page 85. □

Example 67 (Example 59 cont'd) *Note that $F_\Sigma^{\mathcal{O}}$ is finite because no b-tree or f-tree over the ontology $\mathcal{O}$ contains a loop. We have*

$$\begin{aligned}
\mathcal{U}^1(F_\Sigma^{\mathcal{O}}) &= \{\beta_1' : \exists r.(A_3 \sqcap A_4) \sqsubseteq A_2\},\\
\mathcal{U}^2(F_\Sigma^{\mathcal{O}}) &= \{\beta_2' : A \sqsubseteq \exists r.(A_2 \sqcap A_1) \sqcap \exists r.A_1,\\
&\qquad\ \beta_3' : A \sqsubseteq \exists r.(A_2 \sqcap A_1),\\
&\qquad\ \beta_4' : A \sqsubseteq \exists r.A_2 \sqcap \exists r.A_1,\\
&\qquad\ \beta_5' : A \sqsubseteq \exists r.A_1\\
&\qquad\ \beta_6' : A \sqsubseteq \exists r.A_2\},\\
\mathcal{U}^3(F_\Sigma^{\mathcal{O}}) &= \emptyset
\end{aligned}$$

Here,

1. *β_1' is generated from $\mathtt{t}_1^- \in F_\Sigma^{\mathcal{O}}$;*
2. *β_2' are generated from $\mathtt{t}_1^+ \in F_\Sigma^{\mathcal{O}}$;*
3. *β_3' are generated from $\mathtt{t}_2^+ \in F_\Sigma^{\mathcal{O}}$;*
4. *β_4', β_5', β_6' are generated from sub-trees of $\mathtt{t}_1^+$ other than $\mathtt{t}_2^+$.*

Then we have

$$\mathcal{U}(F_\Sigma^\mathcal{O}) = \mathcal{U}^1(F_\Sigma^\mathcal{O}) \cup \mathcal{U}^2(F_\Sigma^\mathcal{O}) \cup \mathcal{U}^3(F_\Sigma^\mathcal{O})$$

is a uniform interpolant for $\mathcal{O}$ and Σ since $F_\Sigma^\mathcal{O}$ is finite.

It may happen that $F_\Sigma^\mathcal{O}$ is infinite and nevertheless a uniform interpolant for $\mathcal{O}$ and Σ exists. In that case, we can still obtain a uniform interpolant from $F_\Sigma^\mathcal{O}$ by extracting a finite subset of $\mathcal{U}(F_\Sigma^\mathcal{O})$ as in [59]. We do not provide further details here as computing uniform interpolant is not our main goal.

Theorem 66 provides a way to compute uniform interpolant based on the forests $F_\Sigma^\mathcal{O}$ although we do not implement it. There are mainly two different approaches for computing uniform interpolant: **(i)** the *forgetting-based approach* like LETHE and FAME [78], which compute a uniform interpolant by forgetting all the concepts and roles outside a signature Σ; **(ii)** *generation approach* like `NUI` [45]. `NUI` works only for $\mathcal{EL}$-terminology and when $F_\Sigma^\mathcal{O}$ is finite (has no Σ-loop in their case). It is shown in [14] that `NUI` is much more efficient than LETHE and FAME on $\mathcal{EL}$-terminologies. Moreover, generating uniform interpolants from $F_\Sigma^\mathcal{O}$ described by Theorem 66 is a generalization of `NUI` from $\mathcal{EL}$-terminologies to $\mathcal{EL}$-ontologies as shown below.

Definition 68 *For an acyclic $\mathcal{EL}$-terminology $\mathcal{O}$ and signature Σ, the uniform interpolant $\mathrm{NUI}(\mathcal{O}, \Sigma)$ for $\mathcal{O}$ and Σ computed by `NUI` has the following form:*

$$\mathrm{NUI}(\mathcal{O}, \Sigma) = \{C \sqsubseteq A \mid A \in \Sigma, C \in P_\Sigma(A)\} \cup \{A \sqsubseteq D \mid A \in \Sigma, D \in Q_\Sigma(A)\}$$

where (1) $P_\Sigma(A)$ is inductively defined as follows. Let $Post_\Sigma(A) = \{B \in \Sigma \mid \mathcal{O} \models A \sqsubseteq B\}$, then

- *if $A \sqsubseteq B_1 \sqcap \ldots \sqcap B_n \in \mathcal{O}$, then $P_\Sigma(A) = \left(\bigcup_{1 \leq i \leq n} P_\Sigma(B_i)\right) \cup Post_\Sigma(A)$;*
- *if $A \sqsubseteq \exists r.B \in \mathcal{O}$ and $r \in \Sigma$, then if $B \in \Sigma$, $P_\Sigma(A) = Post_\Sigma(A) \cup \{\exists r.B\}$; if $B \notin \Sigma$, $P_\Sigma(A) = Post_\Sigma(A) \cup \{\exists r.\left(\sqcap_{C \in P_\Sigma(B)} C\right)\}$;*
- *if $\mathcal{O}$ contains no axiom of the form $A \sqsubseteq B_1 \sqcap \ldots \sqcap B_n$ or $A \sqsubseteq \exists r.B$, then $P_\Sigma(A) = Post_\Sigma(A)$.*

(2) $Q_\Sigma(A)$ is inductively defined by as follows. Let $Pre_\Sigma(A) = \{B \in \Sigma \mid \mathcal{O} \models B \sqsubseteq A\}$, then

- *if $B_1 \sqcap \ldots \sqcap B_n \sqsubseteq A \in \mathcal{O}$, then*

  $Q_\Sigma(A) = \{C_1 \sqcap \ldots \sqcap C_n \mid$ *then $C_i = B_i$if $B_i \in \Sigma$, otherwise $C_i \in Q_\Sigma(B)$*$\}$;

- *if $\exists r.B \sqsubseteq A \in \mathcal{O}$ and $r \in \Sigma$, then if $B \in \Sigma$, $Q_\Sigma(A) = \{\exists r.B\}$; if $B \notin \Sigma$, $Q_\Sigma(A) = \{\exists r.C \mid C \in Q_\Sigma(B)\}$;*

- *if $\mathcal{O}$ contains no axiom of the form $B_1 \sqcap \ldots \sqcap B_n \sqsubseteq A$ or $\exists r.B \sqsubseteq A$, $Q_\Sigma(A) = Pre_\Sigma(A)$.*

Proposition 69 *Assume that $\mathcal{O}$ is an acyclic $\mathcal{EL}$-terminology and Σ is a signature. Let $F \subseteq F_\Sigma^\mathcal{O}$ be the subset consisting of all b-trees and maximal f-trees. Then, $\mathrm{NUI}(\mathcal{O}, \Sigma)$ is equivalent to $\mathcal{U}_1(F) \cup \mathcal{U}_2(F)$ (see Definition 65).*

Proof. The proof is provided on page 94. □

4.3 . Pseudo-minimal Modules

This section introduces a new notion of deductive modules, called pseudo-minimal modules, based on the forest $F_\Sigma^\mathcal{O}$. The definition of pseudo-minimal modules relies on *tree-support* and a finite representative subset of $F_\Sigma^\mathcal{O}$ defined below.

4.3.1 . Tree-support

Intuitively, *tree-supports* can be seen as analogs to justifications although they are related to tree derivation whereas justifications are related to axiom inference.

Definition 70 *A tree-support of a tree $\mathtt{t} \in F_\Sigma^\mathcal{O}$ is a sub-ontology of $\mathcal{O}$ defined as the union of the following axiom sets:*

1. *$\{\alpha\}$, for each edge $e \in \mathtt{t}$ labeled by $\alpha \in \mathcal{O}$;*
2. *a justification $J_{A \sqsubseteq B}$ of $A \sqsubseteq B$ in $\mathcal{O}$, for each edge $e \in \mathtt{t}$ labeled by $\mathcal{O} \models A \sqsubseteq B$.*

$\mathtt{Supp}(\mathtt{t})$ *denotes the collection of all tree-supports of* $\mathtt{t}$.

For example, in Example 61, the only tree-support of $\mathtt{t}_1^-$ is $\{\alpha_4, \alpha_5\}$ since all edges in $\mathtt{t}_1^-$ are labeled by $\alpha_4 \in \mathcal{O}$ or $\alpha_5 \in \mathcal{O}$. Different tree-supports can be obtained depending on the different choices of justification $J_{A \sqsubseteq B}$.

Compared to justifications of $\alpha \in \mathcal{U}(F_\Sigma^\mathcal{O})$, tree-supports are easier to compute since

1. The first component $\{\alpha\}$ is obtained directly;
2. Computing justifications of $A \sqsubseteq B$, where A, B are concept names, is easier than computing the justifications for an arbitrary axiom $\alpha \in \mathcal{U}(F_\Sigma^\mathcal{O})$;
3. The computation of tree-supports can be encoded as *Horn-clauses* and solved by efficient SAT tools as shown in the next section.

4.3.2 . Finding a finite representative subset of $F_\Sigma^\mathcal{O}$

As shown in Example 63, the forest $F_\Sigma^\mathcal{O}$ might be an infinite set of tree in some case. In this case, we extract a finite representative subset of $F_\Sigma^\mathcal{O}$ as follows. First, we partition $F_\Sigma^\mathcal{O}$ into three disjoint sets F_1, F_2 and F_3 as follows:

$$
\begin{aligned}
F_1 = & \{\mathtt{t} \in F_\Sigma^\mathcal{O} \mid \mathtt{t} \text{ contains a loop}\} \\
F_2 = & \{\mathtt{t}^+ \in F_\Sigma^\mathcal{O} \backslash F_1 \mid \exists\ \mathtt{t}^- \in F_1 \text{ such that } \mathtt{A}_{\mathtt{t}^-} \notin \Sigma,\ \mathtt{A}_{\mathtt{t}^-} = \mathtt{A}_{\mathtt{t}^+}\} \\
& \cup \{\mathtt{t}^- \in F_\Sigma^\mathcal{O} \backslash F_1 \mid \exists\ \mathtt{t}^+ \in F_1 \text{ such that } \mathtt{A}_{\mathtt{t}^+} \notin \Sigma,\ \mathtt{A}_{\mathtt{t}^+} = \mathtt{A}_{\mathtt{t}^-}\} \\
F_3 = & F_\Sigma^\mathcal{O} \backslash (F_1 \cup F_2).
\end{aligned}
$$

Then F_2, F_3 are finite sets since they do not contain trees with loop, F_1 is infinite iff $F_\Sigma^\mathcal{O}$ is infinite.

Second, we select a finite subset F_1^* of F_1 as follows. Let us say that two f-trees (or b-trees) are *equivalent* iff they share the same set of edge labels. Then $F_1^* \subseteq F_1$ is obtained by selecting one representative tree for each equivalent class in F_1. Then, because the number of labels of the form $\alpha \in \mathcal{O}$ or $\mathcal{O} \models A \sqsubseteq B$ is finite given an ontology $\mathcal{O}$, the number of equivalent classes is finite and thus F_1^* is finite.

Finally, we obtain a finite representative subset

$$F_1^* \cup F_2 \cup F_3 \subseteq F_\Sigma^\mathcal{O}.$$

Note that if $F_\Sigma^\mathcal{O}$ is finite, then there is no tree in $F_\Sigma^\mathcal{O}$ containing loop and thus $F_1 = F_1^* = \emptyset$, then $F_2 = \emptyset$ by definition. Therefore, $F_3 = F_\Sigma^\mathcal{O}$, and thus $F_\Sigma^\mathcal{O}$ is its own finite representative.

4.3.3 . Pseudo-minimal modules

Now, we formally introduce pseudo-minimal modules as follows:

Definition 71 *A pseudo-minimal module for $\mathcal{O}$ and Σ is a minimal element in the collection $\mathcal{S}_\Sigma^\mathcal{O}$ defined by:*

$$\mathcal{S}_\Sigma^\mathcal{O} = \{ \bigcup_{\mathtt{t} \in F_1^* \cup F_2 \cup F'} S_\mathtt{t} \;\Big|\; S_\mathtt{t} \in \mathtt{Supp}(\mathtt{t}),\ F' \subseteq F_3,\ \mathcal{U}(F_2 \cup F') \models \mathcal{U}(F_2 \cup F_3)\}.$$

We can regard pseudo-minimal modules as an approximation of minimal modules because of the following result:

Theorem 72 *A pseudo-minimal module for $\mathcal{O}$ and Σ is a deductive module for $\mathcal{O}$ and Σ. Moreover, if $F_\Sigma^\mathcal{O}$ is finite, then $\mathcal{M}$ is a pseudo-minimal module for $\mathcal{O}$ and Σ iff $\mathcal{M}$ is a minimal module for $\mathcal{O}$ and Σ.*

Proof. The proof is provided on page 92. □

Now, we show that why in the definition of $\mathcal{S}_\Sigma^\mathcal{O}$ (Definition 71 above) we consider $F' \subseteq F_3$. If we do not consider $F' \subseteq F_3$ satisfying $\mathcal{U}(F_2 \cup F') \models \mathcal{U}(F_2 \cup F_3)$ (i.e., let $F' = F_3$), all the elements in $\mathcal{S}_\Sigma^\mathcal{O}$ are still deductive modules for $\mathcal{O}$ and Σ. However, in this case, we can not guarantee that pseudo-minimal modules are minimal modules when $F_\Sigma^\mathcal{O}$ is finite. For example:

Example 73 (Example 67 cont'd) *Note that, in this example, $F_\Sigma^\mathcal{O}$ is finite, then $F_\Sigma^\mathcal{O} = F_3$. If we require $F' = F_3$, then*

$$\mathcal{S}_\Sigma^\mathcal{O} = \{ \bigcup_{\mathtt{t} \in F_\Sigma^\mathcal{O}} S_\mathtt{t} \;\Big|\; S_\mathtt{t} \in \mathtt{Supp(t)} \}.$$

Since $\mathtt{t}_1^+ \in F_\Sigma^\mathcal{O}$ has a unique tree-support

$$S_{\mathtt{t}_1^+} = \{\{\alpha_1, \alpha_2, \alpha_3\}\},$$

all the sets in $\mathcal{S}_\Sigma^\mathcal{O}$ are super sets of $S_{\mathtt{t}_1^+}$ and thus contain α_3. However, as shown in Example 57, α_3 does not belong to any minimal module for $\mathcal{O}$ and Σ. On the other hand, if we allow $F' \subseteq F_3$, we can get rid of $\mathtt{t}_1^+$ because

$$\mathcal{U}(F') \models \mathcal{U}(F_\Sigma^\mathcal{O}) \text{ for } F' = F_\Sigma^\mathcal{O} \setminus \{\mathtt{t}_1^+\}.$$

Therefore we can get rid of α_3.

When $F_\Sigma^\mathcal{O}$ is infinite, we show by our evaluation (see Table 4.4) that pseudo-minimal modules are still very concise. This provides an experimental validation of the minimal module approximation defined by pseudo-minimal module.

Now, on the other hand, one may expect that pseudo-minimal modules cover all minimal modules. That is, for any minimal module $\mathcal{M}$ for $\mathcal{O}$ and Σ, there exists a pseudo-minimal module $\mathcal{M}^p$ for $\mathcal{O}$ and Σ such that $\mathcal{M} \subseteq \mathcal{M}^p$. However, this claim does not hold as the following counter-example shows.

Example 74 *Consider the ontology*

$$\begin{aligned}\mathcal{O} = \{ & A \sqsubseteq A_1 \\ & A \sqsubseteq B_1, \quad B_1 \sqsubseteq A_1 \\ & \exists r.A_3 \sqsubseteq B_1, \quad B_1 \sqsubseteq \exists r.A_2 \\ & A \sqsubseteq B_2, \quad B_2 \sqsubseteq \exists r.B_2, \quad B_2 \sqsubseteq A_2 \\ & A_3 \sqsubseteq B_3, \quad \exists r.B_3 \sqsubseteq B_3, \quad B_3 \sqsubseteq A_1\}\end{aligned}$$

and signature $\Sigma = \{r, A, A_1, A_2, A_3\}$. Then, there are two minimal modules for $\mathcal{O}$ and Σ:

- $\mathcal{M}_1 = \mathcal{O} \setminus \{A \sqsubseteq A_1\}$;
- $\mathcal{M}_2 = \mathcal{O} \setminus \{A \sqsubseteq B_1, B_1 \sqsubseteq A_1\}$.

However, the only pseudo minimal module for $\mathcal{O}$ and Σ is $\mathcal{M}^p = \mathcal{O} \setminus \{A \sqsubseteq A_1\}$. Then $\mathcal{M}_1 \subseteq \mathcal{M}^p$, but $\mathcal{M}_2 \not\subseteq \mathcal{M}^p$ as $A \sqsubseteq A_1 \in \mathcal{M}_2$.

The above case occurs because of insufficient removal of redundant trees in the definition of pseudo-minimal modules. Consider the partition $F_1 \cup F_2 \cup F_3 = F_\Sigma^\mathcal{O}$ introduced in Section 4.3.2. In this example, $F_2 = \emptyset$ and F_1, F_3 are illustrated by Figure 4.5. We observe that the trees $\mathtt{t}_1^+$ and $\mathtt{t}_1^-$ in F_3 are redundant, as $\mathcal{U}(F_1)$ already entails $\mathcal{U}(\{\mathtt{t}_1^+, \mathtt{t}_1^-\})$. However, the current definition of pseudo-minimal modules does not account for this redundancy, and thus, we obtain $\mathcal{M}^p$ as the unique pseudo-minimal module. In contrast, if we remove $\mathtt{t}_1^+$ and $\mathtt{t}_1^-$ from $F_\Sigma^\mathcal{O}$, then we would obtain $\mathcal{M}_1$ and $\mathcal{M}_2$ instead of $\mathcal{M}^p$.

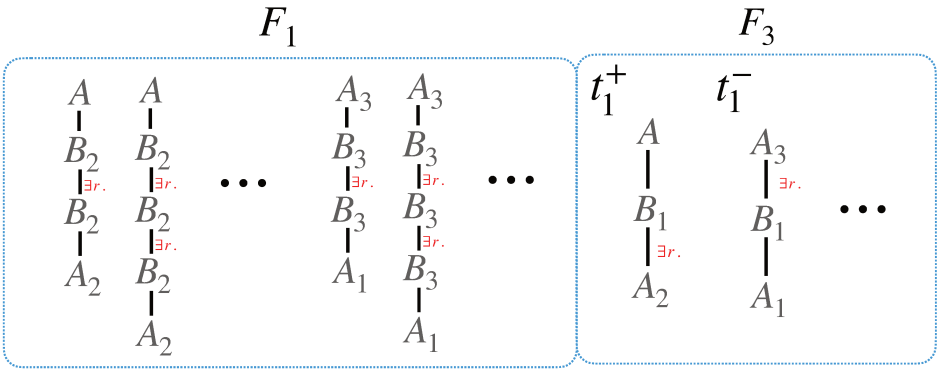


Figure 4.5: some trees in F_1 and F_3

In contrast to pseudo-minimal modules, complete modules are defined directly by minimal modules as follows.

Definition 75 (Complete module) *A* complete module *for an ontology $\mathcal{O}$ and a signature Σ is a sub-ontology $\mathcal{M} \subseteq \mathcal{O}$ such that $\mathcal{M}$ contains all minimal modules for $\mathcal{O}$ and Σ.*

4.4 . `ForMod`: a SAT-based Algorithm

Now, we present our algorithm `ForMod` that constructs a set of Horn-clauses $\mathcal{C}_\Sigma$ and computes pseudo-minimal modules and complete modules using $\mathcal{C}_\Sigma$.

The first step of the algorithm is to compute the finite representative sub-forest $F_1^* \cup F_2 \cup F_3$ of $F_\Sigma^\mathcal{O}$. This computation is essentially simple and done by enumeration of b-trees and f-trees as follows.

4.4.1 . Computing finite representative subset of forest $F_\Sigma^\mathcal{O}$

First, we enumerated all b-trees and f-trees using graph representations of $\mathcal{O}$.

Computing b-trees by hypergraph For computing b-trees, we associate with $\mathcal{O}$ a hypergraph $\mathcal{H}^{\mathcal{O}}{=}(\mathcal{N}_h^{\mathcal{O}}, \mathcal{E}_h^{\mathcal{O}})$, which consists of the node set $\mathcal{N}_h^{\mathcal{O}} := \{N_A \mid A \in \mathsf{N_C}\}$ and the edge set

$$\begin{aligned}\mathcal{E}_h^{\mathcal{O}} :=& \{\{N_{A_1}, \ldots, N_{A_n}\} \rightarrow N_A \mid A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq A \in \mathcal{O}\} \\ & \cup \{N_A \xrightarrow{r} N_B \mid \exists r.A \sqsubseteq B \in \mathcal{O}\} \\ & \cup \{N_A \rightarrow N_B \mid \mathcal{O} \models A \sqsubseteq B\},\end{aligned}$$

where $N_A \xrightarrow{r} N_B$ is an edge with the index $r \in \mathsf{N_R}$. All the edges of the form $N_A \rightarrow N_B$ are computed by classification over $\mathcal{O}$ using ELK [36]. Next, for every direct hyperedge $e = \{N_{A_1}, \ldots, N_{A_n}\} \rightarrow N_A$, we say that

1. $\{A_1, \ldots, A_n\}$ is the *tail* of e, denoted by $Tail(e) = \{A_1, \ldots, A_n\}$;
2. A is the *head* of e, denoted by $Head(e) = A$.

For the case $e = N_A \xrightarrow{r} N_B$ or $N_A \rightarrow N_B$, the tail and head of e are defined in the same way.

By the definitions of b-tree and hyperpath, each b-tree from A to Σ can be regarded as a hyperpath from $S = \{N_B \mid B \in \Sigma\}$ to N_A. The details are presented in the recursive Algorithm 6, which computes b-trees from A to Σ by decomposing them as a single edges $B_1, \cdots, B_n$ to Σ and b-trees from $B_1, \cdots, B_n$ to Σ (Line 3-8 of Algorithm 6). The b-tree from $B_1, \cdots, B_n$ to Σ is computed by recursively apply Algorithm 6 (Line 5 of Algorithm 6). Our implementation is based on the hypergraph package `halp` (avaliable at https://murali-group.github.io/halp/).

Algorithm 6 : `All_b_Trees`

input : an ontology $\mathcal{O}$, a signature Σ, a concept name A
output : T_b^A: all b-trees from A to Σ over $\mathcal{O}$

1 $T_b^A \leftarrow \emptyset$;
2 **for** $e \in \mathcal{E}_h^{\mathcal{O}}$ *such that* $Head(e) = A$ **do**
3 **for** $N_B \in Tail(e)$ **do**
4 **if** $B \notin \Sigma$ **then**
5 $T_b^B \leftarrow$ *All_b_Trees*$(\mathcal{O}, \Sigma, B)$;
6 **end**
7 **end**
8 $T_b^A \leftarrow T_b^A \cup \left(\left(\bigotimes_{N_B \in T(e), B \notin \Sigma} T_b^B\right) \otimes \{\{e\}\}\right)$;
9 **end**
10 **return** T_b^A

Computing f-trees by (directed) graph For computing f-trees, we associate with $\mathcal{O}$ a graph $\mathcal{G}^{\mathcal{O}}=(\mathcal{N},\mathcal{E})$, which consists of a node set $\mathcal{N} := \{N_A \mid A \in \mathsf{N_C}\}$ and an edge set

$$\begin{aligned}\mathcal{E} :=&\{N_A \rightarrow N_B \mid \mathcal{O} \models A \sqsubseteq B\}\\ &\cup \{N_A \xrightarrow{r} N_B \mid A \sqsubseteq \exists r.B \in \mathcal{O}\}.\end{aligned}$$

By definition, an f-tree from A to Σ can be regarded as a union of directed paths from N_A to N_B with $B \in \Sigma$. The details are presented by the recursive Algorithm 7. At line 3 of Algorithm 7, all directed paths from N_A to N_B for some $B \in \Sigma$ are computed by the depth-first search algorithm. Then at line 4, all f-trees from A to Σ are obtained as union of directed paths obtained in line 3.

Algorithm 7 : `All_f_Trees`

input : an ontology $\mathcal{O}$, a signature Σ, a concept name A
output : T_f^A: all f-trees from A to Σ over $\mathcal{O}$

1 $T_f^A \leftarrow \emptyset$;
2 **for** $B \in \Sigma$ **do**
3 $\quad P^{AB} \leftarrow$ *all directed paths from* N_A *to* N_B;
4 $\quad T_f^A \leftarrow T_f^A \bigotimes P^{AB}$;
5 **end**
6 **return** T_f^A

Note that all b-trees (resp. f-trees) in the same equivalent class have the same corresponding sub-graph of $\mathcal{H}^{\mathcal{O}}$ (resp. $\mathcal{G}^{\mathcal{O}}$). Therefore, to extract a finite representative subset $F_1^* \cup F_2 \cup F_3$, it is enough to ignore trees with the repetition edges during the enumeration of trees. It is done by regarding each tree as a set of edges.

4.4.2 . Encoding pseudo-minimal module computation by Horn-clauses

Recall that, all pseudo-minimal modules are minimal elements in the collection $\mathcal{S}_\Sigma^{\mathcal{O}}$. In the following, we encode the extraction of these minimal elements by a set of Horn-clauses $\mathcal{C}_\Sigma$.

We start by associating a literal l to each component (axiom, tree and edge) of the forest $F_\Sigma^{\mathcal{O}}$ as follows

$$\begin{aligned} l_\alpha &\leftrightarrow \alpha \in \mathcal{O}\\ l_\beta &\leftrightarrow \beta \in \mathcal{U}(F_1^* \cup F_2 \cup F_3)\\ l_{\mathtt{t}} &\leftrightarrow \mathtt{t} \in F_\Sigma^{\mathcal{O}}\\ l_e &\leftrightarrow e \in \mathtt{t}\end{aligned}$$

We also introduce a new literal l_Σ to capture pseudo-minimal modules for $\mathcal{O}$ and Σ.

Now, the encoding is decomposed into two parts: **(i)** The first part (items 1, 2, and 3) encodes the computation of the subsets $F' \subseteq F_3$ such that

$$\mathcal{U}(F_2 \cup F') \models \mathcal{U}(F_2 \cup F_3);$$

(ii) The second part (items 4 and 5) encodes the computation of the tree-supports

$$S_\mathtt{t} \in \mathtt{Supp}(\mathtt{t}).$$

In details, $\mathcal{C}_\Sigma$ consists of:

1. $(\wedge_{\beta \in \mathcal{U}(F_2 \cup F_3)}\ l_\beta) \wedge (\wedge_{\mathtt{t} \in F_1^* \cup F_2}\ l_\mathtt{t}) \rightarrow l_\Sigma$;
2. For each axiom $\beta \in \mathcal{U}(F_2 \cup F_3)$, a Horn-clause is built depending on the provenance of β in $\mathcal{U}(F_2 \cup F_3)$ (recall Definition 65):
   (a) if $\beta = \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{A}_{\mathtt{t}^-} \in \mathcal{U}^1(F_2 \cup F_3)$ for some $\mathtt{t}^- \in F_2 \cup F_3$:
   $$l_{\mathtt{t}^-} \rightarrow l_\beta;$$
   (b) if $\beta = \mathtt{A}_{\mathtt{t}^+} \sqsubseteq \mathtt{C}_{\mathtt{t}^+} \in \mathcal{U}^2(F_2 \cup F_3)$ for some $\mathtt{t}^+ \in F_2 \cup F_3$:
   $$l_{\mathtt{t}^+} \rightarrow l_\beta;$$
   (c) if $\beta = \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{C}_{\mathtt{t}^+} \in \mathcal{U}^3(F_2 \cup F_3)$ for some $\mathtt{t}^-, \mathtt{t}^+ \in F_2 \cup F_3$:
   $$l_{\mathtt{t}^+} \wedge l_{\mathtt{t}^-} \rightarrow l_\beta.$$
3. For each $\beta \in \mathcal{U}(F_2 \cup F_3)$ and each justification $\{\beta_1, \ldots, \beta_n\}$ of β with respect to the ontology $\mathcal{U}(F_2 \cup F_3)$:
   $$l_{\beta_1} \wedge \ldots \wedge l_{\beta_n} \rightarrow l_\beta;$$
4. For each $\mathtt{t} \in F_1^* \cup F_2 \cup F_3$:
   $$(\wedge_{e \in t} l_e) \rightarrow l_\mathtt{t};$$
5. For each edge $e \in \mathtt{t}$, where $\mathtt{t} \in F_1^* \cup F_2 \cup F_3$:
   (a) if $\mathtt{lab}(e)$ is $\mathcal{O} \models A \sqsubseteq B$, where $J_{A \sqsubseteq B} \subseteq \mathcal{O}$ is a justification of $A \sqsubseteq B$:
   $$(\wedge_{\alpha \in J_{A \sqsubseteq B}} l_\alpha) \rightarrow l_e; \tag{4.1}$$
   (b) if $\mathtt{lab}(e)$ is $\alpha \in \mathcal{O}$:
   $$l_\alpha \rightarrow l_e. \tag{4.2}$$

Above, for clarity, building the clauses of item 5(a) is presented using justifications. In the implementation, the computation of these justifications is itself encoded by a set of Horn-clauses as in [75, 37].

Example 76 (Example 61 and 67 cont'd) *For simplicity, we assume that*

$$F_\Sigma^{\mathcal{O}} = \{\mathtt{t}_1^+, \mathtt{t}_2^+, \mathtt{t}_1^-\}$$

and

$$\mathcal{U}(F_\Sigma^{\mathcal{O}}) = \{\beta_1', \beta_2', \beta_3'\}.$$

Indeed, this simplification is based on the optimization steps that are explained later. Let $e_1, \ldots, e_7$ be the edges of the trees in $F_\Sigma^{\mathcal{O}}$ as illustrated below. Then the set of Horn-clauses $\mathcal{C}_\Sigma$ is shown in Table 4.1.

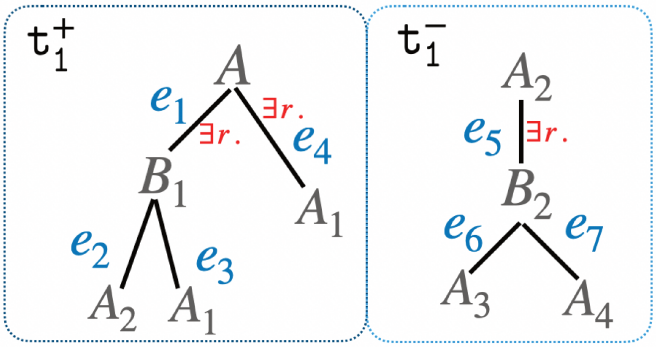


Table 4.1: Clause set $\mathcal{C}_\Sigma$

1:	$l_{\beta_1'} \wedge l_{\beta_2'} \wedge l_{\beta_3'} \rightarrow l_\Sigma$;
2(a):	$l_{\mathtt{t}_1^-} \rightarrow l_{\beta_1'}$;
(b):	$l_{\mathtt{t}_1^+} \rightarrow l_{\beta_2'}$, $l_{\mathtt{t}_2^+} \rightarrow l_{\beta_3'}$;
3:	$l_{\beta_2'} \rightarrow l_{\beta_3'}$, $l_{\beta_3'} \rightarrow l_{\beta_2'}$;
4:	$l_{e_1} \wedge l_{e_2} \wedge l_{e_3} \wedge l_{e_4} \rightarrow l_{\mathtt{t}_1^+}$,
	$l_{e_1} \wedge l_{e_2} \wedge l_{e_3} \rightarrow l_{\mathtt{t}_2^+}$, $l_{e_5} \wedge l_{e_6} \wedge l_{e_7} \rightarrow l_{\mathtt{t}_1^-}$
5(a):	$l_{\alpha_2} \rightarrow l_{e_2}$, $l_{\alpha_2} \rightarrow l_{e_3}$,
(b):	$l_{\alpha_1} \rightarrow l_{e_1}$, $l_{\alpha_3} \rightarrow l_{e_4}$,
	$l_{\alpha_4} \rightarrow l_{e_5}$, $l_{\alpha_5} \rightarrow l_{e_6}$, $l_{\alpha_5} \rightarrow l_{e_7}$;

4.4.3 . Computing complete and pseudo-minimal modules

Let the **answer variables** be the literals l_α associated with axioms $\alpha \in \mathcal{O}$. Pseudo-minimal modules are computed as follows. First, we apply *resolution* over

$$\mathcal{C}_\Sigma \cup \{\neg l_\Sigma\}.$$

Then we obtain a set of clauses denoted by $\mathcal{C}_\Sigma^f$. Assuming

$$\mathtt{M}_\Sigma \subseteq \mathcal{C}_\Sigma^f$$

is the subset of *minimal clauses* composed of answer variables only. We say that a clause c_1 is *minimal* if there does not exist another clause c_2 such that all the literals in c_2 are also in c_1. By the construction of $\mathtt{M}_\Sigma$ and definition of pseudo-minimal module, we have:

Proposition 77 *$\mathcal{M}=\{\alpha_1, \ldots, \alpha_k\}$ is a pseudo-minimal module for $\mathcal{O}$ and Σ iff we have*

$$(\wedge_{i=1}^{k} l_{\alpha_i}) \rightarrow \bot \in \mathtt{M}_\Sigma.$$

For Example 76, we obtain

$$\mathtt{M}_\Sigma = \{l_{\alpha_1} \wedge l_{\alpha_2} \wedge l_{\alpha_4} \wedge l_{\alpha_5} \rightarrow \bot\}$$

by resolution over $\mathcal{C}_\Sigma \cup \{\neg l_\Sigma\}$. Therefore, the only pseudo-minimal module for $\mathcal{O}$ and Σ is

$$\{\alpha_1, \alpha_2, \alpha_4, \alpha_5\},$$

which is also the unique minimal module for $\mathcal{O}$ and Σ since the corresponding forest $F_\Sigma^\mathcal{O}$ is finite.

By definition of $\mathcal{S}_\Sigma^\mathcal{O}$, we have

Corollary 78 *$\mathcal{M}^c=\{\alpha \in \mathcal{O} \mid l_\alpha$ is an answer variable in $\mathcal{C}_\Sigma\}$ is a complete module for $\mathcal{O}$ and Σ.*

For Example 76, the complete module for $\mathcal{O}$ and Σ is $\mathcal{M}^c = \mathcal{O}$.

4.4.4 . Optimization

As discussed in Example 76, we can reduce the size of $F_\Sigma^\mathcal{O}$ by ignoring some redundant trees. For example, we can ignore those trees that do not contribute to the generation of axioms in $\mathcal{U}(F_\Sigma^\mathcal{O})$. Furthermore, we can ignore some sub-f-trees based on the following result:

Corollary 79 *Theorems 66 and 72 still hold if we ignore the f-trees $\mathtt{t}^+ \in F_\Sigma^\mathcal{O}$ that satisfy one of the following conditions:*

1. *t^+ has an edge, starting from the root $\mathtt{root}(\mathtt{t}^+)$, which is not a $\exists r$-edge;*
2. *$\mathtt{t}^+$ is a proper sub-tree of a f-tree $\mathtt{t}_1^+ \in F_\Sigma^\mathcal{O}$ such that $\mathtt{root}(\mathtt{t}) = \mathtt{root}(\mathtt{t}_1^+)$ and $\not\models \mathtt{C}_{\mathtt{t}^+} \equiv \mathtt{C}_{\mathtt{t}_1^+}$.*

Proof. The proof is provided on page 92. □

In Example 76, $\mathtt{t}_1^+$ has 6 different sub-trees, but only $\mathtt{t}_2^+$ satisfies

$$\models \mathtt{C}_{\mathtt{t}_1^+} \equiv \mathtt{C}_{\mathtt{t}_2^+}.$$

Therefore, we can ignore the other five sub-trees and thus now

$$F_\Sigma^\mathcal{O} = \{\mathtt{t}_1^+, \mathtt{t}_2^+, \mathtt{t}_1^-\}.$$

4.5 . Evaluation

We implemented a prototype ForMod of our algorithm in Python and evaluated it over three real-world ontologies: Snomed CT (version Jan 2016), Snomed CT (version Jan 2021)(available at https://www.snomed.org), and NCI (version 16.03d) (available at http://evs.nci.nih.gov/ftp1/NCI_Thesaurus). We denote them as *sn16, sn21, nci*, respectively. Here, *sn16* and *nci* are two $\mathcal{EL}$-terminologies containing 317891 and 165341 axioms respectively. And *sn21* is an $\mathcal{EL}$-ontology with 362638 axioms. All the experiments run on a machine with an Intel Xeon Core 4 Duo CPU 2.50 GHz with 64 GiB of RAM.

For each ontology $\mathcal{O}$, we run the experiments over 2 sets $\Sigma_n^{\mathcal{O}}$ of 1000 randomly generated signatures, where each signature has n concepts ($n \in \{50, 100\}$) and 10 roles.

Pesudo-minimal modules Recall that Zoom [16] is the state-of-the-art algorithm that computes all the minimal modules but only for acyclic $\mathcal{EL}$-terminologies. First, we compare all pseudo-minimal modules computed by ForMod with all minimal modules computed by Zoom.

For each signature, we set the total run-time limit of 600s. The success rates (i.e., the percentage of completed experiments within the time limit) of ForMod and Zoom are summarized in Table 4.2. We can see that the success rate of ForMod is from 13.3% to 46.7% higher than Zoom. Note that Zoom does not work for *sn21*, which is not an $\mathcal{EL}$-terminology.

Table 4.2: Success rate (%)

	Σ_{50}^{sn16}	Σ_{100}^{sn16}	Σ_{50}^{nci}	Σ_{100}^{nci}	Σ_{50}^{sn21}	Σ_{100}^{sn21}
Zoom	57.1	32.5	79.0	57.3	-	-
ForMod	**83.8**	**79.2**	**92.3**	**74.2**	**98.9**	**88.1**

Table 4.3 summarizes the time-cost comparison over signatures that are solved successfully by both ForMod and Zoom. We highlight that, for these signatures, the corresponding forests are indeed finite. Therefore, the pseudo-minimal modules are indeed minimal modules by Theorem 72. According to Table 4.3, ForMod is 99.79 times faster than Zoom on average. Note that, as discussed in [16], Zoom spends most of its running time (94.6% on average) on computing justifications using Beacon [2], which is less efficient than the resolution we use. However, even if we ignore the time cost of Beacon, i.e. only consider 5.4% computation time of Zoom, ForMod is still 5.67, 7.48, 3.72, 5.33 times faster than Zoom on average for Σ_{50}^{sn16}, Σ_{100}^{sn16}, Σ_{50}^{nci}, Σ_{100}^{nci}, respectively.

Second, in our experiments, there are 66 signatures in $\Sigma_{50}^{nci} \cup \Sigma_{100}^{nci}$ for which $F_{\Sigma}^{\mathcal{O}}$ is infinite, and 43 of them are solved within the time limit by ForMod, but all

Table 4.3: Time cost (max / min / mean / median)

Time(s)	ForMod	Zoom
Σ_{50}^{sn16}	**17.20 / 1.09 / 1.77 / 1.73**	558.76 / 34.85 / 186.04 / 143.53
Σ_{100}^{sn16}	**9.75 / 1.29 / 2.18 / 2.10**	563.24 / 71.38 / 302.24 / 294.19
Σ_{50}^{nci}	**8.89 / 0.74 / 1.32 / 1.28**	560.89 / 7.81 / 91.08 / 60.42
Σ_{100}^{nci}	**12.47 / 0.89 / 1.47 / 1.42**	576.35 / 15.49 / 145.32 / 105.92

of them are time-out for Zoom. So we compare pseudo-minimal modules of these signatures with $\top\bot^*$-modules implemented by OWL API [27] instead. The result is summarized in Table 4.4. We observe that the pseudo-minimal modules are still very concise in size and significantly smaller than the $\top\bot^*$-modules.

Table 4.4: Signatures with infinite $F_\Sigma^{\mathcal{O}}$ (max / min / mean)

Module size	*pseudo-minimal modules*	$\top\bot^*$-modules
Σ_{50}^{nci}	80 / 12 / 31.06	7499 / 6340 / 7126.24
Σ_{100}^{nci}	111 / 9 / 52.81	8091 / 1936 / 7317.31

Complete modules Here, we compare our complete modules generated by ForMod with $\top\bot^*$-modules. Note that, for all signatures tested, the corresponding complete modules and $\top\bot^*$-modules are all computed within $32s$, i.e., the success rate is 100%.

The size comparison of those modules is shown in Table 4.5. We can see that the size of complete modules is significantly smaller than that of $\top\bot^*$-modules (103.5 times smaller on average) for all ontologies.

Table 4.5: Mean size of different deductive modules

Module Size	Σ_{50}^{sn16}	Σ_{100}^{snt16}	Σ_{50}^{nci}	Σ_{100}^{nci}	Σ_{50}^{sn21}	Σ_{100}^{snt21}
$\top\bot^*$-module	6628.21	13159.89	5560.64	7327.67	4384.08	15845.43
Complete	74.53	140.68	52.02	73.91	88.48	81.28
pseudo-minimal	8.80	20.97	24.36	37.53	7.60	16.11

Table 4.6 summarizes the time cost comparison between complete modules and $\top\bot^*$-modules. It shows that the computation of complete modules is faster than that of $\top\bot^*$-modules except for the worst cases (i.e., the maximal time cost).

4.6 . Proofs

Table 4.6: Time cost (max / min / mean / median)

Time(s)	Complete module (ForMod)	$\top\bot^*$-module (OWL API)
$\sum_{50}^{sn16}$	27.10 / **1.09** / **2.92** /**1.95**	**9.36** / 5.60 / 6.55 / 6.45
$\sum_{100}^{sn16}$	31.60 / **1.29** / **4.76** / **2.60**	**9.68** / 5.76 / 6.74 / 6.65
$\sum_{50}^{nci}$	3.45 / **0.74** / **1.32** / **1.30**	**2.88** / 1.16 / 1.90 / 1.97
$\sum_{100}^{nci}$	23.49 / **0.89** / **1.46** / **1.42**	**2.90** / 1.28 / 1.98 / 1.97
$\sum_{50}^{sn21}$	28.06 / **1.34** / **2.62** / **2.27**	**12.90** / 3.55 / 8.57 / 8.88
$\sum_{100}^{sn21}$	17.71 / **2.10** / **3.48** / **3.17**	**14.19** / 3.99 / 9.28 / 9.71

Proofs of Theorem 66

Theorem 80 *For any axiom α with sig$(\alpha) \subseteq \Sigma$, we have*

$$\mathcal{O} \models \alpha \text{ iff } \mathcal{U}(F_\Sigma^{\mathcal{O}}) \models \alpha.$$

Therefore, $\mathcal{U}(F_\Sigma^{\mathcal{O}})$ is a uniform interpolant for $\mathcal{O}$ and Σ if $F_\Sigma^{\mathcal{O}}$ is finite.

$$\mathcal{R}_1: \frac{A \sqsubseteq A_1, \ \ldots, \ A \sqsubseteq A_n}{A \sqsubseteq B} : A_1 \sqcap \ldots \sqcap A_n \sqsubseteq B \in \mathcal{O}$$

$$\mathcal{R}_2: \frac{A \sqsubseteq A_1}{A \sqsubseteq \exists r.B} : A_1 \sqsubseteq \exists r.B \in \mathcal{O}$$

$$\mathcal{R}_3: \frac{A \sqsubseteq \exists r.B_1, \ B_1 \sqsubseteq B_2}{A \sqsubseteq B} : \exists r.B_2 \sqsubseteq B \in \mathcal{O}$$

Table 4.7: Inference rules over $\mathcal{EL}$-ontology.

Our proof is based on the complete and sound inference rules presented in Table 4.7, which are obtained by adapting Table 2.1 (Section 2.3) to $\mathcal{EL}$-ontologies. The proof of Theorem 66 is done in two steps: **First**, we introduce two notions: (i) *normalized ontologies for concepts*, which makes it possible to apply the rules in Table 4.7 for complex concepts $C, D \notin \mathsf{N}_C$; and (ii) *b*-trees* and *f*-trees*, which are weaker forms of b-trees and f-trees. **Second**, we prove three lemmas: (i) Lemma 89, which is a stronger version of Theorem 66 but only works for b*-trees and f*-trees; (ii) Lemma 90 shows that Lemma 89 still holds if b*-trees are replaced by b-trees; (iii) Lemma 91 similarly shows that Lemma 89 still holds if f*-trees are replaced by f-trees. Finally, Theorem 66 is direct conclusion of these three lemmas.

Normalized ontologies for concepts

In order to apply the rules in Table 4.7 for complex concepts $C \notin \mathsf{N}_C$, we construct two normalized ontologies, $\mathcal{O}^+(C)$ and $\mathcal{O}^-(C)$, for each $\mathcal{EL}$-concept C as follows.

Next, we use $\overline{C} \in \mathsf{N}_C$ to denote a concept name associated with the given complex concept C. For a concept name $A \in \mathsf{N}_C$, we assume $\overline{A} = A$. Our primary objective is to establish the equivalence between the subsumption relationship expressed by

$$\mathcal{O} \models C \sqsubseteq D$$

and the derivation generated through the application of inferences listed in Table 4.7 expressed by

$$\mathcal{O} \cup \mathcal{O}^+(C) \cup \mathcal{O}^-(D) \ \vdash \ \overline{C} \sqsubseteq \overline{D}$$

.

Definition 81 *For an $\mathcal{EL}$-concept C, $\mathcal{O}^+(C)$ is a normalized ontology inductively defined as follows:*

1. *If C is a concept name, then $\mathcal{O}^+(C) = \emptyset$;*
2. *If $C = C_1 \sqcap \ldots \sqcap C_n$, then $\mathcal{O}^+(C)$ is defined by:*

$$\mathcal{O}^+(C) = \left(\bigcup_{i=1}^{n} \mathcal{O}^+(C_i) \right) \cup \{ \ \overline{C} \sqsubseteq \overline{C}_1, \ \ldots, \ \overline{C} \sqsubseteq \overline{C}_n \ \}$$

3. *If $C = \exists r.C_1$, then $\mathcal{O}^+(C)$ is defined by:*

$$\mathcal{O}^+(C) = \mathcal{O}^+(C_1) \cup \{ \ \overline{C} \sqsubseteq \exists r.\overline{C}_1 \ \}.$$

Example 82 *Consider the concept $C = \exists r.(A_1 \sqcap A_2) \sqcap B_1 \sqcap B_2$. Assume $C_1 = \exists r.(A_1 \sqcap A_2)$ and $C_2 = A_1 \sqcap A_2$, then we have*

$$\begin{aligned} C &= C_1 \sqcap B_1 \sqcap B_2 \\ C_1 &= \exists r.C_2 \\ C_2 &= A_1 \sqcap A_2 \end{aligned}$$

Therefore, we have

$$\begin{aligned} \mathcal{O}^+(C) &= \mathcal{O}^+(C_1) \cup \{\overline{C} \sqsubseteq \overline{C}_1, \ \overline{C} \sqsubseteq B_1, \ \overline{C} \sqsubseteq B_2\} \\ \mathcal{O}^+(C_1) &= \mathcal{O}^+(C_2) \cup \{\overline{C}_1 \sqsubseteq \exists r.\overline{C}_2\} \\ \mathcal{O}^+(C_2) &= \{\overline{C}_2 \sqsubseteq A_1, \ \overline{C}_2 \sqsubseteq A_2\} \end{aligned}$$

Similarly, we define $\mathcal{O}^-(C)$ as follows.

Definition 83 *For an $\mathcal{EL}$-concept C, $\mathcal{O}^-(C)$ is a normalized ontology inductively defined as follows:*

1. *If C is a concept name, then $\mathcal{O}^-(C) = \emptyset$;*

2. *If* $C = C_1 \sqcap \ldots \sqcap C_n$, *then* $\mathcal{O}^-(C)$ *is defined by:*

$$\mathcal{O}^-(C) = \left(\bigcup_{i=1}^{n} \mathcal{O}^-(C_i)\right) \cup \{\ \overline{C}_1 \sqcap \ldots \sqcap \overline{C}_n \sqsubseteq \overline{C}\ \};$$

3. *If* $C = \exists r.C_1$, *then* $\mathcal{O}^-(C)$ *is defined by:*

$$\mathcal{O}^-(C) = \mathcal{O}^-(C_1) \cup \{\ \exists r.\overline{C}_1 \sqsubseteq \overline{C}\ \}.$$

Example 84 *Consider the concept* $D = (\exists r.\exists s.B_3) \sqcap B_4$. *Assume* $D_1 = \exists r.\exists s.B_3$ *and* $D_2 = \exists s.B_3$, *then we have*

$$D = D_1 \sqcap B_4$$
$$D_1 = \exists r.D_2$$
$$D_2 = \exists s.B_3$$

and thus

$$\mathcal{O}^-(D) = \mathcal{O}^-(D_1) \cup \{\ \overline{D}_1 \sqcap B_4 \sqsubseteq \overline{D}\},$$
$$\mathcal{O}^-(D_1) = \mathcal{O}^-(D_2) \cup \{\ \exists r.\overline{D}_2 \sqsubseteq \overline{D}_1\},$$
$$\mathcal{O}^-(D_2) = \{\ \exists s.B_3 \sqsubseteq \overline{D}_2\ \}.$$

We have the following result:

Theorem 85 *Given an* $\mathcal{EL}$*-ontology* $\mathcal{O}$. *Let* C, D *be two* $\mathcal{EL}$*-concepts, and let*

$$\overline{\mathcal{O}}_{CD} = \mathcal{O} \cup \mathcal{O}^+(C) \cup \mathcal{O}^-(D).$$

Then we have

$$\mathcal{O} \models C \sqsubseteq D \text{ iff } \overline{\mathcal{O}}_{CD} \models \overline{C} \sqsubseteq \overline{D}.$$

Proof. Each model $\mathcal{I}$ of $\mathcal{O}$ can be extended to a model $\overline{\mathcal{I}}$ of $\overline{\mathcal{O}}_{CD}$ by setting

1. $A^{\overline{\mathcal{I}}} = A^{\mathcal{I}}$, $r^{\overline{\mathcal{I}}} = r^{\mathcal{I}}$ for every $A, r \in$ *sig*$(\mathcal{O})$;
2. $\overline{C}_i^{\overline{\mathcal{I}}} = C_i^{\mathcal{I}}, \overline{D}_j^{\overline{\mathcal{I}}} = D_j^{\mathcal{I}}$ for each $\overline{C}_i, \overline{D}_j \in$ *sig*$(\overline{\mathcal{O}}_{CD})$, where C_i is a sub-concept of C, D_j is a sub-concept of D;

Therefore, if $\overline{\mathcal{O}}_{CD} \models \overline{C} \sqsubseteq \overline{D}$, then we have $C^{\mathcal{I}} = \overline{C}^{\overline{\mathcal{I}}} \subseteq \overline{D}^{\overline{\mathcal{I}}} = D^{\mathcal{I}}$. Since this argument works for any model $\mathcal{I}$ of $\mathcal{O}$, we have

$$\overline{\mathcal{O}}_{CD} \models \overline{C} \sqsubseteq \overline{D} \Rightarrow \mathcal{O} \models C \sqsubseteq D.$$

On the other hand, for any model $\overline{\mathcal{I}}$ of $\overline{\mathcal{O}}_{CD}$, we have $\overline{C}^{\overline{\mathcal{I}}} \subseteq C^{\overline{\mathcal{I}}}$, $D^{\overline{\mathcal{I}}} \subseteq \overline{D}^{\overline{\mathcal{I}}}$ by the definitions of $\mathcal{O}^+(C)$ and $\mathcal{O}^-(D)$. Therefore, we have

$$\begin{aligned}
\mathcal{O} \models C \sqsubseteq D &\Rightarrow \overline{\mathcal{O}}_{CD} \models C \sqsubseteq D \text{ (since } \mathcal{O} \subset \overline{\mathcal{O}}_{CD}\text{)},\\
&\Rightarrow C^{\overline{\mathcal{I}}} \subseteq D^{\overline{\mathcal{I}}}, \text{ for any model } \overline{\mathcal{I}} \text{ of } \overline{\mathcal{O}}_{CD},\\
&\Rightarrow \overline{C}^{\overline{\mathcal{I}}} \subseteq \overline{D}^{\overline{\mathcal{I}}}, \text{ for any model } \overline{\mathcal{I}} \text{ of } \overline{\mathcal{O}}_{CD},\\
&\Rightarrow \overline{\mathcal{O}}_{CD} \models \overline{C} \sqsubseteq \overline{D}.
\end{aligned}$$

This concludes our proof. □

Remark 86 *Note that a concept name $A \in sig(\overline{\mathcal{O}}_{CD})$ has one of the following forms:*

1. *$A \in sig(\mathcal{O})$ is a concept name in $\mathcal{O}$;*
2. *$A = \overline{C}_i$ is a new concept name introduced in $\mathcal{O}^+(C)$;*
3. *$A = \overline{D}_j$ is a new concept name introduced in $\mathcal{O}^-(D)$.*

b*-trees and f*-trees

Here, we introduce two kinds of trees: *b*-tree* or *f*-tree*, where

1. *b*-tree* is defined by (i) replacing the second requirement of Definition 58 by: $\mathtt{lab}(\mathtt{n}) \in \Sigma$ for leaf $\mathtt{n}$ of $\mathtt{t}^+$; (ii) removing the requirement "B_i is *non-primitive* or $B_i \in \Sigma$" in Definition 58 case 3(b);
2. *f*-tree* is defined by (i) replacing the second requirement of Definition 60 by: $\mathtt{lab}(\mathtt{n}) \in \Sigma$ for leaf $\mathtt{n}$ of $\mathtt{t}^-$; (ii) removing the requirement "B_1 is *non-primitive*" in Definition 60 case 3(b).

The formal definitions of b*-tree and f*-tree are given below:

Definition 87 *A labeled tree $\mathtt{t}^+$ is a f*-tree from A to Σ over $\mathcal{O}$ iff:*

1. *the label of the root of $\mathtt{t}^+$ is A (i.e., $\mathtt{A}_{\mathtt{t}^+} = A$);*
2. *$\mathtt{lab}(\mathtt{n}) \in \Sigma$ for leaf $\mathtt{n}$ of $\mathtt{t}^+$;*
3. *if the child set of a node $\mathtt{n}_0 \in \mathtt{t}^+$ is*

$$\{\mathtt{n}_1, \ldots, \mathtt{n}_m\}$$

and

$$B_i = \mathtt{lab}(\mathtt{n}_i), 0 \leq i \leq m,$$

then for each $0 \leq i \leq m$, one of the following conditions holds:

(a) $B_0 \sqsubseteq \exists r.B_i \in \mathcal{O}$ and $r \in \Sigma$,

(b) $\mathcal{O} \models B_0 \sqsubseteq B_i$.

The edge e from $\mathtt{n}_0$ to $\mathtt{n}_i$ is labeled by the condition that generates e.

Definition 88 *A labeled tree $\mathtt{t}^-$ is a b*-tree from A to Σ over $\mathcal{O}$ iff:*

1. *the label of the root of $\mathtt{t}^-$ is A (i.e., $\mathtt{A}_{\mathtt{t}^-} = A$);*
2. *$\mathtt{lab}(\mathtt{n}) \in \Sigma$ for leaf $\mathtt{n}$ of $\mathtt{t}^-$;*

3. *if the child set of a node* $\mathtt{n}_0 \in \mathtt{t}^-$ *is*

$$\{\mathtt{n}_1, \ldots, \mathtt{n}_m\}$$

and

$$B_i = \mathtt{lab}(\mathtt{n}_i), 0 \leq i \leq m,$$

then one of the following conditions holds:

(a) $m = 1$, $\exists r.B_1 \sqsubseteq B_0 \in \mathcal{O}, r \in \Sigma$;

(b) $m = 1$, $\mathcal{O} \models B_1 \sqsubseteq B_0$;

(c) $m > 1$, $B_1 \sqcap \ldots \sqcap B_m \sqsubseteq B_0 \in \mathcal{O}$.

Again, here, the edge e *from* $\mathtt{n}_0$ *to* $\mathtt{n}_i$ *is labeled by the condition generating* e.

As in Definition 64, we associate each b*-tree or f*-tree $\mathtt{t}$ with a $\mathcal{EL}$-concept $\mathtt{C}_\mathtt{t}$.

Proof of Theorem 66

Now, we prove Theorem 66 using Lemma 89, 90 and 91.

Lemma 89 *Let* $\mathcal{O}$ *be an normalized* $\mathcal{EL}$*-ontology, and let* C, D *be two* $\mathcal{EL}$*-concepts such that* $\mathcal{O} \models C \sqsubseteq D$. *Then, there exists a set* $T_{C \sqsubseteq D}$ *of b*-trees and f*-trees such that*

$$\overline{\mathcal{U}}(T_{C \sqsubseteq D}) \models C \sqsubseteq D,$$

where

- $T_{C \sqsubseteq D}$ *consists of two kinds of trees:*
  1. *b*-trees from* A *to sig*(C),
  2. *f*-trees from* A *to sig*(D),

  for some concept names $A \in$ *sig*$(\mathcal{O})$;
- $\overline{\mathcal{U}}(T_{C \sqsubseteq D})$ *is defined by*

$$\begin{aligned}\overline{\mathcal{U}}(T_{C \sqsubseteq D}) = &\{\mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{A}_{\mathtt{t}^-} | \mathtt{A}_{\mathtt{t}^-} \in \textit{sig}(C \sqsubseteq D), \mathtt{t}^- \in T_{C \sqsubseteq D}\} \\ &\cup \{\mathtt{A}_{\mathtt{t}^+} \sqsubseteq \mathtt{D}_{\mathtt{t}^+} \mid \mathtt{A}_{\mathtt{t}^+} \in \textit{sig}(C \sqsubseteq D), \mathtt{t}^+ \in T_{C \sqsubseteq D}\} \\ &\cup \{\mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{D}_{\mathtt{t}^+} \mid \mathtt{A}_{\mathtt{t}^-} = \mathtt{A}_{\mathtt{t}^+} \notin \textit{sig}(C \sqsubseteq D), \\ &\qquad \mathtt{t}^-, \mathtt{t}^+ \in T_{C \sqsubseteq D}\}.\end{aligned}$$

  Recall that $\mathtt{A}_\mathtt{t}$ *is the label of the root node of* $\mathtt{t}$.

Moreover:

1. *If $C \in \mathsf{N}_C$ is a concept name, we can require that $T_{C \sqsubseteq D}$ contains only one f*-tree.*

2. *If $D \in \mathsf{N}_C$ is concept name, we can require that $T_{C \sqsubseteq D}$ contains only one b*-tree.*

Proof. We prove the lemma by induction over $k(\overline{C} \sqsubseteq \overline{D})$, which is the minimal number of inferences (generated by Table 4.7) needed to derive $\overline{C} \sqsubseteq \overline{D}$ from $\overline{\mathcal{O}}_{CD}$.

1. If $k(\overline{C} \sqsubseteq \overline{D}) = 0$, then C, D are concept names. The lemma holds directly;

2. Now, assume that the lemma is true when $k(\overline{C}' \sqsubseteq \overline{D}') < K$. Next, we prove that the lemma is still true when $k(\overline{C} \sqsubseteq \overline{D}) = K$.

   Assume that $k(\overline{C} \sqsubseteq \overline{D}) = K$. Let

   $$\rho_1, \ldots, \rho_K$$

   be a minimal sequence of inferences generated by the rules in Table 4.7 that derives $\overline{C} \sqsubseteq \overline{D}$ from $\overline{\mathcal{O}}_{CD}$. We are going to show that there exists a set $T_{C \sqsubseteq D}$ of b*-trees or f*-trees such that $\overline{\mathcal{U}}(T_{C \sqsubseteq D}) \models C \sqsubseteq D$.

   There are two different situations:

   (a) If the last inference ρ_K is of the form

   $$(\mathcal{R}_1) \quad \rho_K : \frac{\overline{C} \sqsubseteq A_1, \ \ldots, \ \overline{C} \sqsubseteq A_n}{\overline{C} \sqsubseteq \overline{D}} : A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq \overline{D} \in \overline{\mathcal{O}_{CD}}$$

   Then, $k(\overline{C} \sqsubseteq A_i) < k(\overline{C} \sqsubseteq \overline{D}) = K$. By applying the induction assumption over $\overline{C} \sqsubseteq A_i$, we have:

   i. if A_i is a concept name in $\mathit{sig}(\mathcal{O})$, then there exists $T_{C \sqsubseteq A_i} = \{\mathtt{t}_\mathtt{i}^-\}$ such that
   $$\overline{\mathcal{U}}(T_{C \sqsubseteq A_i}) \models C \sqsubseteq A_i;$$

   ii. if $A_i = \overline{D}_m$ for some concept D_m, which is sub-concept of D (thus, $\mathit{sig}(D_m) \subseteq \mathit{sig}(D)$), then there exists $T_{C \sqsubseteq D_m}$ such that
   $$\overline{\mathcal{U}}(T_{C \sqsubseteq D_m}) \models C \sqsubseteq D_m.$$

   Now, two cases arise concerning D.

   - $D \in \mathsf{N}_C$ is a concept name, then $A_1, \ldots, A_n$ are concept names in $\mathit{sig}(\mathcal{O})$. Let $\mathtt{t}^-$ be the b*-tree from $\mathit{sig}(C)$ to D obtained by concatenating all b*-tree $\mathtt{t}_\mathtt{i}^- \in T_{C \sqsubseteq A_i}, 1 \leq i \leq n$ with the edge generated by
   $$A_1 \sqcap A_2 \sqcap \ldots \sqcap A_n \sqsubseteq D \in \mathcal{O}.$$
   Then we have $\overline{\mathcal{U}}(T_{C \sqsubseteq D}) \models C \sqsubseteq D$, and thus, $T_{C \sqsubseteq D} = \{\mathtt{t}^-\}$ is what we want. Note that $T_{C \sqsubseteq D}$ also satisfies Item 2 of the lemma.
   - $D \notin \mathsf{N}_C$. Note that for A_i in Case i and D_m in Case ii, we have $A_i \in \mathit{sig}(D)$ and $\mathit{sig}(D_m) \subseteq \mathit{sig}(D)$. Let $T_{C \sqsubseteq D}$ be the union of all trees in $T_{C \sqsubseteq A_i}$ and $T_{C \sqsubseteq D_m}$ above. Then, $T_{C \sqsubseteq D}$ is what we want.

Moreover, if $C \in \mathsf{N}_C$ is a concept name, by Item 1 of the Lemma, we can assume that $T_{C \sqsubseteq D_m}$ in Case ii contains only one f*-tree from C to $\mathit{sig}(D_m)$. Let $\mathtt{t}^+$ be the f*-tree with the root labelled by C, and consisting of the following edges:

- edges with label $\mathcal{O} \models C \sqsubseteq A_i, i = 1, 2, \ldots, n$;
- edges of f*-trees in $T_{C \sqsubseteq D_m}$.

Then $T_{C \sqsubseteq D} = \{\mathtt{t}^+\}$ satisfies the requirement of Item 1 in the Lemma, and it is what we want.

(b) The case where the last inference is generated by rule $\mathcal{R}_3$ never happens since $\overline{D}$ cannot have form $\exists r.A$ by Remark 86.

(c) If the last inference ρ_K is of the form

$$(\mathcal{R}_3) \quad \rho_K : \frac{\overline{C} \sqsubseteq \exists r.B, B \sqsubseteq \overline{D}_1}{\overline{C} \sqsubseteq \overline{D}} : \exists r.\overline{D}_1 \sqsubseteq \overline{D} \in \overline{\mathcal{O}}_{CD}$$

Then, since $\exists r.\overline{D}_1 \sqsubseteq \overline{D} \in \overline{\mathcal{O}}_{CD}$, we have $D = \exists r.D_1$ by definition of $\mathcal{O}^-(D)$. Moreover, $\overline{C} \sqsubseteq \exists r.B$ must be derived from a $\mathcal{R}_2$-inference with premise set $\{\overline{C} \sqsubseteq B_1\}$ and side condition $B_1 \sqsubseteq \exists r.B \in \overline{\mathcal{O}}_{CD}$, where B_1 is a concept name (B_1 may be equal to $\overline{C}$). Two cases arise:

i. if $B_1 = \overline{C}_n$, for some subconcept C_n of C. Then since $\overline{C}_n \sqsubseteq \exists r.\overline{C}_m \in \overline{\mathcal{O}}_{CD}$, $B = \overline{C}_m$ for some subconcept C_m of C_n. We have

$$k(\overline{C}_m \sqsubseteq \overline{D}_1) < k(\overline{C} \sqsubseteq \overline{D}) = K.$$

By applying the induction assumption over $\overline{C}_m \sqsubseteq \overline{D}_1$, there exists $T_{C_m \sqsubseteq D_1}$ such that $\overline{\mathcal{U}}(T_{C_m \sqsubseteq D_1}) \models C_m \sqsubseteq D_1$.
Since $\overline{\mathcal{O}}_{CD} \vdash \overline{C} \sqsubseteq \overline{C}_n$ and $\overline{C}_n \sqsubseteq \exists r.\overline{C}_m \in \overline{\mathcal{O}}_{CD}$, C is of the form $\exists r.C_m \sqcap C'$ by definition of $\mathcal{O}^+(C)$. Recall that $D = \exists r.D_1$. Therefore, we have $\{C_m \sqsubseteq D_1\} \models C \sqsubseteq D$, and thus, $\overline{\mathcal{U}}(T_{C_m \sqsubseteq D_1}) \models C \sqsubseteq D$. Finally, $T_{C \sqsubseteq D} = T_{C_m \sqsubseteq D_1}$ is the desired set of trees.

ii. Otherwise, since $\overline{\mathcal{O}}_{CD}$ does not contain axioms of the form $\overline{D}_i \sqsubseteq \exists r_j.\overline{D}_j$, we have that B_1, B are concept names in $\mathit{sig}(\mathcal{O})$.
Note that (recall $\overline{B}_1 = B_1, \overline{B} = B$ by definition of $\bar{*}$)

$$k(\overline{C} \sqsubseteq \overline{B_1}) < K, \;\; k(\overline{B} \sqsubseteq \overline{D}_1) < K.$$

By induction assumption, for $\overline{C} \sqsubseteq \overline{B_1}$, $\overline{B} \sqsubseteq \overline{D}_1$, there exists two sets $T_{C \sqsubseteq B_1}, T_{B \sqsubseteq D_1}$ such that

- $T_{C \sqsubseteq B_1} = \{\mathtt{t}^-\}$ consists of one b*-tree $\mathtt{t}^-$ from $\mathit{sig}(C)$ to B_1 and $\overline{\mathcal{U}}(T_{C \sqsubseteq B_1}) \models C \sqsubseteq B_1$;
- $T_{B \sqsubseteq D_1} = \{\mathtt{t}^+\}$ consists of one f*-tree $\mathtt{t}^+$ from B to $\mathit{sig}(D_1)$, which is a subset of $\mathit{sig}(D)$, and $\overline{\mathcal{U}}(T_{B \sqsubseteq D_1}) \models B \sqsubseteq D_1$;

Let $T_1 = \{\mathtt{t}_1^+\}$, where $\mathtt{t}_1^+$ is the f*-tree from B_1 to $\mathit{sig}(D_1)$ obtained by adding an edge

$$B_1 \xrightarrow{r} B$$

above the root of $\mathtt{t}^+ \in T_{B \sqsubseteq D_1}$. Then, $\overline{\mathcal{U}}(T_1) \models B_1 \sqsubseteq \exists r.D_1$. Since $D = \exists r.D_1$, for $T_{C \sqsubseteq D} = T_{C \sqsubseteq B_1} \cup T_1$, we have $\overline{\mathcal{U}}(T_{C \sqsubseteq D}) \models C \sqsubseteq D$, and thus the set $T_{C \sqsubseteq D}$ is what we want.

Moreover, if $C \in \mathsf{N}_C$ is a concept name, let $\mathtt{t}^-$ be the f*-tree consisting of

i. the edge generated by $\mathcal{O} \models C \sqsubseteq B_1$;
ii. the edge generated by $B_1 \sqsubseteq \exists r.B \in \mathcal{O}$;
iii. edges of the unique f*-tree in $T_{B \sqsubseteq D_1}$

Then $T_{C \sqsubseteq D} = \{\mathtt{t}^-\}$ satisfies all the requirements of the lemma, especially Item 1.

Note that in this case, D can not be a concept name in N_C.

□

By Lemma 89, we have proved Theorem 66 for the case where all the trees are b* or f*-trees. Now, we connect b* or f*-trees with b-trees and f-trees by the following two results:

Lemma 90 *For a b*-tree $\mathtt{t}^-$ from A to Σ, there always exists a set T_b of b-trees from A' to Σ, where $A' = A$ or $A' \in \Sigma$, such that:*

$$\{\mathtt{C}_{\mathtt{t}_\mathtt{i}^-} \sqsubseteq \mathtt{A}_{\mathtt{t}_\mathtt{i}^-} \mid \mathtt{t}_\mathtt{i}^- \in T_b\} \models \mathtt{C}_{\mathtt{t}^-} \sqsubseteq A.$$

Proof. The proof is conducted in two distinct cases.

Case 1 First assume that for each node $\mathtt{n}_0$ in the b*-tree $\mathtt{t}^-$ from A to Σ, if $\mathtt{lab}(\mathtt{n}_0) \in \Sigma$ then $\mathtt{n}_0$ is a leaf. In this case, we prove the lemma by translating $\mathtt{t}^-$ into a b-tree $\mathtt{t}_0^-$ from A to Σ such that $\models \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{C}_{\mathtt{t}_0^-}$. Then, $T_b = \{\mathtt{t}_0^-\}$ satisfies the requirement of the lemma.

Assume that $\mathtt{t}^-$ is not a b-tree. Otherwise nothing need to be proven. Then, there exists a node $\mathtt{n}_1 \in \mathtt{t}^-$, whose label is B_1, such that:

- B_1 is primitive and $B_1 \notin \Sigma$;
- There is one edge $e \in \mathtt{t}^-$ ending with $\mathtt{n}_1$ and having the label $\mathcal{O} \models B_1 \sqsubseteq B$.

Let $\mathtt{n}$ be the parent node of $\mathtt{n}_1$, then $\mathtt{n}$ is labelled by B.

Two cases arise:

1. If $\mathtt{n}_1$ has only one child $\mathtt{n}_2$ and the edge from $\mathtt{n}_1$ to $\mathtt{n}_2$ is labelled by $\mathcal{O} \models B_2 \sqsubseteq B_1$ (see Figure 4.6). Let $\mathtt{t}_1^-$ be the new tree obtained from $\mathtt{t}^-$ by

   - deleting $\mathtt{n}_1$;
   - setting the child of $\mathtt{n}$ as $\mathtt{n}_2$,

   We have $\models \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{C}_{\mathtt{t}_1^-}$;

2. If $\mathtt{n}_1$ has a child $\mathtt{n}_2$ and the edge from $\mathtt{n}_1$ to $\mathtt{n}_2$ is labelled by $C_1 \sqsubseteq B_1 \in \mathcal{O}$ (see Figure 4.7).

   Since B_1 is primitive, we know

   $$C_1 \sqsubseteq B_1,\ B_1 \sqsubseteq C_1 \in \mathcal{O}$$

   are the only two axioms of the form $* \sqsubseteq B_1$ or $B_1 \sqsubseteq *$ in $\mathcal{O}$.

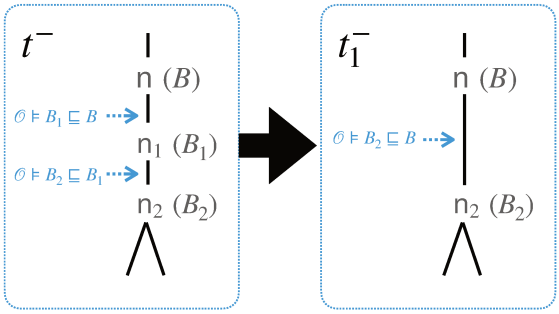


Figure 4.6: Case 1

Since $B_1 \sqsubseteq C_1 \in \mathcal{O}$ is the only axiom in $\mathcal{O}$ of the form $B_1 \sqsubseteq *$, we have

$$\mathcal{O} \models B_1 \sqsubseteq B \text{ iff } \mathcal{O}_1 \models C_1 \sqsubseteq B,$$

where $\mathcal{O}_1 = \mathcal{O} \setminus \{B_1 \sqsubseteq C_1\}$.

Then by Lemma 89, we can find a b*-tree $\mathtt{t}^-_{\mathtt{new}}$ from B to *sig*(C_1) over $\mathcal{O}_1$ such that

$$\{\mathtt{C}_{\mathtt{t}^-_{\mathtt{new}}} \sqsubseteq B\} \models C_1 \sqsubseteq B.$$

Note that $\mathcal{O}_1 \not\models B_1 \sqsubseteq B'$ for any $B' \in \mathsf{N}_C, B' \neq B$ because there are no axioms of the form $B_1 \sqsubseteq *$ in $\mathcal{O}_1$. Therefore, no edge in $\mathtt{t}^-_{\mathtt{new}}$ has the label $\mathcal{O}_1 \models B_1 \sqsubseteq B'$.

Let $\mathtt{t}_1^-$ be the b*-tree obtained from $\mathtt{t}^-$ by

- replacing the edges from $\mathtt{n}$ to $\mathtt{n}_2$ by $\mathtt{t}^-_{\mathtt{new}}$;

We have $\models \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{C}_{\mathtt{t}_1^-}$.

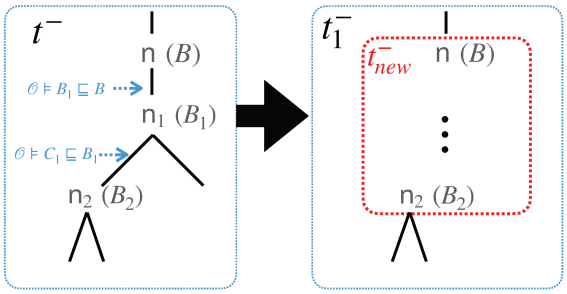


Figure 4.7: Case 2

We repeat the above process exhaustively until the resulting new tree, denoted by $\mathtt{t}_0^-$, is a b-tree. Then we have $\models \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{C}_{\mathtt{t}_0^-}$. Therefore, $\{\mathtt{C}_{\mathtt{t}_0^-} \sqsubseteq A\} \models \mathtt{C}_{\mathtt{t}^-} \sqsubseteq A$, and thus $T_b = \{\mathtt{t}_0^-\}$ satisfies the requirement of the lemma.

Case 2 There exists a node $\mathtt{n}_0$ in $\mathtt{t}^-$ such that $\mathtt{n}_0$ is not a leaf and $\mathtt{lab}(\mathtt{n}_0) \in \Sigma$. Let S_0 be the collection of all such nodes, then $\mathtt{n}_0 \in S_0$. We split $\mathtt{t}^-$ into several b*-trees $\mathtt{t}_1^-, \ldots, \mathtt{t}_\mathtt{n}^-$ by cutting $\mathtt{t}^-$ at all nodes in S_0. For an example see Figure 4.8. Then, $\mathtt{t}_1^-, \ldots, \mathtt{t}_\mathtt{n}^-$ satisfies the requirement of Case 1, and thus there exists b-trees

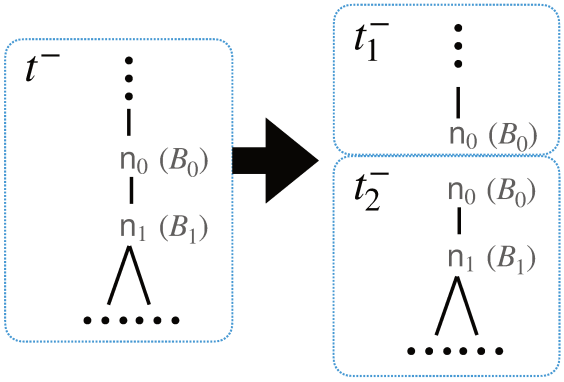


Figure 4.8: Split $\mathtt{t}^-$ at node $\mathtt{n}_0$.

$\mathtt{t}_{1,0}^-, \ldots, \mathtt{t}_{\mathtt{n},0}^-$ such that

$$\{\mathtt{C}_{\mathtt{t}_{i,0}^-} \sqsubseteq \mathtt{A}_{\mathtt{t}_{i,0}^-}\} \models \mathtt{C}_{\mathtt{t}_i^-} \sqsubseteq \mathtt{A}_{\mathtt{t}_i^-},\ 1 \leq i \leq n.$$

Therefore, $T_b = \{\mathtt{t}_{1,0}^-, \ldots, \mathtt{t}_{\mathtt{n},0}^-\}$ satisfies the requirement of the lemma. This concludes our proof. □

For f*-trees, we have a similar result stated below:

Lemma 91 *For a f*-tree $\mathtt{t}^+$ from A to Σ, there always exists a set T_f of f-trees from A' to Σ, where $A' = A$ or $A' \in \Sigma$, such that:*

$$\{\mathtt{A}_{\mathtt{t}_i^+} \sqsubseteq \mathtt{C}_{\mathtt{t}_i^+} \mid \mathtt{t}_i^+ \in T_f\} \models A \sqsubseteq \mathtt{C}_{\mathtt{t}^+}.$$

Proof. The proof consists of two cases.

Case 1 First assume that for each node $\mathtt{n}_0$ in the f*-tree $\mathtt{t}^+$ from A to Σ, if $\mathtt{lab}(\mathtt{n}_0) \in \Sigma$ then $\mathtt{n}_0$ is a leaf. In this case, we prove the lemma by translating $\mathtt{t}^+$ to a f-tree $\mathtt{t}_0^+$ such that $\models \mathtt{C}_{\mathtt{t}_0^+} \sqsubseteq \mathtt{C}_{\mathtt{t}^+}$. Then, $T_f = \{\mathtt{t}_0^+\}$ satisfies the requirement of the lemma.

Assume that $\mathtt{t}^+$ is not a f-tree. Then, there exists a node $\mathtt{n}_1 \in \mathtt{t}^+$, whose label is B_1, such that:

- B_1 is primitive and $B_1 \notin \Sigma$;
- There is one edge $e \in \mathtt{t}^+$ ending with $\mathtt{n}_1$ has the label $\mathcal{O} \models B \sqsubseteq B_1$.

Let $\mathtt{n}$ be the parent node of $\mathtt{n}_1$, then $\mathtt{n}$ has the label B. There are two different cases:

1. If $\mathtt{n}_1$ has only one child $\mathtt{n}_2$, and the edge from $\mathtt{n}_1$ to $\mathtt{n}_2$ is labelled by $\mathcal{O} \models B_1 \sqsubseteq B_2$ (see Figure 4.9), then we can build a new tree $\mathtt{t}_1^-$ by removing $\mathtt{n}_1$, set $\mathtt{n}_2$ as the child of $\mathtt{n}$. Then, $\models \mathtt{C}_{\mathtt{t}_1^-} \sqsubseteq \mathtt{C}_{\mathtt{t}^-}$;

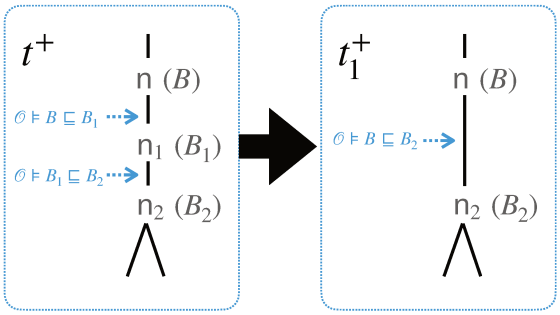


Figure 4.9: Point 1 of Case 1

2. If $\mathtt{n}_1$ has a child $\mathtt{n}_2$, and the edge from $\mathtt{n}_1$ to $\mathtt{n}_2$ has the label $B_1 \sqsubseteq \exists r.B_2 \in \mathcal{O}$ (see Figure 4.10). Since B_1 is primitive, we know that

$$\exists r.B_2 \sqsubseteq B_1,\ B_1 \sqsubseteq \exists r.B_2 \in \mathcal{O}$$

are the only two axioms of the form $* \sqsubseteq B_1$ or $B_1 \sqsubseteq *$ in $\mathcal{O}$.

Since $\exists r.B_2 \sqsubseteq B_1 \in \mathcal{O}$ is the only axiom in $\mathcal{O}$ of the form $* \sqsubseteq B_1$, for $\mathcal{O}_1 = \mathcal{O} \setminus \{\exists r.B_2 \sqsubseteq B_1\}$, we have

$$\mathcal{O} \models B \sqsubseteq B_1 \text{ iff } \mathcal{O}_1 \models B \sqsubseteq \exists r.B_2.$$

Then by Lemma 89, we can find a f*-tree $\mathtt{t}^+_{\mathtt{new}}$ from B to a set of the form $\{B_2, \ldots\}$ over $\mathcal{O}_1$ such that

$$\{B \sqsubseteq \mathtt{C}_{\mathtt{t}^+_{\mathtt{new}}}\} \models B \sqsubseteq \exists r.B_2.$$

Note that $\mathcal{O}_1 \not\models B' \sqsubseteq B_1$ for any $B' \in \mathsf{N}_C, B' \neq B$ because there is no axiom of the form $* \sqsubseteq B_1$ in $\mathcal{O}_1$. Therefore, no edge in $\mathtt{t}^+_{\mathtt{new}}$ has the label $\mathcal{O} \models B' \sqsubseteq B_1$.

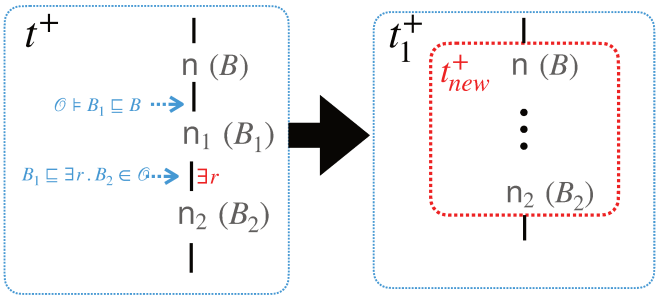


Figure 4.10: Point 2 of Case 1

Let $\mathtt{t}^+_1$ be the f*-tree obtained by replacing the edges from $\mathtt{n}_1$ to $\mathtt{n}_2$ and $\mathtt{n}_1$ to $\mathtt{n}_2$ by $\mathtt{t}^+_{\mathtt{new}}$. Then, we have $\models \mathtt{C}_{\mathtt{t}^-_1} \sqsubseteq \mathtt{C}_{\mathtt{t}^-}$.

We repeat the above process exhaustively until the resulting new tree, denoted by $\mathtt{t}^+_0$, is a f-tree. Then we have $\models \mathtt{C}_{\mathtt{t}^+_0} \sqsubseteq \mathtt{C}_{\mathtt{t}^+}$. Therefore, we have $\{A \sqsubseteq \mathtt{C}_{\mathtt{t}^+_0}\} \models A \sqsubseteq \mathtt{C}_{\mathtt{t}^-}$, and thus $T_f = \{\mathtt{t}^+_0\}$ satisfies the requirement of the lemma.

Case 2 There exists node $\mathtt{n}_0$ in $\mathtt{t}^+$ such that $\mathtt{n}_0$ is not a leaf and $\mathtt{lab}(\mathtt{n}_0) \in \Sigma$. Let S_0 be the collection of all such nodes, then $\mathtt{n}_0 \in S_0$. We split $\mathtt{t}^+$ into several f*-trees $\mathtt{t}_1^+, \ldots, \mathtt{t}_\mathtt{n}^+$ by cutting $\mathtt{t}^+$ at all nodes in S_0. Then, we can apply arguments on Case 1 for $\mathtt{t}_1^+, \ldots, \mathtt{t}_\mathtt{n}^+$ and find f-trees $\mathtt{t}_{1,0}^+, \ldots, \mathtt{t}_{\mathtt{n},0}^+$ such that

$$\{\mathtt{A}_{\mathtt{t}_{i,0}^+} \sqsubseteq \mathtt{C}_{\mathtt{t}_{i,0}^+}\} \models \mathtt{A}_{\mathtt{t}_i^+} \sqsubseteq \mathtt{C}_{\mathtt{t}_i^+},\ 1 \leq i \leq n.$$

Therefore, $T_f = \{\mathtt{t}_{1,0}^+, \ldots, \mathtt{t}_{\mathtt{n},0}^+\}$ satisfies the lemma. This concludes our proof. □

Finally, Theorem 66 is a direct corollary of Lemma 89, 90 and 91.

Proof of Theorem 72

Theorem 92 *A pseudo-minimal module for $\mathcal{O}$ and Σ is a deductive module for $\mathcal{O}$ and Σ. Moreover, if $F_\Sigma^\mathcal{O}$ is finite, then $\mathcal{M}$ is a pseudo-minimal module for $\mathcal{O}$ and Σ iff $\mathcal{M}$ is a minimal module for $\mathcal{O}$ and Σ.*

Proof. Recall the definition of $F_1^*,\ F_2,\ F_3$ in Section 4.3.2 and

$$\mathcal{S}_\Sigma^\mathcal{O} = \{ \bigcup_{\mathtt{t} \in F_1^* \cup F_2 \cup F'} S_\mathtt{t} \;\Big|\; S_\mathtt{t} \in \mathtt{Supp}(\mathtt{t}),\ F' \subseteq F_3,\ \mathcal{U}(F_2 \cup F') \models \mathcal{U}(F_2 \cup F_3)\}.$$

According to the definition of $\mathcal{S}_\Sigma^\mathcal{O}$ and $\mathcal{U}_\Sigma(\mathcal{O})$, every ontology $\mathcal{O}_1 \in \mathcal{S}_\Sigma^\mathcal{O}$ satisfies $\mathcal{U}_\Sigma(\mathcal{O}_1) \models \mathcal{U}_\Sigma(\mathcal{O})$. Therefore, by Theorem 66, all ontologies in $\mathcal{S}_\Sigma^\mathcal{O}$ are deductive modules for $\mathcal{O}$ and Σ. Hence, we can infer that pseudo-minimal modules are also deductive modules.

If $F_\Sigma^\mathcal{O}$ is finite, then we have $F_1^* \cup F_2 = \emptyset$ and $F_\Sigma^\mathcal{O} = F_3$. Next, we show that if $\mathcal{M}$ is a minimal module for $\mathcal{O}$ and Σ, then $\mathcal{M} \in \mathcal{S}_\Sigma^\mathcal{O}$.

Since $\mathcal{M} \subseteq \mathcal{O}$ is a minimal module, we have $F_\Sigma^\mathcal{M} \subseteq F_\Sigma^\mathcal{O}$ and $\mathcal{U}(F_\Sigma^\mathcal{M}) \models \mathcal{U}(F_\Sigma^\mathcal{O})$ by Theorem 66. Therefore, if $F'' \subseteq F_\Sigma^\mathcal{M}$ and $\mathcal{U}(F'') \models \mathcal{U}(F_\Sigma^\mathcal{M})$, then $\mathcal{U}(F'') \models \mathcal{U}(F_\Sigma^\mathcal{O})$. By definition of $\mathcal{S}_\Sigma^\mathcal{M}$ and $\mathcal{S}_\Sigma^\mathcal{O}$, we have $\mathcal{S}_\Sigma^\mathcal{M} \subseteq \mathcal{S}_\Sigma^\mathcal{O}$. Since $\mathcal{M} \subseteq \mathcal{O}$ is a minimal module, we have $\mathcal{S}_\Sigma^\mathcal{M} = \{\mathcal{M}\}$ and thus $\mathcal{M} \in \mathcal{S}_\Sigma^\mathcal{O}$.

In conclusion, we have all minimal ontologies in $\mathcal{S}_\Sigma^\mathcal{O}$ are identical to all minimal modules for $\mathcal{O}$ and Σ. This concludes the theorem. □

Proof of Corollary 79

Corollary 93 *Theorems 66 and 72 still hold if we ignore the f-trees $\mathtt{t}^+ \in F_\Sigma^\mathcal{O}$ that satisfy one of the following conditions:*

1. *t^+ has an edge, starting from the root $\mathtt{root}(\mathtt{t}^+)$, which is not a $\exists r$-edge;*
2. *$\mathtt{t}^+$ is a proper sub-tree of a f-tree $\mathtt{t}_1^+ \in F_\Sigma^\mathcal{O}$ such that $\mathtt{root}(\mathtt{t}) = \mathtt{root}(\mathtt{t}_1^+)$ and $\not\models \mathtt{C}_{\mathtt{t}^+} \equiv \mathtt{C}_{\mathtt{t}_1^+}$.*

Proof. It is enough to prove that Theorem 66 still holds when we remove all the f-trees satisfying conditions 1 or 2. Then, Theorem 72 is implied by the original proof based on Theorem 66.

Let's call a f-tree satisfying conditions 1 or 2 a *redundant f-tree*. We claim that

Claim 94 *If S is a set of redundant f-trees, if $\mathtt{t}^+$ is a new redundant f-tree, and if $\alpha \in \mathcal{U}(F_\Sigma^\mathcal{O} \setminus S)$, then we have $\mathcal{U}\big(F_\Sigma^\mathcal{O} \setminus (S \cup \{\mathtt{t}^+\})\big) \models \alpha$.*

The proof of the claim is provided below. By this claim, we can remove all redundant f-trees from $F_\Sigma^{\mathcal{O}}$ iteratively, where each step preserves the conclusion of Theorem 66. This concludes the proof of the corollary. □

proof of Claim 94. Two cases arise depending on the type of $\mathtt{t}^+$.

Case 1 Assume that $\mathtt{t}^+ \in F_\Sigma^{\mathcal{O}}$ is a redundant f-tree with an edge, which is not a $\exists r$-edge, starting from the root $\mathtt{root}(\mathtt{t}^+)$. There are two cases:

- $\mathtt{A}_{\mathtt{t}^+} \in \Sigma$. Next, we show that the axiom

$$\alpha := \mathtt{A}_{\mathtt{t}^+} \sqsubseteq \mathtt{D}_{\mathtt{t}^+} \in \mathcal{U}(F_\Sigma^{\mathcal{O}} \setminus S)$$

  induced by $\mathtt{t}^+$ is inferred by $\mathcal{U}(F_\Sigma^{\mathcal{O}} \setminus S \cup \{\mathtt{t}^+\})$.
  The idea is illustrated by Figure 4.11. Assume $\mathtt{t}^+$ has the form shown in Figure 4.11. Then, we have

$$\mathtt{A}_{\mathtt{t}^+} \sqsubseteq \mathtt{D}_{\mathtt{t}^+} = A \sqsubseteq \exists r.(A_1 \sqcap A_2) \sqcap \exists r.A_3.$$

  We could split the f-tree $\mathtt{t}^+$ of the figure into two different f-trees $\mathtt{t}_1^+, \mathtt{t}_2^+$ and a b-tree $\mathtt{t}_1^-$, where all edges in $\mathtt{t}_\mathtt{i}^+$ $(\mathtt{i} = 1, 2)$ starting from the root $\mathtt{root}(\mathtt{t}_\mathtt{i}^+)$ are $\exists r$-edges. Then, the axiom $A \sqsubseteq \exists r.(A_1 \sqcap A_2) \sqcap \exists r.A_3$ is inferred by

$$\{\alpha_1 : A \sqsubseteq \exists r.(A_1 \sqcap A_2),$$
$$\alpha_2 : A \sqsubseteq \exists r.A_3\},$$

  where $\alpha_1 \in \mathcal{U}(\{\mathtt{t}_1^+, \mathtt{t}_1^-\})$ and $\alpha_2 \in \mathcal{U}(\{\mathtt{t}_2^+\})$. Therefore, we have $\mathcal{U}\big(F_\Sigma^{\mathcal{O}} \setminus (S \cup \{\mathtt{t}^+\})\big) \models \alpha$.
  The general case is proved in the same way. In details, we can always split $\mathtt{t}^+$ into $\mathtt{t}_1^+, \ldots, \mathtt{t}_\mathtt{m}^+$ and b-trees $\mathtt{t}_1^-, \ldots, \mathtt{t}_\mathtt{n}^-$, where $n \leq m$, such that

  - All edges in $\mathtt{t}_1^+, \ldots, \mathtt{t}_\mathtt{m}^+$ starting from the root are $\exists r$-edges;
  - $\mathtt{t}_1^-, \ldots, \mathtt{t}_\mathtt{n}^-$ consist of a single edge labelled by $\mathcal{O} \models \mathtt{A}_{\mathtt{t}^+} \sqsubseteq B$ for some $B \in \mathsf{N}_C$ (e.g., $\mathtt{t}_1^-$ in Figure 4.11);
  - For $1 \leq i \leq n$, roots of $\mathtt{t}_\mathtt{i}^+, \mathtt{t}_\mathtt{i}^-$ have the same label (e.g., $\mathtt{t}_1^+, \mathtt{t}_1^-$ in Figure 4.11);
  - $\mathcal{U}(\{\mathtt{t}_\mathtt{i}^+, \mathtt{t}_\mathtt{i}^- \mid 1 \leq i \leq n\}) \cup \mathcal{U}(\{\mathtt{t}_\mathtt{j}^+ \mid n < j \leq m\}) \models \mathtt{A}_{\mathtt{t}^+} \sqsubseteq \mathtt{D}_{\mathtt{t}^+}$ (e.g., $\mathcal{U}(\{\mathtt{t}_1^+, \mathtt{t}_1^-\}) \cup \mathcal{U}(\{\mathtt{t}_2^+\}) \models A \sqsubseteq \exists r.(A_1 \sqcap A_2) \sqcap \exists r.A_3$ at above).

- $\mathtt{A}_{\mathtt{t}^+} \notin \Sigma$. In this case, all axioms induced from $\mathtt{t}^+$ are of the form

$$\alpha := \mathtt{C}_{\mathtt{t}^-} \sqsubseteq \mathtt{D}_{\mathtt{t}^+} \in \mathcal{U}(F_\Sigma^{\mathcal{O}} \setminus S)$$

  where $\mathtt{t}^-$is a b-tree over $\mathcal{O}$ with $\mathtt{A}_{\mathtt{t}^-} = \mathtt{A}_{\mathtt{t}^+}$. By the same argument as in the previous case, we have α is inferred from $\mathcal{U}\big(F_\Sigma^{\mathcal{O}} \setminus (S \cup \{\mathtt{t}^+\})\big)$. The only difference is that now $\mathtt{t}^-$ is involved when looking for axioms in $\mathcal{U}\big(F_\Sigma^{\mathcal{O}} \setminus (S \cup \{\mathtt{t}^+\})\big)$ that derive α. For instance, for the example of Figure 4.11, we have

  - α_1 has the form $\mathtt{C}_{\mathtt{t}^-} \sqsubseteq \exists r.(A_1 \sqcap A_2)$ and is induced by the pair $(\mathtt{t}_1^+, \mathtt{t}^- \cup \mathtt{t}_1^-)$. That is, $\alpha_1 \in \mathcal{U}(\{\mathtt{t}_1^+, \mathtt{t}^- \cup \mathtt{t}_1^-\})$;
  - α_2 has the form $\mathtt{C}_{\mathtt{t}^-} \sqsubseteq \exists r.A_3$ and is induced by the pair $(\mathtt{t}_2^+, \mathtt{t}^-)$. That is, $\alpha_2 \in \mathcal{U}(\{\mathtt{t}_2^+, \mathtt{t}^-\})$.

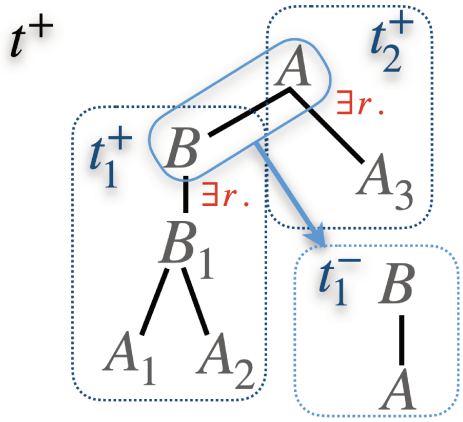


Figure 4.11: An example

Case 2 Assume that f-tree $\mathtt{t}^+ \in F_\Sigma^\mathcal{O}$ is a proper sub-tree of a f-tree $\mathtt{t}_1^+ \in F_\Sigma^\mathcal{O}$ such that $\mathtt{A}_{\mathtt{t}_1} = \mathtt{A}_{\mathtt{t}_2}, \not\models \mathtt{C}_{\mathtt{t}^+} \equiv \mathtt{C}_{\mathtt{t}_1^+}$. Then all the axioms $\alpha \in \mathcal{U}(F_\Sigma^\mathcal{O} \setminus S)$ induced by $\mathtt{t}^+$ are inferred by the an axiom $\alpha_1 \in \mathcal{U}\big(F_\Sigma^\mathcal{O} \setminus (S \cup \{\mathtt{t}^+\})\big)$, which is induced in the same way as α but with $\mathtt{t}^+$ replaced by $\mathtt{t}_1^+$.

The two cases above conclude the proof. □

Proof of Proposition 69

Proposition 95 *Assume that $\mathcal{O}$ is an acyclic $\mathcal{EL}$-terminology and Σ is a signature. Let $F \subseteq F_\Sigma^\mathcal{O}$ be the subset consisting of all b-trees and maximal f-trees. Then, $\mathrm{NUI}(\mathcal{O}, \Sigma)$ is equivalent to $\mathcal{U}_1(F) \cup \mathcal{U}_2(F)$ (see Definition 65).*

Proof. Let $conj(C) = \{C_1, \ldots, C_n\}$ for each concept $C = C_1 \sqcap \ldots \sqcap C_n$. Then, by Definitions 58, 60, 64 and 68, we know that

$$P_\Sigma(A) = \{\mathtt{C}_{\mathtt{t}^-} \mid \mathtt{t}^- \in F_\Sigma^\mathcal{O},\ \mathtt{A}_{\mathtt{t}^-} = A\}, \tag{4.3}$$

$$Q_\Sigma(A) = \bigcup_{\mathtt{t}^+ \in F_\Sigma^\mathcal{O}, \mathtt{A}_{\mathtt{t}^+} = A} conj(\mathtt{C}_{\mathtt{t}^-}). \tag{4.4}$$

Recall that the uniform interpolant computed by `NUI` has the form:

$$\begin{aligned}\mathcal{U} = &\{C \sqsubseteq A \mid A \in \Sigma,\ C \in P_\Sigma(A)\} \\ &\cup \{A \sqsubseteq C \mid A \in \Sigma,\ C \in Q_\Sigma(A)\},\end{aligned}$$

Therefore, the uniform interpolant $\mathcal{U}$ is equivalent to $\mathcal{U}^1(F) \cup \mathcal{U}^2(F)$ in the sense that $\mathcal{U}$ is obtained from $\mathcal{U}^1(F) \cup \mathcal{U}^2(F)$ by rewriting axioms $A \sqsubseteq C \in \mathcal{U}^1(F) \cup \mathcal{U}^2(F)$ to $A \sqsubseteq C_1, \ldots, A \sqsubseteq C_n$, where $C = C_1 \sqcap \ldots \sqcap C_n$.

□

5 - Computing General Modules

In this chapter, we present a method of computing *general modules* for $\mathcal{ALC}$, which was first investigated for the lightweight DLs $\mathcal{EL}$ [58]. A general module for an ontology is an ideally substantially smaller ontology that preserves all entailments for a user-specified set of terms. As such, it has applications such as ontology reuse and ontology analysis. Different from classical modules, general modules may use axioms not explicitly present in the input ontology, which allows for additional conciseness. General modules generalize both uniform interpolants and deductive modules, where the former requires all names in the result to be in the given signature, and the latter requires all axioms in the final result to be in the original ontology. This makes them useful in their own right: as our evaluation shows, our general modules are often smaller than both locality-based modules and uniform interpolants.

The main steps of our approach are shown in Figure 5.1. Our method is based on a normalized version of the input ontology (Section 5.2). Note that in this chapter and more precisely in Section 5.2, we are focussing on a notion of normalization for $\mathcal{ALC}$-ontologies that has been developed in [46] for computing uniform interpolant and is different from the one introduced in definition 2.7 of the preliminary section. Then, the general module building process relies on an efficient role names elimination (Section 5.3) and concept names eliminations (Section 5.4), which include the elimination of special concept names, called *definer names*, that are introduced during normalization. The use of definer names is inspired by the method for uniform interpolation in [47]. However, since general modules allow names outside the given signature in contrast to uniform interpolants, we are able to develop a more efficient definer elimination step. A detailed comparative discussion of our method and the method for computing uniform interpolant is postponed to Section 5.1.

Moreover, our method is able to compute both deductive modules and uniform interpolants (Section 5.5). Our evaluation (Section 5.6) shows that all our methods, including the one for uniform interpolation, can compete with the run times of locality-based module extraction, while at the same time resulting in substantially smaller ontologies.

This chapter is organized as follows. First, we discuss some related works in Section 5.1. Then, Section 5.2 shows how to normalize $\mathcal{ALC}$ ontologies. In Section 5.3, we develop a role forgetting method by introducing *role isolated ontologies*. Our method for computing general modules, based on the role forgetting result, is presented in Section 5.4. Furthermore, we extend our method to computing both deductive modules and uniform interpolants in Section 5.5. Finally, we present the result of the evaluation of our methods on real-world ontologies in Section 5.6. Proof of all the results of this chapter could be found in Section 5.7.

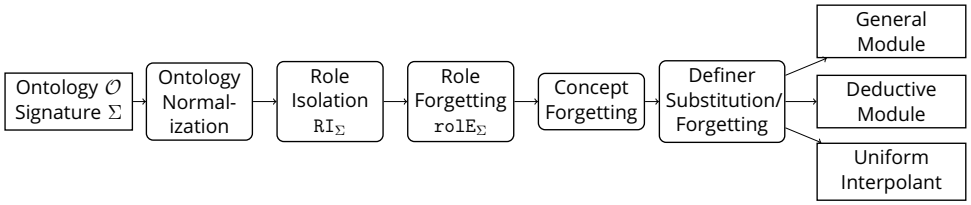


Figure 5.1: Overview of our unified method for computing general modules, deductive modules, and uniform interpolants

5.1 . Related Work

A detailed discussion on computing deductive modules for $\mathcal{ALC}$ and $\mathcal{EL}$ ontologies has been provided in Section 4.1. Next, we discuss related works on computing uniform interpolants for $\mathcal{ALC}$ ontologies.

Let us start with a brief introduction of LETHE [47], a method that computes uniform interpolants. For a given $\mathcal{ALC}$-ontology $\mathcal{O}$ and signature Σ, LETHE computes a uniform interpolant for $\mathcal{O}$ and Σ by mainly three steps:

1. **Transformation into $\mathcal{ALC}$ normal form**

   During this step, $\mathcal{O}$ is translated into the *normal form* $cl(\mathcal{O})$, whose axioms are of the form:

   $$\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n \qquad L_i ::= A \mid \neg A \mid \mathsf{Q}r.D,$$

   where $A \in \mathsf{N}_C$, $\mathsf{Q} \in \{\forall, \exists\}$ and $D \in \mathsf{N}_C \setminus \mathit{sig}(\mathcal{O})$ is a newly introduced concept names called *definers*. Details of how to construct $cl(\mathcal{O})$ are postponed to Section 5.2. For simplicity, we omit "$\top \sqsubseteq$" in the left-hand-side of all axioms occurring in $cl(\mathcal{O})$ or the following computations;

2. **Resolution over $cl(\mathcal{O})$**

   Next, LETHE recursively applies the resolution rules in Figure 5.9 over $cl(\mathcal{O})$. Here, the rule **A-Res** applies only for $A \notin \Sigma$. The rule **r-Res**, which determines whether $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ relying on an external reasoner, applies only for $r \notin \Sigma$. The rule **r-Prop** introduces a new definer D_{12}, which refers to a new concept name representing $D_1 \sqcap D_2$. If D_{12} was not introduced before, the two new axioms $\neg D_{12} \sqcup D_1,\ \neg D_{12} \sqcup D_2$ are also added to the resolution result.

3. **Definer Elimination**

   Finally, all definers D are eliminated by the following two steps:

   (a) First, for each definer D, the set S, which consists of all axioms containing $\neg D$, is replaced by a single axiom $D \sqsubseteq C_1 \sqcap C_2 \sqcap \ldots \sqcap C_n$.

   $$S = \{\neg D \sqcup C_1, \neg D \sqcup C_2, \ldots, \neg D \sqcup C_n\};$$

$$\textbf{A-Res}: \frac{C_1 \sqcup A,\ C_2 \sqcup \neg A}{C_1 \sqcup C_2}$$

$$\textbf{r-Prop}: \frac{C_1 \sqcup \forall r.D_1,\ C_2 \sqcup Qr.D_2}{C_1 \sqcup C_2 \sqcup Qr.D_{12}}$$

$$\textbf{r-Res}: \frac{C_1 \sqcup \exists r.D_1,\ C_2 \sqcup \forall r.D_2,\ \ldots, C_n \sqcup \forall r.D_n}{C_1 \sqcup \ldots \sqcup C_n},$$
$$\text{where } cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot, n \geq 1.$$

Figure 5.2: Lethe resolution rules

(b) Then, the following substitution process is repeated exhaustively: For each axiom of the form $D \sqsubseteq C$, this axiom is removed and all occurrences of D are replaced by C in the remaining axioms.

Step (a) may generate axioms of the form $D \sqsubseteq C$ with $D \in \mathit{sig}(C)$. In this case, one can not remove $D \sqsubseteq C$ directly as it may lead to the loss of some logical information over Σ. This problem can be solved by using the *greatest fixpoint* proposed in [12]. More details are given in [46].

Example 96 *Assume that $\mathcal{O} = \{A \sqsubseteq \forall r.B, B \sqsubseteq A_1\}$ and $\Sigma = \{A, A_1, r\}$. Then a uniform interpolant for $\mathcal{O}$ and Σ is obtained by LETHE as follows:*

1. *$\mathcal{O}$ is translated into the $\mathcal{ALC}$ normal form*

$$cl(\mathcal{O}) = \{\ \neg A \sqcup \forall r.D,\ \neg D \sqcup B,\ \neg B \sqcup A_1\ \},$$

where D is a definer;

2. *By applying the resolution rules in Figure 5.9 over $cl(\mathcal{O})$, one obtains:*

$$\{\ \neg A \sqcup \forall r.D,\ \neg D \sqcup A_1\};$$

3. *The elimination of definers takes two steps:*

   (a) *for the unique definer D, we have $S = \{\neg D \sqcup A_1\}$. Replacing S by $D \sqsubseteq A_1$, one obtains*

$$\{\ \neg A \sqcup \forall r.D,\ D \sqsubseteq A_1\};$$

   (b) *There is only one axiom $D \sqsubseteq A_1$ satisfying the requirement. By removing this axiom and replacing all occurrences of D by A_1, one obtains*

$$\mathcal{U} = \{\ \neg A \sqcup \forall r.A_1\ \} = \{\ A \sqsubseteq \forall r.A_1\ \}.$$

Finally, the resulting ontology $\mathcal{U}$ is a uniform interpolant for $\mathcal{O}$ and Σ.

For computing uniform interpolant more efficiently for $\mathcal{ALC}$-ontologies, [76] proposed to split the computation of uniform interpolant into two steps (i) Forgetting role names outside the given signature; (ii) Forgetting concept names outside the given signature. These two steps have also been called *role forgetting* and *concept forgetting* [74]. For each step, [76] proposed a different set of rules. Experiments on real-world ontologies show that such an approach is more efficient than LETHE [76]. However, as for LETHE, [76] relies on an external reasoner (for the role forgetting step) and may introduce new definers during the computation progress (during the concept forgetting step). There are also other methods, such as FAME [78], FAME(Q) [79] and the invariant of LETHE proposed in [48], that consider the *universal role* ∇ and compute uniform interpolants under $\mathcal{ALC}^{\nabla}$ or more general semantics (e.g., $\mathcal{ALCOQH}(\nabla)$). The universal role ∇ is a notion analog to $\top$, but for role names (i.e., ∇ is "containing" all role names). Indeed, FAME and FAME(Q) have a similar mechanism as [76] and compute uniform interpolants by forgetting concept and role names separately. As stated in [48], introducing universal roles could accelerate the computation of uniform interpolants by simplifying the rules for forgetting role names. However, there are two disadvantages to use the universal role: First, one would produce axioms outside of $\mathcal{ALC}$ (i.e., axioms containing ∇). Second, the resulting uniform interpolant under $\mathcal{ALC}^{\nabla}$-semantic are usually more complicated than uniform interpolant under $\mathcal{ALC}$-semantic. A simple example is shown below.

Example 97 *Consider the ontology $\mathcal{O} = \{A \sqsubseteq \exists r.B\}$ and the signature $\Sigma = \{A, B\}$, then the uniform interplant for $\mathcal{O}$ and Σ under $\mathcal{ALC}^{\nabla}$-semantic is $\{A \sqsubseteq \exists \nabla.B\}$. In constrast, the uniform interplant for $\mathcal{O}$ and Σ under $\mathcal{ALC}$-semantic is an empty set.*

Our method can be regarded as a combination of [76] and LETHE. As shown in Figure 5.1, our method computes general modules for a given signature Σ by four steps:(i) normalising the given $\mathcal{ALC}$ ontology; (ii) forgetting role names outside Σ; (iii) forgetting concept names outside Σ; (iv) eliminating definer names. However, our method changes the mechanism of the method proposed in [76] and LETHE in the following four aspects:

1. Differently from LETHE (r-Prop rule) or [76], in our method, normalization is the only step that introduces definers;

2. We introduce a new efficient role name elimination process based on *role isolation*, which does not rely on an external reasoner nor on the universal role;

3. Our concept name elimination process only depends on a single resolution rule (i.e., *A-Rule* in Figure 5.4), which does not introduce new definers. In

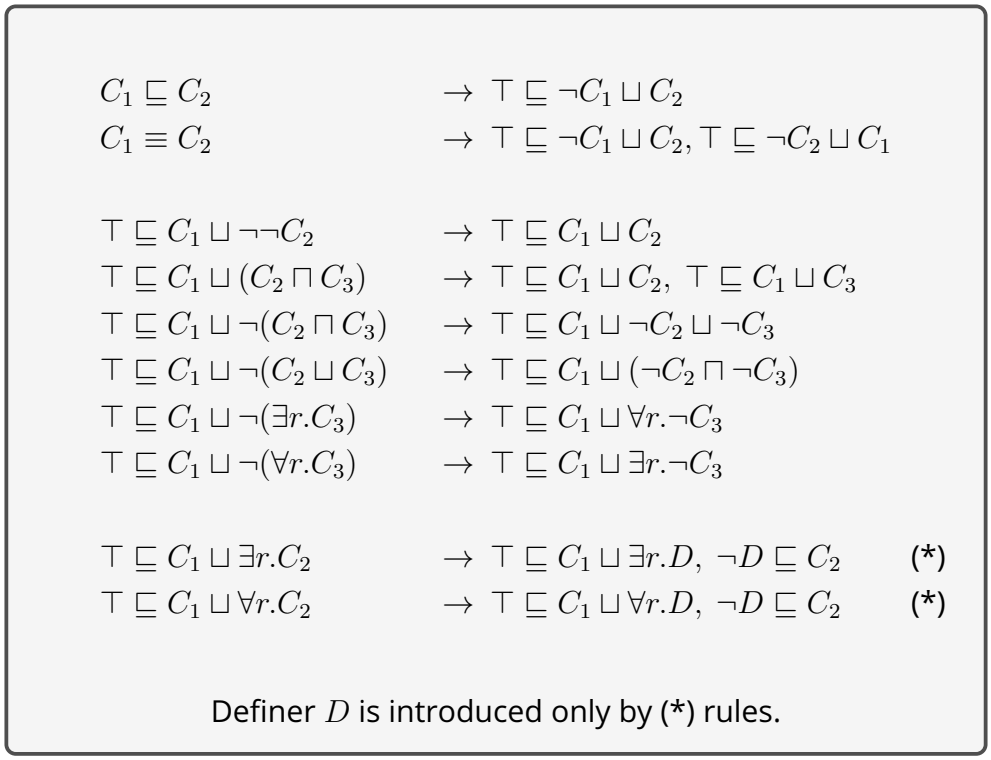


Figure 5.3: Rules for transforming any $\mathcal{ALC}$ ontology into normal form

contrast, in [76], the concept forgetting rules are much more complicated and may introduce new definers;

4. Our definer elimination step may reintroduce names outside the given signature, which is not the case for LETHE.

5.2 . Ontology Normalization

Our method performs forgetting on a normalized view of the ontology, which is obtained via the introduction of fresh names as in [46]. Next, we say an ontology $\mathcal{O}$ is in *normal form* if every axiom is of the following form:

$$\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n \qquad L_i ::= A \mid \neg A \mid \mathsf{Q}r.A,$$

where $A \in \mathsf{N}_C$, and $\mathsf{Q} \in \{\forall, \exists\}$. We call the disjuncts L_i *literals*. For simplicity, we omit the "$\top \sqsubseteq$" on the left-hand side of normalized axioms, which are regarded as *sets*, in order to avoid dealing with duplicated literals and order. As an example, the axiom $A_2 \sqsubseteq A_3 \sqcup \forall s.B_3$ is equivalent to $\neg A_2 \sqcup A_3 \sqcup \forall s.B_3$ in normal form.

As introduced in [46], an $\mathcal{ALC}$-ontology $\mathcal{O}$ can be translated into a *normal form* $cl(\mathcal{O})$ using the rules in Figure 5.3. In particular, concepts C occurring

under role restrictions $\mathsf{Q}r.C$ are replaced by fresh names D taken from a set $\mathsf{N_D} \subseteq \mathsf{N_C} \setminus \mathit{sig}_{\mathsf{C}}(\mathcal{O})$ of *definers*. We use D, D', D_1, D_2, ... to denote definers.

Example 98 *An axiom $A_1 \sqsubseteq \exists r.\exists s.B_1 \sqcup \exists r.B_2$ is translated into a normal form by the following steps:*

1. *First, we translate the axiom into the following equivalent form:*

$$(\top \sqsubseteq) \neg A_1 \sqcup \exists r.\exists s.B_1 \sqcup \exists r.B_2;$$

2. *Next, we replace $\mathcal{ALC}$-concepts under a role restriction by definers.*

$$\begin{array}{lll} \textit{adding } D_1: & \neg A_1 \sqcup \exists r.\boldsymbol{D_1} \sqcup \exists r.B_2, & \neg \boldsymbol{D_1} \sqcup \exists s.B_1 \\ \textit{adding } D_2: & \neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.B_2, & \neg D_1 \sqcup \exists s.\boldsymbol{D_2}, \\ & \neg \boldsymbol{D_2} \sqcup B_1 & \\ \textit{adding } D_3: & \neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.\boldsymbol{D_3}, & \neg D_1 \sqcup \exists s.D_2, \\ & \neg D_2 \sqcup B_1, & \neg \boldsymbol{D_3} \sqcup B_2 \end{array}$$

Here, the definers D_1, D_2, D_3 stand for the concepts $C_{D_1} = \exists s.B_1$, $C_{D_2} = B_1$, $C_{D_3} = B_2$, respectively.

For each introduced definer D, we remember the concept C_D that it replaced. We assume that distinct occurrences of the same concept are replaced by distinct definers. Thus, in the resulting normalization of $\mathcal{O}$ denoted $\mathit{cl}(\mathcal{O})$, every literal $\mathsf{Q}r.D$ satisfies $D \in \mathsf{N_D}$, and for every $D \in \mathsf{N_D}$, $\mathit{cl}(\mathcal{O})$ contains at most one literal of the form $\mathsf{Q}r.D$. Note that each definer D only occurs in literals of the form $\neg D$ or $\mathsf{Q}r.D$, that is, positive literals of the form D are not allowed.

By [46], we also have that $\mathit{cl}(\mathcal{O}) \equiv_{\mathit{sig}(\mathcal{O})} \mathcal{O}$.

Example 99 *Let $\mathcal{O}$ be the ontology defined in the first row of Table 5.1. By normalizing $\mathcal{O}$, we obtain the set $\mathit{cl}(\mathcal{O})$ shown in the second row of Table 5.1. The definers D_1, D_2 and D_3 in $\mathit{cl}(\mathcal{O})$ replace the concepts $C_{D_1} = \exists s.B_1$, $C_{D_2} = B_1$ and $C_{D_3} = B_2$, respectively.*

For a fixed normalization, we introduce a partial order $\preceq_d$ over all introduced definers, which is defined as the smallest reflexive-transitive relation over $\mathsf{N_D}$ s.t.

- $D' \preceq_d D$ if $\neg D \sqcup C \in \mathit{cl}(\mathcal{O})$ and $D' \in \mathit{sig}(C)$.

Intuitively, $D' \preceq_d D$ whenever $C_{D'}$ is contained in C_D. In Example 99, we have $D_2 \preceq_d D_1$, since $\neg D_1 \sqcup \exists s.D_2 \in \mathit{cl}(\mathcal{O})$. Note that our normalization ensures that $\preceq_d$ is always acyclic.

In the following, we always assume that ontology $\mathcal{O}$ and signature Σ do not contain definers, unless stated otherwise.

$\mathcal{O}$:	$A_1 \sqsubseteq \exists r.\exists s.B_1 \sqcup \exists r.B_2$	$B_1 \sqcap B_3 \sqsubseteq \bot$	$A_2 \sqsubseteq A_3 \sqcup \forall s.B_3$	$B_4 \sqsubseteq A_4$	$B_2 \sqsubseteq B_4$
$cl(\mathcal{O})$:	$\neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.D_3$	$\neg D_1 \sqcup \exists s.D_2$	$\neg D_2 \sqcup B_1$	$\neg D_3 \sqcup B_2$	$\neg A_2 \sqcup A_3 \sqcup \forall s.D_4$
	$\neg D_4 \sqcup B_3$	$\neg B_1 \sqcup \neg B_3$	$\neg B_2 \sqcup B_4$	$\neg B_4 \sqcup A_4$	
$RI_\Sigma(\mathcal{O})$:	$\neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.D_3$	$\neg D_1 \sqcup \exists s.D_2$	$\neg D_3 \sqcup B_2$	$A_2 \sqcup A_3 \sqcup \forall s.D_4$	$\neg B_1 \sqcup \neg B_3$
	$\neg B_2 \sqcup B_4$	$\neg B_4 \sqcup A_4$	$\neg D_2 \sqcup \neg D_4$		
$\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))$:	$\neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.D_3$	$\neg D_3 \sqcup B_2$	$\neg B_1 \sqcup \neg B_3$	$\neg B_2 \sqcup B_4$	$\neg B_4 \sqcup A_4$
	$\neg D_2 \sqcup \neg D_4$	$\neg D_1 \sqcup \neg A_2 \sqcup A_3$			
$\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$:	$\neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.D_3$	$\neg D_1 \sqcup \neg A_2 \sqcup A_3$	$\neg D_3 \sqcup A_4$		
$gm(\mathcal{O}, \Sigma)$:	$A_1 \sqsubseteq \exists r.\exists s.B_1 \sqcup \exists r.B_2$	$A_2 \sqcap \exists s.B_1 \sqsubseteq A_3$	$B_2 \sqsubseteq A_4$		
$gm^*(\mathcal{O}, \Sigma)$:	$A_1 \sqsubseteq \exists r.(\neg A_2 \sqcup A_3) \sqcup \exists r.A_4$				

Table 5.1: Ontologies generated throughout the running example.

5.3 . Role Forgetting

An ontology $\mathcal{M}$ is called a *role forgetting* [74] for $\mathcal{O}$ and Σ iff $\mathcal{M}$ is a uniform interpolant for $\mathcal{O}$ and $\Sigma' = \Sigma \cup \mathit{sig}_{\mathsf{C}}(\mathcal{O})$. Existing methods to compute role forgetting either rely on an external reasoner [76, 47] or use the *universal role* ∇ [77, 48]. The former approach can be expensive, while the latter produces axioms outside of $\mathcal{ALC}$. The normalization allows us to implement a more efficient solution within $\mathcal{ALC}$, which relies on an integrated reasoning procedure and an additional transformation step that produces so-called *role isolated ontologies*.

5.3.1 . Role isolated ontologies

The main idea is to separate concept names $A \in \mathsf{N_C}$ that occur with roles outside of the signature, using the following notations.

$$\mathit{Rol}(A, \mathcal{O}) = \{r \in \mathit{sig}(\mathcal{O}) \mid \mathsf{Q}r.A \text{ appears in } \mathcal{O},\ \mathsf{Q} \in \{\forall, \exists\}\}$$
$$\mathit{Out}_\Sigma(\mathcal{O}) = \{A \in \mathit{sig}(\mathcal{O}) \mid \mathit{Rol}(A, \mathcal{O}) \not\subseteq \Sigma\}$$

Definition 100 (Role-isolated ontology) *An ontology $\mathcal{O}$ is* role isolated *for Σ if (i) $\mathcal{O}$ is in normal form, and (ii) every axiom $\alpha \in \mathcal{O}$ is of one of the following forms:*

(c1) $L_1 \sqcup \ldots \sqcup L_n,\ L_i := \neg A$, *where* $A \in \mathit{Out}_\Sigma(\mathcal{O})$, *for all* $1 \leq i \leq n$;

(c2) $L_1 \sqcup \ldots \sqcup L_m,\ L_i := \mathsf{Q}r.A \mid B \mid \neg B$, *where* $r, A \in \mathit{sig}(\mathcal{O})$ *and* $B \notin \mathit{Out}_\Sigma(\mathcal{O})$, *for all* $1 \leq i \leq n$.

Thus, an axiom in a *role isolated* ontology falls into two disjoint categories: either (c1) it contains literals built only over concepts in $\mathit{Out}_\Sigma(\mathcal{O})$ or (c2) it contains role restrictions or literals built over concepts outside $\mathit{Out}_\Sigma(\mathcal{O})$. This means that a concept in $\mathit{Out}_\Sigma(\mathcal{O})$ can only appear together with other concepts in $\mathit{Out}_\Sigma(\mathcal{O})$. In other words, these concepts are "isolated" from other concepts of the form A or $Qr.A'$, where A is not in $\mathit{Out}_\Sigma(\mathcal{O})$, and r and A' are arbitrary role and concept names.

Example 101 (Example 99 cont'd) *For $\Sigma = \{r,\ A_1,\ A_1,\ A_2,\ A_3,\ A_4\}$, we have $\mathit{Out}_\Sigma(\mathit{cl}(\mathcal{O})) = \{D_2, D_4\}$. $\mathit{cl}(\mathcal{O})$ is not role isolated for Σ because of $\neg D_2 \sqcup B_1$.*

Given an ontology, we compute its role isolated form using the following definition.

Definition 102 *The* role isolated form $RI_\Sigma(\mathcal{O})$ *of $\mathcal{O}$ is defined as $RI_\Sigma(\mathcal{O}) := \mathit{cl}_\Sigma(\mathcal{O}) \cup \mathcal{D}_\Sigma(\mathcal{O})$, where*

- $\mathit{cl}_\Sigma(\mathcal{O}) \subseteq \mathit{cl}(\mathcal{O})$ *contains all $\alpha \in \mathit{cl}(\mathcal{O})$ s.t. if $\neg D$ is a literal of α, then $\mathit{Rol}(D', \mathit{cl}(\mathcal{O})) \subseteq \Sigma$ for all $D' \in \mathsf{N_D}$ s.t. $D \preceq_d D'$.*
- $\mathcal{D}_\Sigma(\mathcal{O})$ *is the set of axioms $\neg D_1 \sqcup \ldots \sqcup \neg D_n$ s.t.*

1. *$cl_\Sigma(\mathcal{O})$ contains axioms of the form*

$$C_1 \sqcup \mathsf{Q}_1 r.D_1,\ C_2 \sqcup \forall r.D_2,\ \ldots,\ C_n \sqcup \forall r.D_n,$$

where $r \in \mathsf{N_R} \setminus \Sigma$, $\mathsf{Q}_1 \in \{\forall, \exists\}$, and

2. *$\{D_1 \ldots, D_n\}$ is a minimal set of definers s.t.*

$$cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot\ .$$

Intuitively, if a definer D appears in $cl_\Sigma(\mathcal{O})$, then it should not depend on definers in $Out_\Sigma(\mathcal{O})$.

Example 103 (Example 101 cont'd) *We have:*

- *$cl_\Sigma(\mathcal{O}) = cl(\mathcal{O}) \setminus \{\neg D_2 \sqcup B_1, \neg D_4 \sqcup B_3\}$ because*

$$Rol(D_2, cl(\mathcal{O})) = Rol(D_4, cl(\mathcal{O})) = \{s\} \not\subseteq \Sigma$$

and,

- $\mathcal{D}_\Sigma(\mathcal{O}) = \{\neg D_2 \sqcup \neg D_4\}$.

Theorem 104 *$RI_\Sigma(\mathcal{O})$ is role isolated for Σ and we have $\mathcal{O} \equiv_{\Sigma \cup sig_\mathsf{C}(\mathcal{O})} RI_\Sigma(\mathcal{O})$.*

Proof. The proof is provided on Page 120. □

To compute $\mathcal{D}_\Sigma(\mathcal{O})$, we saturate $cl(\mathcal{O})$ using the inference rules shown in Fig. 5.4, which is sufficient due to the following lemma.

Lemma 105 *Let $\mathcal{S}$ be the set of axioms $\neg D_1 \sqcup \ldots \sqcup \neg D_n$, obtained by applying the rules in Fig. 5.4 exhaustively on $cl(\mathcal{O})$. Then, for all D_1, ..., $D_n \in \mathsf{N_D}$, we have $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ iff $\neg D_{i_1} \sqcup \ldots \sqcup \neg D_{i_k} \in \mathcal{S}$ for some subset $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$.*

Proof. The proof is provided on Page 114. □

Example 106 (Example 103 cont'd) *The axiom $\neg D_2 \sqcup \neg D_4$ in $\mathcal{D}_\Sigma(\mathcal{O})$ is obtained by applying two A-Rule inferences:*

$$\dfrac{\dfrac{\neg D_2 \sqcup B_1, \quad \neg B_1 \sqcup \neg B_3}{\neg D_2 \sqcup \neg B_3,} \qquad \neg D_4 \sqcup B_3}{\neg D_2 \sqcup \neg D_4}$$

$$\underline{\textit{A-Rule}}: \quad \frac{C_1 \sqcup A_1 \qquad \neg A_1 \sqcup C_2}{C_1 \sqcup C_2}$$

$$\underline{\textit{r-Rule}}: \quad \frac{C_1 \sqcup \exists r.D_1,\ \bigsqcup_{j=2}^{n}\{C_j \sqcup \forall r.D_j\},\ K_D}{C_1 \sqcup \ldots \sqcup C_n},$$

where $K_D = \neg D_1 \sqcup \ldots \sqcup \neg D_n$ or $\neg D_2 \sqcup \ldots \sqcup \neg D_n$, $D_1, \ldots, D_n \in \mathsf{N}_D$.

Figure 5.4: Inference rules for computing $\mathcal{D}_\Sigma(\mathcal{O})$

$$\underline{\textit{r-Rule}}: \quad \frac{C_1 \sqcup \exists r.A_1,\ \bigsqcup_{j=2}^{n}\{C_j \sqcup \forall r.A_j\},\ K_A}{C_1 \sqcup \ldots \sqcup C_n},$$

where $K_A = \neg A_1 \sqcup \ldots \sqcup \neg A_n$ or $\neg A_2 \sqcup \ldots \sqcup \neg A_n$, $A_1, \ldots, A_n \in \mathsf{N}_C$.

Figure 5.5: r-Rule for role isolated ontologies

5.3.2 . Role forgetting for role isolated ontologies

If $\mathcal{O}$ is role isolated for Σ, a role forgetting for $\mathcal{O}$ and Σ can be obtained using the *r-Rule* in Figure 5.4.

Note that unlike $RI_\Sigma(\mathcal{O})$, which is built from $cl(\mathcal{O})$, a real-world ontology $\mathcal{O}_1$ does not contain definers, regardless of whether it is role isolated or not. Definers are only introduced during the normalization process of $\mathcal{O}_1$ following the rules outlined in Figure 5.3. Thus, only $cl(\mathcal{O}_1)$ contains definers. Therefore, to apply the r-Rule on every role isolated ontology $\mathcal{O}$, we allow the concept names $D_1, \ldots, D_n$ in the r-Rule to include concept names outside $\mathsf{N_D}$. In other words, the r-Rule for a role isolated ontology $\mathcal{O}$ is of the form shown in Figure 5.5.

Definition 107 *$\mathtt{rolE}_\Sigma(\mathcal{O})$ is obtained by:*

1. *applying the* r-Rule *exhaustively for each $r \in sig_{\mathsf{R}}(\mathcal{O}) \setminus \Sigma$, and then*
2. *removing all axioms containing some $r \in sig_{\mathsf{R}}(\mathcal{O}) \setminus \Sigma$.*

The second step ensures that all role names in the resulting ontology $\mathtt{rolE}_\Sigma(\mathcal{O})$ are in Σ and therefore, we have $sig(\mathtt{rolE}_\Sigma(\mathcal{O})) \subseteq \Sigma \cup sig_{\mathsf{C}}(\mathcal{O})$.

Example 108 (Example 106 cont'd) *For the ontology $RI_\Sigma(\mathcal{O})$, Table 5.1 (fourth row) shows $\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))$ which is obtained through the following two steps:*

1. *The new axiom* $\neg D_1 \sqcup \neg A_2 \sqcup A_3$ *is generated by the r-Rule inference:*

$$\frac{\neg D_1 \sqcup \exists s.D_2,\ \neg A_2 \sqcup A_3 \sqcup \forall s.D_4,\ \neg D_2 \sqcup \neg D_4}{\neg D_1 \sqcup \neg A_2 \sqcup A_3}$$

2. *The two axioms* $\neg D_1 \sqcup \exists s.D_2,\ \neg A_2 \sqcup A_3 \sqcup \forall s.D_4$ *are removed because they contain* $s \in \mathit{sig}(RI_\Sigma(\mathcal{O})) \setminus \Sigma$.

Theorem 109 *If* $\mathcal{O}$ *is role isolated for* Σ*, then* $\mathtt{rolE}_\Sigma(\mathcal{O})$ *is a role-forgetting for* $\mathcal{O}$ *and* Σ.

Proof. The proof is provided on Page 122. □

5.4 . Computing General Modules via $\mathtt{rolE}_\Sigma$

Our goal now is to compute a general module starting from forgetting also the concept names (obtaining a hopefully more concise representation), and eliminating all definers (ensuring the result is entailed by $\mathcal{O}$).

5.4.1 . Concept forgetting

We say that an ontology $\mathcal{M}$ is a *concept forgetting* [74] for $\mathcal{O}$ and Σ iff $\mathcal{M}$ is a uniform interpolant for $\mathcal{O}$ and the signature $\Sigma' = \Sigma \cup \mathit{sig}_{\mathsf{R}}(\mathcal{O}) \cup \mathsf{N}_{\mathsf{D}}$.

A concept forgetting can be computed through the *A-Rule* in Figure 5.4.

Definition 110 $\mathtt{conE}_\Sigma(\mathcal{O})$ *is the ontology obtained as follows:*

1. *apply the* A-Rule *exhaustively for each* $A \in \mathit{sig}_{\mathsf{C}}(\mathcal{O}) \setminus \Sigma$*, and then*
2. *delete every axiom* α *that contains* A *or* $\neg A$*, where* $A \in \mathsf{N}_{\mathsf{C}} \setminus \Sigma$ *and no axiom contains* $\mathsf{Q}r.A$ *for* $\mathsf{Q} \in \{\forall, \exists\}$ *and* $r \in \mathsf{N}_{\mathsf{R}}$.

Example 111 (Example 108 cont'd) *Table 5.1 (the 5th row) shows the axioms in* $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ *obtained as follows.*

1. $\neg D_3 \sqcup B_4$*,* $\neg B_2 \sqcup A_4$*, and* $\neg D_3 \sqcup A_4$ *are first generated by applying the A-Rule on* B_2 *and* B_4.
2. *Axioms containing* B_i *or* $\neg B_i$*,* $i \in \{1, \ldots, 4\}$*, are removed since* $B_i \notin \Sigma$. $\neg D_2 \sqcup \neg D_4$ *is also removed because there are no literals of the form* $\mathsf{Q}r.D_2$ *or* $\mathsf{Q}r.D_4$.

The following is a consequence of [77, Theorem 1].

Theorem 112 *If* $\mathcal{O}$ *is in normal form, then* $\mathtt{conE}_\Sigma(\mathcal{O})$ *is a concept forgetting for* $\mathcal{O}$ *and* Σ.

Theorems 104, 109 and 112, imply the following corollary.

Corollary 113 $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \equiv_\Sigma \mathcal{O}$.

5.4.2 . Constructing general modules

Now, in order to obtain our general modules, we need to eliminate the definers from $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$. This is done by simple substitution. That is, each definer D is replaced by their corresponding concept C_D assigned during the normalisation process. Moreover, we apply the following two operations to improve the results:

1. We delete *subsumed axioms* in $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ before eliminating definers. Here, we say an axiom $\top \sqsubseteq C \sqcup D \in \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ is subsumed if there exists another axioms in $\top \sqsubseteq C \in \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$. That is, $\top \sqsubseteq C \sqcup D$ is redundant and can be removed from $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ without loss of any logical information;

2. We rewrite axioms of the form $\top \sqsubseteq \neg C \sqcup D$ in the standard form $C \sqsubseteq D$ to make it more readable.

Theorem 114 *Let $gm_\Sigma(\mathcal{O})$ be the ontology obtained from $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ by*

- *deleting subsumed axioms,*
- *replacing each definer D by C_D, and*
- *exhaustively applying the two transformations below*

$$C_1 \sqsubseteq \neg C_2 \sqcup C_3 \;\Rightarrow\; C_1 \sqcap C_2 \sqsubseteq C_3$$
$$C_1 \sqsubseteq \mathsf{Q}r.\neg C_2 \sqcap C_3 \;\Rightarrow\; C_1 \sqcap \overline{\mathsf{Q}}r.C_2 \sqsubseteq C_3,$$

where $\overline{\exists} = \forall$ and $\overline{\forall} = \exists$.

Then, $gm_\Sigma(\mathcal{O})$ is a general module for $\mathcal{O}$ and Σ.

Proof. The proof is provided on Page 122. □

Example 115 (Example 111 cont'd) *Table 5.1 (the 6th row) shows the general module $gm_\Sigma(\mathcal{O})$, which has been obtained using $C_{D_1}{=}\exists s.B_1$ and $C_{D_3}{=}B_2$.*

Eliminating definers in this way may reintroduce previously forgotten names, which is why our general modules are in general not uniform interpolants. This has the advantage of avoiding the triple exponential blow-up (wrt ontology length) caused by uniform interpolation [55]. In contrast, the length of our result is at most single exponential in the size of the input.

Proposition 116 *For any ontology $\mathcal{O}$ and signature Σ, we have $\|gm_\Sigma(\mathcal{O})\| \leq 2^{O(\|cl(\mathcal{O})\|)}$. On the other hand, there exists a family of ontologies $\mathcal{O}_n$ and signatures Σ_n s.t. $\|\mathcal{O}_n\|$ is polynomial in $n \geq 1$ and $\|gm_{\Sigma_n}(\mathcal{O}_n)\| = n \cdot 2^{O(\|cl(\mathcal{O}_n)\|)}$.*

conD-Elim:

$$\frac{C_1 \sqcup \mathsf{Q}r.D_1, \bigcup_{j=2}^{n}\{C_j \sqcup \forall r.D_j\}, \neg D_1 \sqcup \ldots \sqcup \neg D_n}{C_1 \sqcup \ldots \sqcup C_n \sqcup \mathsf{Q}r.\bot}$$

D-Prop : $$\frac{C_1 \sqcup \mathsf{Q}r.D, \quad \bigcup_{j=2}^{n}\{\neg D \sqcup C_j\}}{C_1 \sqcup \mathsf{Q}r.(C_2 \sqcap \ldots \sqcap C_n)}$$

where $\bigcup_{j=2}^{n}\{\neg D \sqcup C_j\}$ ($n \geq 1$) are all the axioms of the form $\neg D \sqcup C$. This rule is applicable only if no C_j contains definers.

Figure 5.6: Rules to eliminate definers

Proof. The proof is provided on Page 124. □

For some module extraction methods, such as for locality-based modules [27], iterating the computation can lead to smaller modules. The following result shows that this is never the case for our method.

Proposition 117 *Let $(\mathcal{M}_i)_{i\geq 1}$ be the sequence of ontologies defined by (i) $\mathcal{M}_1 = gm_\Sigma(\mathcal{O})$ and (ii) $\mathcal{M}_{i+1} = gm_\Sigma(\mathcal{M}_i)$ for $i \geq 1$. Then, we have*

$$\mathcal{M}_i \subseteq \mathcal{M}_{i+1} \text{ for } i \geq 1.$$

Moreover, there exists $i_0 \geq 0$ s.t. $\mathcal{M}^k = \mathcal{M}^{i_0}$ for all $k \geq i_0$.

Proof. The proof is provided on Page 128. □

This property is due to the substitution step of Theorem 114. This step may reintroduce in $gm_\Sigma(\mathcal{O})$ concept and role names outside of Σ. As a result, the repeated application of $\mathtt{rolE}_\Sigma$ and $\mathtt{conE}_\Sigma$ on $gm_\Sigma(\mathcal{O})$ can produce additional but unnecessary axioms. However, for ontologies in normal form, our method is *stable* in the sense that repeated applications produce the same ontology.

Proposition 118 *Let $\mathcal{O}$ be an ontology in normal form and $\mathcal{M} = gm_\Sigma(\mathcal{O})$. Then, $gm_\Sigma(\mathcal{M})$=$\mathcal{M}$.*

Proof. The proof is provided on Page 126. □

5.4.3 . Optimizing the result

The general module $gm_\Sigma(\mathcal{O})$ may contain complex axioms since the definers D can stand for complex concepts C_D. To make the result more concise, we eliminate some definers before substituting them. In particular, we use the following operations on $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$, inspired by [68].

Op1. **Eliminating conjunctions of definers** aims to eliminate disjunctions of negative definers ($\neg D_1 \sqcup \ldots \sqcup \neg D_n$). This is done in two steps: (i) Applying the *conD-Elim* rule of Figure 5.6 on $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$, and then (ii) deleting all axioms of the form $\neg D_1 \sqcup \ldots \sqcup \neg D_n$.

Op2. **Eliminating single definers** aims to get rid of definers D that do not occur in axioms of the form $\neg D \sqcup \neg D_1 \sqcup C$. This is done in two steps: (i) applying the *D-Prop* rule of Figure 5.6 exhaustively and then (ii) deleting all axioms containing definers for which *D-Prop* has been applied.

Theorem 119 *Let $gm^*_\Sigma(\mathcal{O})$ be the ontology obtained by:*

- *successive application of Op1 and Op2 over $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$, and then*
- *application of the steps described in Theorem 114.*

Then, $gm^*_\Sigma(\mathcal{O})$ is a general module for $\mathcal{O}$ and Σ.

Proof. The proof is provided on Page 130. □

Example 120 (Example 115 cont'd) *Table 5.1 (the 7th row) shows the general module $gm^*_\Sigma(\mathcal{O})$, which has been obtained by applying Op2 (i.e., apply D-Prop rule for D_1 and D_3) on $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$.*

To illustrate operation Op1, we provide another example below.

Example 121 *Assume $\Sigma = \{r, A, A_1\}$ and*

$$\mathcal{O} = \{A \sqsubseteq \forall r.\exists s.B_1,\ A_1 \sqsubseteq \forall r.\forall s.B_2,\ B_1 \sqcap B_2 \sqsubseteq \bot\}.$$

Then, $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ is:

$$\{\neg A \sqcup \forall r.D_1,\ \neg A_1 \sqcup \forall r.D_2,\ \neg D_1 \sqcup \neg D_2\},$$

where $C_{D_1} = \exists s.B_1,\ C_{D_2} = \forall s.B_2$. And thus, by replacing D_i by C_{D_i}, we obtain $gm_\Sigma(\mathcal{O})$=

$$\{A \sqsubseteq \forall r.\exists s.B_1,\ A_1 \sqsubseteq \forall r.\forall s.B_2,\ \exists s.B_1 \sqcap \forall s.B_2 \sqsubseteq \bot\},$$

which is actually more intricate than $\mathcal{O}$. We can avoid this by applying the two optimizations described above.

The elimination of definer conjunctions (Op1) produces

$$\{\neg A \sqcup \forall r.D_1, \neg A_1 \sqcup \forall r.D_2, \neg A \sqcup \neg A_1 \sqcup \forall r.\bot\}. \tag{5.1}$$

(i) The first step of Op1 applies the conD-Elim inference:

$$\frac{\neg A \sqcup \forall r.D_1, \neg A_1 \sqcup \forall r.D_2, \neg D_1 \sqcup \neg D_2}{\neg A \sqcup \neg A_1 \sqcup \forall r.\bot}.$$

(ii) The second step of Op1 removes the axiom $\neg D_1 \sqcup \neg D_2$.
Then, the elimination of definers (Op2) produces

$$\{\neg A \sqcup \forall r.\top, \neg A_1 \sqcup \forall r.\top, \neg A \sqcup \neg A_1 \sqcup \forall r.\bot\} \tag{5.2}$$

by replacing D_1, D_2 *by* $\top$ *as there is no axioms with negative* $D_i, i = 1, 2$ *in Equation (5.1). Note that the first two axioms in Equation (5.2) are tautologies and thus can be ignored.*

Finally, we have $gm^*_\Sigma(\mathcal{O}) = \{A \sqcap A_1 \sqsubseteq \forall r.\bot\}$.

5.5 . Deductive Modules and Uniform Interpolants

Deductive modules Depending on the situation, users might prefer to get a view (module) containing the axioms of the original ontology $\mathcal{O}$ rather than new axioms (e.g., axioms in $gm_\Sigma(\mathcal{O})$ or $gm^*_\Sigma(\mathcal{O})$). For such situations, we can compute a deductive module for $\mathcal{O}$ and Σ by tracing back the inferences performed during the computation of the general module $gm^*_\Sigma(\mathcal{O})$.

Let $Res_\Sigma(\mathcal{O})$ be the set of all axioms generated by the computation process of $gm^*_\Sigma(\mathcal{O})$. Clearly, $gm^*_\Sigma(\mathcal{O}) \subseteq Res_\Sigma(\mathcal{O})$. Then, the deductive module $dm_\Sigma(\mathcal{O})$ is defined using a relation links certain elements of $Res_\Sigma(\mathcal{O})$. A relation R over the set $Res_\Sigma(\mathcal{O})$ is defined as a set of ordered pairs of elements in $Res_\Sigma(\mathcal{O})$. That is, a set of the form

$$R = \{(\alpha, \beta) \mid \alpha, \beta \in Res_\Sigma(\mathcal{O})\}.$$

Next, we write $\alpha R \beta$ iff $(\alpha, \beta) \in R$.

The relation R on $Res_\Sigma(\mathcal{O})$, used to define the deductive module $dm_\Sigma(\mathcal{O})$, is built iteratively during the computation of $gm^*_\Sigma(\mathcal{O})$ as follows

- Start with the empty relation $R = \emptyset$;
- Each time a new axiom β is generated from a premise set $\{\alpha_1, \ldots, \alpha_n\}$ (e.g., if β is obtained by applying r-Rule on $\{\alpha_1, \ldots, \alpha_n\}$), we add to R the elements $\alpha_1 R \beta$, $\ldots$, $\alpha_n R \beta$.

Let R^* be the smallest *transitive closure* of R. That is, R^* satisfies the following two conditions:

- For any axioms α, β, if $\alpha R \beta$, then $\alpha R^* \beta$;
- For any axioms α, β, γ, if $\alpha R^* \beta$ and $\beta R^* \gamma$, then $\alpha R^* \beta$.

Then, we are able to define a deductive module as follows.

Theorem 122 *Let us define* $dm_\Sigma(\mathcal{O})$ *by*

$$dm_\Sigma(\mathcal{O}) = \{\alpha \in \mathcal{O} \mid \alpha R^* \beta \text{ for some } \beta \in gm^*_\Sigma(\mathcal{O})\}.$$

Then, $dm_\Sigma(\mathcal{O})$ *is a deductive module for* $\mathcal{O}$ *and* Σ.

Proof. The proof is provided on Page 132. □

Example 123 (Example 115 cont'd) *In the computation progress of $gm^*_\Sigma(\mathcal{O})$, we obtain the relation R that include the following relations:*

1. *We have $\alpha_1 R \alpha_2$, where*

$$\alpha_1 = A_1 \sqsubseteq \exists r.\exists s.B_1 \sqcup \exists r.B_2 \in \mathcal{O},$$
$$\alpha_2 = \neg A_1 \sqcup \exists r.D_1 \sqcup \exists r.D_3 \in cl(\mathcal{O}),$$

   as α_2 is obtained by normlizing α_1;

2. *We have $\beta_1 R \beta_4$, $\beta_2 R \beta_4$, $\beta_3 R \beta_4$, where*

$$\beta_1 = \neg D_1 \sqcup \exists s.D_2 \in RI_\Sigma(\mathcal{O}),$$
$$\beta_2 = \neg A_2 \sqcup A_3 \sqcup \forall s.D_4 \in RI_\Sigma(\mathcal{O}),$$
$$\beta_3 = \neg D_2 \sqcup \neg D_4 \in RI_\Sigma(\mathcal{O}),$$
$$\beta_4 = \neg D_1 \sqcup \neg A_2 \sqcup A_3 \in \mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})),$$

   as β_4 is generated by applying the r-Rule on $\{\beta_1,\ \beta_2,\ \beta_3\}$.

For simplicity, here, we do not provided all elements of R and R^*. Indeed, in this running example, we have $dm_\Sigma(\mathcal{O}) = \mathcal{O}$ since all the axioms of $\mathcal{O}$ are necessary for generating $gm^*_\Sigma(\mathcal{O})$.

Uniform interpolants While general modules can be a good alternative to uniform interpolants for ontology reuse, uniform interpolation has applications that require the ontology to be fully over the selected signature, as stated in the introduction. If instead of substituting definers D by C_D, we eliminate them using existing uniform interpolation tools [47, 78], we can compute a uniform interpolant for the input.

When computing $gm^*_\Sigma(\mathcal{O})$ for an ontology $\mathcal{O}$ and a signature Σ, if all definers have been eliminated by Op1 and Op2 from Section 5.4.3 (as in Example 121), then $gm^*_\Sigma(\mathcal{O})$ is indeed a uniform interpolant for $\mathcal{O}$ and Σ. If this is not the case, we compute a uniform interpolant by forgetting the remaining definers using an existing uniform interpolation tool such as LETHE or FAME [47, 78]. In detail, the uniform interpolant is obtained by the following three steps:

1. Computing $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$;
2. Applying successively Op1 and Op2 over $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$. Let us now denote by $\mathcal{O}_1$ the obtained ontology;
3. If $\mathcal{O}_1$ contains definers, applying LETHE or FAME on $\mathcal{O}_1$ in order to output a uniform interpolant for $\mathcal{O}_1$ and Σ. Otherwise, output $gm^*_\Sigma(\mathcal{O})$, it is a uniform interpolant.

As we will see in Section 5.6, this allows us to compute uniform interpolants much faster than using the tool alone.

5.6 . Evaluation

We implemented a prototype in Python (version 3.7) called GEMO. To validate our methods, we compare them with each other and with the state-of-the-art tools implementing module extraction and uniform interpolation methods for $\mathcal{ALC}$. As evaluation metrics, we are interested in run time, length of computed ontologies, and length of largest axioms in the result. All the experiments were run on a machine with an Intel Xeon Silver 4112 2.6GHz, 64 GiB of RAM, Ubuntu 18.04, and OpenJDK 11.

Corpus The ontologies used in our experiment are generated from the OWL Reasoner Evaluation (ORE) 2015 classification track [61] by the following two steps. First, we removed axioms outside of $\mathcal{ALC}$ from each ontology in ORE 2015. Then, we kept the ontologies $\mathcal{O}$ for which $cl(\mathcal{O})$ contained between 100 and 100,000 names. This resulted in 222 ontologies.

Signatures For each ontology, we generated 50 signatures consisting of 100 concept and role names. As in [48], we selected each concept/role name with a probability proportional to its occurrence frequency in the ontology. We fix the size of the sigantures at 100 because we are primarily interested in situations where the signature is relatively small, such as the symptoms for individual patients.

In the following, a *request* is a pair consisting of an ontology and a signature.

Methods For each request $(\mathcal{O}, \Sigma)$, GEMO produces three different (general) modules $gm_\Sigma(\mathcal{O})$, $gm^*_\Sigma(\mathcal{O})$ and $dm_\Sigma(\mathcal{O})$, respectively denoted by `gm`, `gm`*, and `dm`. Moreover, `gmLethe` denotes the uniform interpolation method described in Section 5.5, where we use GEMO for computing `gm`* and then LETHE for definer forgetting. In the implementation, for each request, we first extracted a locality based $\top\bot^*$*-module* [27] to accelerate the computation. Since removing subsumed axioms as mentioned in Theorems 114 and 119 can be challenging, we set a time limit of 10s for this task. Thus, subsumed axioms may be only partially removed.

We compared our methods with four different alternatives:

- $\top\bot^*$*-module*s [27] as implemented in the OWL API [29];
- `minM` [48] that computes *minimal deductive modules* under $\mathcal{ALCH}^\nabla$-semantics;
- LETHE (version 0.6, avaliable at https://lat.inf.tu-dresden.de/~koopmann/LETHE/)[47] that computes uniform interpolants;
- FAME (version 1.0, avaliable at http://www.cs.man.ac.uk/~schmidt/sf-fame/) [78] that computes uniform interpolants.

Success rate We say a method *succeeds* on a request if it outputs the expected results within 600s. Table 5.2 summarizes the success rate for the methods considered. Except for the $\top\bot^*$-modules, our method GeMo had the highest success rate.

$\top\bot^*$-module	minM	Lethe	Fame	GeMo	gmLethe
100 %	84.34 %	85.27 %	91.25 %	97.34%	96.17 %

Table 5.2: Success rate evaluation. The first (resp. second) best-performing method is highlighted in red (resp. blue).

Module length and run time Because the axiom shape in general modules is arbitrary for $\mathcal{ALC}$, one can easily transform any ontology into an equivalent one with just one axiom. Thus, we chose to use length as defined in Definition 9, rather than size, for our evaluation. Table 5.3 shows the length and run time for the requests on which all methods were successful (78.45% of all requests).

We observe that dm and gmLethe have the best overall performance: their results had a substantially smaller average length and were computed significantly faster than others. Note that the average size of results for dm was even smaller than that for minM. The reason is that minM preserves entailments over $\mathcal{ALCH}^{\nabla}$, while we preserve only entailments over $\mathcal{ALC}$. Therefore, the minM results may contain additional axioms compared to the $\mathcal{ALC}$ deductive modules.

Comparing gm and gm* regarding length, we conclude that the optimization in Section 5.4.3 is effective. On the other hand, minM produced results of small length but at the cost of long computation times. Fame and $\top\bot^*$-module were quite time-efficient but less satisfactory in size, especially for Fame, whose results are often considerably larger than for the other methods. Lethe took more time than Fame, but produced more concise uniform interpolants on average.

For 87.87% of the requests reported in Table 5.3, gm* already computed a uniform interpolant, so that gmLethe did not need to perform any additional computations.

Figure 5.7 provides a detailed comparison of minM, gm*, and gmLethe. It shows that gm* was often faster but produced larger results. In contrast, gmLethe produced more concise results at the cost of longer computation time. While minM avoided large modules, it was generally much slower than our methods.

Table 5.4 summarizes the results concerning all requests for which GeMo (resp. gmLethe) was successful. We see that the results of dm had a small average size. However, as for gm* and gmLethe, the median size of results was much smaller, which suggests that gm* and gmLethe perform better over relatively simple cases.

Methods		Resulting ontology length	Time cost (s)
minM		2355 / 392.59 / 264	595.88 / 51.82 / 8.86
$\top\bot^*$-module		4,008 / 510.77 / 364	5.94 / 1.03 / 0.90
Fame		9,446,325 / 6,661.01 / 271	526.28 / 3.20 / 1.17
Lethe		131,886 / 609.30 / 196	598.20 / 49.21 / 13.57
GeMo	gm	179,999 / 2,335.05 / 195	17.50 / 2.44 / 1.63
	gm*	21,891 / 466.15 / 166	
	dm	2,789 / 366.36 / 249	
gmLethe		21,891 / 364.10 /162	513.15 / 3.08 / 1.68

Table 5.3: Comparison of different methods (max. / avg. / med.).

Methods		Resulting ontology length	Time cost (s)
GeMo	gm	17,335,040 / 35,008.2 / 310	585.97 / 4.89 / 1.75
	gm*	2,318,878 / 2,978.77 / 214	
	dm	18,218 / 638.74 / 309	
gmLethe		353,107 / 1,006.34 /192	579.70 / 7.56 / 2.02

Table 5.4: GeMo and gmLethe: Summary of results for all their own successful experiments (max. / avg. / med.).

Uniform Interpolants For 80.23% of GeMo successful requests, $gm^*_\Sigma(\mathcal{O})$ are uniform interpolants. In the cases where gm* did not already produce a uniform interpolant, the success rate for gmLethe was 93.96%.

In the cases where Lethe failed, the success rate for gmLethe was 36.23%. On the other hand, there are 87 requests (0.78% of all tested requests), on which Lethe succeeded but gmLethe failed.

The comparison of Lethe with gmLethe in Figure 5.8 shows that gmLethe was significantly faster than Lethe in most of the cases.

Axiom Size A potential shortcoming of general modules compared to classical modules is that they could contain axioms that are more complex than those of the input, and thus be harder to handle by human end-users. For the requests reported in Table 5.3, the largest axiom in the output of minM had length 352, while for gm*, it had length 5,815. In contrast, for the uniform interpolants computed by Lethe and Fame, the situation was much worse: here, the largest axiom had a length of 26,840 and 130,700, respectively, which is clearly beyond what can be understood by a human end-user. Besides these extreme cases, we can also observe differences wrt. the median values: for gm* the longest axiom had a median length of 3, which is even lower than the corresponding value for minM (5), and, as expected, lower than for Lethe (4) and Fame (6). This indicates that, in most cases, general

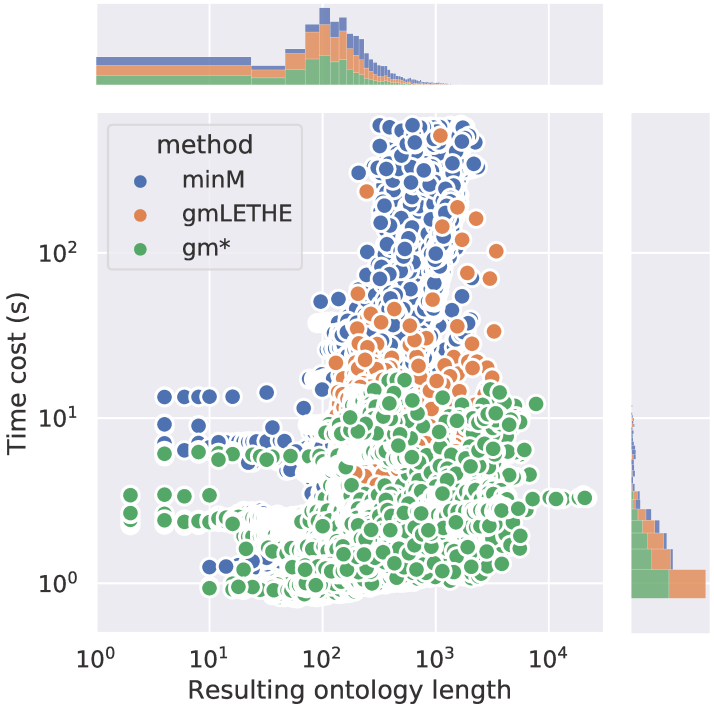


Figure 5.7: Comparison of `minM`, `gmLETHE`, and `gm`* using the logarithmic scale. The points are drawn in the sequence of blue, orange, and green (the green points may cover the blue points).

modules computed by `gm`* are simpler.

5.7 . Proofs of Results

The order in which we prove the results differs slightly from the order in which they appear in the main text. In particular, since the proof of Proposition 117 depends on some arguments used for Proposition 118, we show it after the proof for Proposition 118.

Lemma 105

Fix an ontology $\mathcal{O}$, and fix a set $\mathsf{N_P} \subseteq \mathsf{N_C}$ of concept names such that $\mathsf{N_P} \cap \mathit{sig}(\mathcal{O}) = \mathsf{N_P} \cap \mathsf{N_D} = \emptyset$. We will use the concept names $P \in \mathsf{N_P}$ as *placeholders* to substitute literals of the from $\mathsf{Q}r.D$ in $\mathit{cl}(\mathcal{O})$. In particular, we normalize the axioms in $\mathit{cl}(\mathcal{O})$ further so that every axiom is of one of the following forms.

1. $\neg P \sqcup \mathsf{Q}r.D$, where $\mathsf{Q} \in \{\forall, \exists\}$, and $P \in \mathsf{N_P}$;
2. or $L_1 \sqcup \ldots \sqcup L_n$, where each L_i is of the forms A, or $\neg B$, where $A \in \mathsf{N_C} \setminus \mathsf{N_D}$, $B \in \mathsf{N_C} \setminus \mathsf{N_P}$.

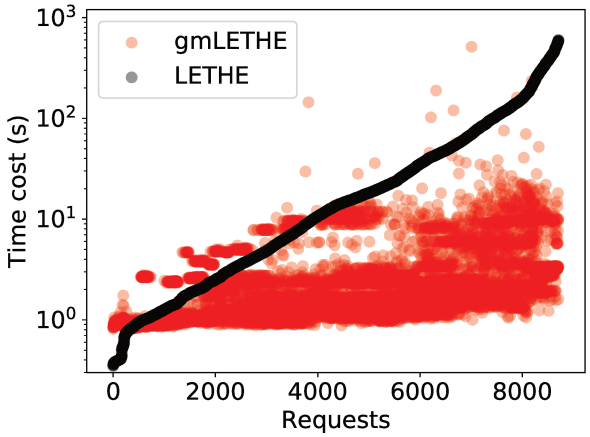


Figure 5.8: Run time comparison of Lethe (grey points) and `gmLethe` (red points) using the logarithmic scale in time cost. The requests are arranged in ascending order of time cost for Lethe.

Denote the resulting ontology by $cl_P(\mathcal{O})$. Similar as with the normalization introduced in Section 5.2, we can ensure that $cl_P(\mathcal{O}) \equiv_{sig(cl(\mathcal{O}))} cl(\mathcal{O})$.

In order to prove Lemma 105, we make use of the inference calculus introduced in [70], which can be seen in Table 5.5, adapted to our syntax of normalized axioms. Here, H and K stand for conjunctions of concept names (i.e, $A_1 \sqcap \ldots \sqcap A_n$), and N for a disjunction of concept names (i.e, $A_1 \sqcup \ldots \sqcup A_n$). We write

$$\mathcal{O} \vdash_P H \sqsubseteq \bot$$

if $H \sqsubseteq \bot$ can be derived from $\mathcal{O}$ using the rules in Table 5.5. We have the following theorem as the adaptation of Table 2.2 and Theorem 22 on page 20 to $cl_P(\mathcal{O})$.

Theorem 124 *Let $\mathcal{O}$ be an ontology and H be any conjunction of concept names. Then, $cl_P(\mathcal{O}) \models H \sqsubseteq \bot$ iff $cl_P(\mathcal{O}) \vdash_P H \sqsubseteq \bot$.*

In the following, we write

$$cl_P(\mathcal{O}) \vdash \neg D_1 \sqcup \ldots \sqcup \neg D_n$$

if we can derive $\neg D_1 \sqcup \ldots \sqcup \neg D_n$ from $cl_P(\mathcal{O})$ using applications of the A-Rule and the r-Rule. Before we show Lemma 105, we need an auxiliary lemma.

Lemma 125 *Let $N = A_1 \sqcup \ldots \sqcup A_m$ be an axiom s.t. $A_j \in N_C$ for each $1 \leq j \leq m$, let $D_1, \ldots, D_n \in N_D$ be definers, and assume $cl_P(\mathcal{O}) \vdash_P D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N$. Then, $D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N$ is a tautology, or $cl_P(\mathcal{O}) \vdash N'$, where N' is a disjunction over some subset of $\{\neg D_1, \ldots, \neg D_n, A_1, \ldots, A_n\}$.*

$$\mathbf{R}^{+}_{\mathbf{A}}\ \frac{}{H \sqsubseteq A} : A \in H \qquad \mathbf{R}^{-}_{\mathbf{A}}\ \frac{H \sqsubseteq N \sqcup A}{H \sqsubseteq N} : \neg A \in H$$

$$\mathbf{R}^{\mathbf{n}}_{\sqcap}\ \frac{\{H \sqsubseteq N_i \sqcup A_i\}_{i=1}^{n}}{H \sqsubseteq \sqcup_{i=1}^{n} N_i \sqcup N} : (\bigsqcup_{i=1}^{n} \neg A_i) \sqcup N \in cl_P(\mathcal{O})$$

$$\mathbf{R}^{+}_{\exists}\ \frac{H \sqsubseteq N \sqcup P}{H \sqsubseteq N \sqcup \exists r.D} : \neg P \sqcup \exists r.D \in cl_P(\mathcal{O})$$

$$\mathbf{R}^{\bot}_{\exists}\ \frac{H \sqsubseteq N \sqcup \exists r.K, K \sqsubseteq \bot}{H \sqsubseteq N}$$

$$\mathbf{R}_{\forall}\ \frac{H \sqsubseteq N \sqcup \exists r.K, H \sqsubseteq N_1 \sqcup P}{H \sqsubseteq N \sqcup N_1 \sqcup \exists r.(K \sqcap D)} : \neg P \sqcup \forall r.D \in cl_P(\mathcal{O})$$

Table 5.5: Adaptation of Table 2.2 to the syntax in $cl_P(\mathcal{O})$.

Proof. Let $k_P(D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N)$ be the minimal number of applications of rules from Table 5.5 required to derive $D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N$. We prove the lemma by induction on $k_P(D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N)$.

1. If $k_P(D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N) = 1$, then we have $N = D_i$ for some $1 \leq i \leq n$. In this case, the lemma holds directly.

2. Assume that the lemma holds for all $D'_1, \ldots, D'_{n'}, N'$ s.t. $k_P(D'_1 \sqcap \ldots \sqcap D_{n'} \sqsubseteq N') < k_0$ for some $k_0 \geq 1$.

   We show that the lemma also holds for $D_1, \ldots, D_n, N$ s.t. $k_P(D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N) = k_0$. For simplicity, let $H_D := D_1 \sqcap \ldots \sqcap D_n$, and let

   $$\rho_1, \ldots, \rho_{k_0}$$

   be a sequence of inferences generated by inference rules in Table 5.5 that derives $H_D \sqsubseteq N$. Then there are two different cases, depending on which was the last inference performed.

   (a) The last inference ρ_{k_0} is of the form

   $$\frac{\{H_D \sqsubseteq N_i \sqcup A_i\}_{i=1}^{n}}{H_D \sqsubseteq \bigsqcup_{i=1}^{n} N_i \sqcup N_0}, \left(\bigsqcup_{i=1}^{n} \neg A_i\right) \sqcup N_0 \in cl_P(\mathcal{O}).$$

   In particular, $N = \bigsqcup_{i=1}^{n} N_i \sqcup N_0$. We show that we then also have $cl_P(\mathcal{O}) \vdash D_1 \sqcap \ldots \sqcap D_n \sqsubseteq N$.

   If $H_D \sqsubseteq N_{i_0} \sqcup A_{i_0}$ is a tautology for some $1 \leq i_0 \leq n$, then one of $H_D \sqsubseteq N_{i_0}$, $H_D \sqsubseteq A_{i_0}$ must be a tautology. There are two cases:

   (i) If $H_D \sqsubseteq N_{i_0}$ is a tautology, then $H_D \sqsubseteq N$ is also a tautology since N_{i_0} is a sub-concept of N. Therefore, the lemma holds for this case;

   (ii) If $H_D \sqsubseteq A_{i_0}$ is a tautology, then $A_{i_0} \in \{D_1, \cdots, D_n\}$ must be a definer. By the construction of $cl_P(\mathcal{O})$, we have $n = 1$ (in the formula of

inference ρ_{k_0}) and thus $\neg A_{i_0} \sqcup N_0 \in cl_P(\mathcal{O})$. Then $cl_P \models \neg A_{i_0} \sqcup N_0$. Therefore, the lemma holds also for this case.

We obtain that the lemma holds for the case where $H_D \sqsubseteq N_{i_0} \sqcup A_{i_0}$ is a tautology for some $1 \leq i_0 \leq n$.

Now assume that $H_D \sqsubseteq N_i \sqcup A_i$ is not a tautology for any $1 \leq i \leq n$.

Since $k(H_D \sqsubseteq N_i \sqcup A_i) < k_0$, by applying our inductive hypothesis on $H_D \sqsubseteq N_i \sqcup A_i$, $i \in [1, n]$, we have

$$cl_P(\mathcal{O}) \vdash K_D^i \sqcup N_i^{sub}$$
$$\text{or} \quad cl_P(\mathcal{O}) \vdash K_D^i \sqcup N_i^{sub} \sqcup A_i$$

for some $K_D^i = \bigsqcup_{j \in I_i} \neg D_j$, $I_i \subseteq \{1, \ldots, n\}$ and N_i^{sub} being a disjunction over some concept names occurring in N_i. We distinguish those two cases.

(i) If $cl_P(\mathcal{O}) \vdash K_D^i \sqcup N_i^{sub}$ for some $1 \leq i \leq n$, then $cl_P(\mathcal{O}) \vdash K_D^i \sqcup N_i^{sub}$ is as desired.

(ii) Otherwise, $cl_P(\mathcal{O}) \vdash K_D^i \sqcup N_i^{sub} \sqcup A_i$ for all $1 \leq i \leq n$. By applying the A-Rule for all $A_i, 1 \leq i \leq n$ on

$$K_D^i \sqcup N_i^{sub} \sqcup A_i, \quad (1 \leq i \leq n)$$
$$\text{and } (\bigsqcup_{i=1}^{n} \neg A_i) \sqcup N_0 \in cl_P(\mathcal{O}),$$

we obtain the desired conclusion

$$cl_P(\mathcal{O}) \vdash K_D \sqcup N^{sub},$$

where

$$K_D = \bigsqcup_{1 \leq i \leq n} K_D^i, \text{ and}$$
$$N^{sub} = \bigsqcup_{1 \leq i \leq n} N_i^{sub} \sqcup N_0.$$

Therefore, the lemma holds for this case.

(b) The last inference ρ_{k_0} is generated by Rule $\mathbf{R}_{\exists}^{\perp}$ and is of the form

$$\frac{H_D \sqsubseteq N \sqcup \exists r.K,\ K \sqsubseteq \perp}{H_D \sqsubseteq N}.$$

Note that $H_D \sqsubseteq N \sqcup \exists r.K$ must be obtained by applying

- first an $\mathbf{R}_{\exists}^{+}$ inference of the form

$$\frac{H_D \sqsubseteq N_0 \sqcup P_0}{H_D \sqsubseteq N_0 \sqcup \exists r.D_0'}, \quad \neg P_0 \sqcup \exists r.D_0' \in cl_P(\mathcal{O});$$

- followed by m $\mathbf{R}_{\forall}$ inferences of the form

$$\frac{H_D \sqsubseteq (\bigsqcup_{i=0}^{j-1} N_i) \sqcup \exists r.K_{j-1},\ H_D \sqsubseteq N_j \sqcup P_j}{H_D \sqsubseteq (\bigsqcup_{i=0}^{j} N_i) \sqcup \exists r.(K_{j-1} \sqcap D_j')},$$

where $\neg P_j \sqcup \forall r.D'_j \in cl_P(\mathcal{O})$ and $K_j = D'_0 \sqcap \ldots \sqcap D'_j$ for $1 \leq j \leq m$. Moreover, we have

$$N = (\bigsqcup_{i=0}^{m} N_i), \;\; K = K_m.$$

By applying the inductive hypothesis on $K \sqsubseteq \bot$, we obtain

$$cl_P(\mathcal{O}) \vdash \bigsqcup_{i \in I^*} \neg D'_i, \text{ for some } I^* \subseteq \{0, \ldots m\},$$

and by applying the inductive hypothesis on $H_D \sqsubseteq N_j \sqcup P_j$, $j \in [0, m]$, we obtain

$$\begin{aligned} & cl_P(\mathcal{O}) \vdash H_D^j \sqcup N_j^{sub}, \\ \text{or} \;\; & cl_P(\mathcal{O}) \vdash H_D^j \sqcup N_j^{sub} \sqcup P_j, \end{aligned}$$

for some $H_D^j = \bigsqcup_{j \in I_i} \neg D_j$ with $I_i \subseteq \{1, \ldots, n\}$ and N_j^{sub} a disjunction of concepts from N_j,

We again distinguish both cases.

(i) If $cl_P(\mathcal{O}) \vdash H_D^j \sqcup N_j^{sub}$ for some $0 \leq j \leq m$, then $cl_P(\mathcal{O}) \vdash H_D^j \sqcup N_j^{sub}$ directly holds.

(ii) Otherwise, $cl_P(\mathcal{O}) \vdash H_D^j \sqcup N_j^{sub} \sqcup P_j$ for all $0 \leq j \leq m$. By applying the A-Rules for all P_j, $1 \leq j \leq m$ on

$$\begin{aligned} H_D^0 \sqcup N_0^{sub} \sqcup P_0, \;\; & \neg P_0 \sqcup \exists r.D'_0 \\ H_D^j \sqcup N_j^{sub} \sqcup P_j, \;\; & \neg P_j \sqcup \forall r.D'_j, \\ & (1 \leq j \leq m); \end{aligned}$$

and applying the r-Rule on

$$\begin{aligned} & \bigsqcup_{i \in I^*} \neg D'_i, \\ & H_D^0 \sqcup N_0^{sub} \sqcup \exists r.D'_0, \\ & H_D^j \sqcup N_j^{sub} \sqcup \forall r.D'_j, \\ & \quad (j \in I^* \cap \{1, \ldots, m\}), \end{aligned}$$

we obtain

$$cl_P(\mathcal{O}) \vdash H_D \sqcup N^{sub},$$

where

$$H_D = \bigsqcup_{j \in I^* \cup \{0\}} H_D^j, \text{ and}$$

$$N^{sub} = \bigsqcup_{j \in I^* \cup \{0\}} N_j^{sub}.$$

We obtain that the lemma also holds in this case. □

Using Theorem 124 and Lemma 125, we can now prove Lemma 105.

Lemma 126 *Let $\mathcal{S}$ be the set of axioms $\neg D_1 \sqcup \ldots \sqcup \neg D_n$, obtained by applying the rules in Fig. 5.4 exhaustively on $cl(\mathcal{O})$. Then, for all D_1, ..., $D_n \in \mathsf{N_D}$, we have $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ iff $\neg D_{i_1} \sqcup \ldots \sqcup \neg D_{i_k} \in \mathcal{S}$ for some subset $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$.*

Proof. For any definers $D_1, \ldots, D_n \in \mathsf{N_D}$, since $cl_P(\mathcal{O}) \equiv_{sig(cl(\mathcal{O}))} cl(\mathcal{O})$, we have $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ iff $cl_P(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$.

Note that we can exchange application order of a A-Rule on concept name $P \in \mathsf{N_P}$ and other rules without influence the final result. For instance: the following two rules that produces $C_1 \sqcup \ldots \sqcup C_n \sqcup \mathsf{Q}r.D$.

$$\text{(r-Rule):}\ \frac{C_1 \sqcup P \sqcup \exists r.D_1,\ \bigcup_{j=2}^{n}\{C_j \sqcup \forall r.D_j\},\ K_D}{C_1 \sqcup \ldots \sqcup C_n \sqcup P},$$

$$\text{(A-Rule on } P\text{):}\ \frac{C_1 \sqcup \ldots \sqcup C_n \sqcup P,\ \neg P \sqcup \mathsf{Q}r.D}{C_1 \sqcup \ldots \sqcup C_n \sqcup \mathsf{Q}r.D}$$

We also obtain $C_1 \sqcup \ldots \sqcup C_n \sqcup \mathsf{Q}r.D$ by following two rules, where a A-Rule on $P \in \mathsf{N_P}$ is applied first.

$$\text{(A-Rule on } P\text{):}\ \frac{C_1 \sqcup P \sqcup \exists r.D_1,\ \neg P \sqcup \mathsf{Q}r.D}{C_1 \sqcup \mathsf{Q}r.D \sqcup \exists r.D_1},$$

$$\text{(r-Rule):}\ \frac{C_1 \sqcup \mathsf{Q}r.D \sqcup \exists r.D_1,\ \bigcup_{j=2}^{n}\{C_j \sqcup \forall r.D_j\},\ K_D}{C_1 \sqcup \ldots \sqcup C_n \sqcup \mathsf{Q}r.D}$$

Therefore, when applying A-Rule and r-Rule on $cl_P(\mathcal{O})$, we could assume that A-Rules on concept names $P \in \mathsf{N_P}$ are applied first. Since applying A-Rules on concept names $P \in \mathsf{N_P}$ on $cl_P(\mathcal{O})$ produces exactly the axioms in $cl(\mathcal{O}) \setminus cl_P(\mathcal{O})$, we have $cl(\mathcal{O}) \vdash \neg D_1 \sqcup \ldots \sqcup \neg D_n$ iff $cl_P(\mathcal{O}) \vdash \neg D_1 \sqcup \ldots \sqcup \neg D_n$ for any definers $D_i \in \mathsf{N_D}$.

It is thus enough to show that for any definers $D_i \in \mathsf{N_D}$, $cl_P(\mathcal{O}) \vdash_P D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ iff $cl_P(\mathcal{O}) \vdash \neg D_{i_1} \sqcup \ldots \sqcup \neg D_{i_k}$ for some subset $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$.

We first prove the "$\Leftarrow$" direction. If $cl(\mathcal{O}) \vdash \neg D_{i_1} \sqcup \ldots \sqcup \neg D_{i_k}$, then we have $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$. Consequently, by Theorem 124, $cl_P(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$, and thus $cl_P(\mathcal{O}) \vdash_P D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$.

The "$\Rightarrow$" direction is a direct result of Lemma 125. □

Moreover, we have the following lemma that will be used in the proof of Theorem 104 in the next section.

Lemma 127 *Let $t \in \mathsf{N_R} \setminus \Sigma$ be a role name and D_1, ..., $D_n \in \mathsf{N_D}$ be definers s.t. $RI_\Sigma(\mathcal{O})$ contains a literal of the form $\mathsf{Q}t.D_1$, and for $2 \leq i \leq n$, $\forall t.D_i$ occurs in $RI_\Sigma(\mathcal{O})$. Then, $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ iff $RI_\Sigma(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$.*

Proof. If $RI_\Sigma(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$, then also $cl(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ because $RI_\Sigma(\mathcal{O})$ consists only of axioms from $cl(\mathcal{O})$ or axioms that have been derived from $cl(\mathcal{O})$. The other direction follows directly from the definition of $\mathcal{D}_\Sigma(\mathcal{O})$ in Definition 102. □

$$\textbf{r-Res}: \frac{C_1 \sqcup \exists r.D_1,\ C_2 \sqcup \forall r.D_2,\ \ldots, C_n \sqcup \forall r.D_n}{C_1 \sqcup \ldots \sqcup C_n},$$
$$\text{where } \mathcal{M} \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot, n \geq 1.$$

Figure 5.9: Rule for eliminating role name r.

Theorem 104

In order to prove Theorem 104, we make use of a result for role forgetting from [76], which describes the method for uniform interpolation used in FAME. In [76], the rule shown in Figure 5.9 is used for forgetting roles, which assumes the ontology to be in normal form, and uses a fixed ontology $\mathcal{M}$ as a side condition.

Fix a signature $\Sigma \subseteq \mathit{sig}(\mathcal{O})$. We denote by $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathcal{M})$ the ontology obtained by applying the following two operations on $\mathit{cl}(\mathcal{O})$:

1. apply **r-Res** exhaustively for all role names $r \in \mathit{sig}(\mathcal{O}) \setminus \Sigma$, and
2. remove all axioms that contain a role name $r \in \mathit{sig}(\mathcal{O}) \setminus \Sigma$.

By [76, Lemma 3], we have the following result.

Lemma 128 *For any ontology $\mathcal{O}$ and signature $\Sigma \subseteq \mathit{sig}(\mathcal{O})$, we have*

$$\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})) \equiv_{\Sigma \cup \mathit{sig}_\mathsf{c}(\mathcal{O})} \mathit{cl}(\mathcal{O}),$$

Let $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathcal{M})\big|_\Sigma$ be the sub ontology of $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathcal{M})$ that contains only those axioms $\alpha \in \mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathcal{M})$ that satisfy:

- if $L := \neg D$ is a literal of α, then for any definer $D' \in \mathit{sig}(\mathit{cl}(\mathcal{O}))$ such that $D \preceq_d D'$, we have $\mathit{Rol}\big(D', \mathit{cl}(\mathcal{O})\big) \subseteq \Sigma$.

To prove Theorem 104, we also need the following lemma.

Lemma 129 *For any ontology $\mathcal{O}$ and signature $\Sigma \subseteq \mathit{sig}(\mathcal{O})$, we have*

$$\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})) \equiv_{\Sigma \cup \mathit{sig}_\mathsf{c}(\mathcal{O})} \mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))\big|_\Sigma.$$

Proof. Recall that we assume that each definer occurs at most once positively in $\mathit{cl}(\mathcal{O})$. In particular, for each definer $D \in \mathit{sig}(cl(\mathcal{O}))$, there is at most one occurrence of a literal of the form $\mathsf{Q}r.D$ in $\mathit{cl}(\mathcal{O})$.

For any definer $D \in \mathit{sig}(\mathit{cl}(\mathcal{O}))$, if there exists $D' \in \mathit{sig}(\mathit{cl}(\mathcal{O}))$ such that $D \preceq_d D'$ and $\mathit{Rol}\big(D', \mathit{cl}(\mathcal{O})\big) = \{r_0\} \not\subseteq \Sigma$, then we can find a sequence of axioms in $\mathit{cl}(\mathcal{O})$ such as the following.

$$\begin{aligned}
\alpha_0 &: C_0 \sqcup \mathsf{Q}_0 r_0.D_0,\\
\alpha_1 &: \neg D_0 \sqcup C_1 \sqcup \mathsf{Q}_1 r_1.D_1,\\
&\ldots,\\
\alpha_n &: \neg D_n \sqcup C_{n+1} \sqcup \mathsf{Q}_n r_n.D_{n+1}.\\
\alpha_{n+1} &: \neg D_{n+1} \sqcup C_{n+2}.
\end{aligned}$$

where $D_0 = D', D_{n+1} = D$. Then, for every $1 \leq i \leq n$, $\mathsf{Q}_i r_i.D_i$ is the unique literal containing D_i positively, and $\mathsf{Q}_i r_i.D_i$ appears only in α_i.

Since $r_0 \notin \Sigma$, there is no axiom in $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))$ that contains a literal of the form $\mathsf{Q}_0 r_0.D_0$. Therefore, D_0 does not appear positively in $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))$. It is well known that, if a concept D_0 occurs only negatively in an ontology, we can preserve all entailments of axioms not using D_0 if we replace D_0 by $\bot$, which with our normal form means, we can delete all axioms with the literal $\neg D_0$ without losing any consequences in Σ (see also [77, Theorem 1]). Specifically, if we set $\Sigma_{Res} = \mathit{sig}(\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})))$, and let $\mathit{Res}^0_\Sigma(\mathcal{O})$ be the ontology obtained from $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))$ by removing all axioms that contain the literal $\neg D_0$, then we have

$$\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})) \equiv_{\Sigma_{Res} \setminus \{D_0\}} \mathit{Res}^0_\Sigma(cl(\mathcal{O})).$$

Note that if a literal $L_1 = \mathsf{Q}r.D$ always occurs in $\mathit{cl}(\mathcal{O})$ together with another literal L_2 which is of the form A or $\neg A$ (i.e., every axiom $\alpha \in \mathit{cl}(\mathcal{O})$ either contains both L_1 and L_2 or none of them), then L_1 also always appears together with L_2 in $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))$. Because the rule r-Res preserves all literals of the form A or $\neg A$, and since $\mathsf{Q}_1 r_1.D_1$ always appears with $\neg D_0$ in $\mathit{cl}(\mathcal{O})$, D_1 cannot appear positively in $\mathit{Res}^0_\Sigma(\mathit{cl}(\mathcal{O}))$, because we removed all occurrences of D_1 along with the occurrences of $\neg D_0$. Then, if $\mathit{Res}^1_\Sigma(\mathcal{O})$ is the ontology obtained by removing all axioms containing the literal $\neg D_1$ from $\mathit{Res}^0_\Sigma(\mathcal{O})$, we have

$$\mathit{Res}^0_\Sigma(\mathit{cl}(\mathcal{O})) \equiv_{\Sigma_{Res} \setminus \{D_1\}} \mathit{Res}^1_\Sigma(\mathit{cl}(\mathcal{O})).$$

Consequently, we have

$$\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), cl(\mathcal{O})) \equiv_{\Sigma_{Res} \setminus \{D_0, D_1\}} \mathit{Res}^1_\Sigma(\mathit{cl}(\mathcal{O})).$$

Repeat this process for all $D_2, \ldots, D_{n+1}$, and we have for $0 \leq i \leq n$ that

$$\mathit{Res}^i_\Sigma(\mathit{cl}(\mathcal{O})) \equiv_{\Sigma_{Res} \setminus \{D_{i+1}\}} \mathit{Res}^{i+1}_\Sigma(\mathit{cl}(\mathcal{O})) \text{ and}$$

$$\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), cl(\mathcal{O})) \equiv_{\Sigma_{Res} \setminus \{D_0, \ldots, D_{i+1}\}} \mathit{Res}^{i+1}_\Sigma(\mathit{cl}(\mathcal{O})),$$

where $\mathit{Res}^{i+1}_\Sigma(\mathcal{O})$ is the ontology obtained from $\mathit{Res}^i_\Sigma(\mathcal{O})$ by removing all axioms containing the literal $\neg D_{i+1}$. We conclude that removing all axioms that contain the literal $\neg D_{n+1} = \neg D$ preserves all logical consequences over $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))$ in the signature that excludes $\{D_0, \ldots, D_{n+1}\}$.

Finally, we have

$$\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})) \equiv_{\Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})} \mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O}))\big|_\Sigma$$

by repeating the process above for all D such that there exists $D' \in \mathit{sig}(\mathit{cl}(\mathcal{O}))$ such that $D \preceq_d D'$ and $\mathit{RoI}\big(D', cl(\mathcal{O})\big) = \{r_0\} \not\subseteq \Sigma$. □

We now have everything to prove Theorem 104.

Theorem 130 *$RI_\Sigma(\mathcal{O})$ is role isolated for Σ and we have $\mathcal{O} \equiv_{\Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})} RI_\Sigma(\mathcal{O})$.*

Proof. $RI_\Sigma(\mathcal{O})$ is role isolated by definition. We show that $\mathcal{O} \equiv_{\Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})} RI_\Sigma(\mathcal{O})$.

By Theorem 128 and Lemma 129, it is sufficient to show that

$$\mathit{Res}_\Sigma(cl(\mathcal{O}), cl(\mathcal{O}))\big|_\Sigma = \mathit{Res}_\Sigma(RI_\Sigma(\mathcal{O}), RI_\Sigma(\mathcal{O}))\big|_\Sigma. \tag{5.3}$$

This follows from the following observations.

1. If an axiom α contains the literal $\neg D$, any axiom obtained from α using **r-Res** also contains $\neg D$. Therefore, we have after Definition 102 defining $\mathit{cl}_\Sigma(\mathcal{O})$:

$$\mathit{Res}_\Sigma(cl(\mathcal{O}), cl(\mathcal{O}))\big|_\Sigma = \mathit{Res}_\Sigma(\mathit{cl}_\Sigma(\mathcal{O}), cl(\mathcal{O})).$$

2. Assume that $C_1 \sqcup \exists r.D_1$, $C_2 \sqcup \forall r.D_2$, ..., $C_n \sqcup \forall r.D_n \in \mathit{Res}_\Sigma(\mathit{cl}_\Sigma(\mathcal{O}), \mathit{cl}(\mathcal{O}))$ and $r \notin \Sigma$. Then, by Lemma 127, we have $\mathit{cl}(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$ iff $RI_\Sigma(\mathcal{O}) \models D_1 \sqcap \ldots \sqcap D_n \sqsubseteq \bot$. Therefore, for a given premise set $P = \{C_1 \sqcup \exists r.D_1,\ C_2 \sqcup \forall r.D_2,\ \ldots, C_n \sqcup \forall r.D_n\} \subseteq \mathit{Res}_\Sigma(\mathit{cl}_\Sigma(\mathcal{O}), \mathit{cl}(\mathcal{O}))$, an **r-Res** inference ρ is applicable on P with $\mathcal{M} = \mathit{cl}(\mathcal{O})$ iff ρ is applicable on P with $\mathcal{M} = RI_\Sigma(\mathcal{O})$. Consequently, we have

$$\mathit{Res}_\Sigma(\mathit{cl}_\Sigma(\mathcal{O}), cl(\mathcal{O})) = \mathit{Res}_\Sigma(\mathit{cl}_\Sigma(\mathcal{O}), RI_\Sigma(\mathcal{O})).$$

3. Since $RI_\Sigma(\mathcal{O}) = \mathit{cl}_\Sigma(\mathcal{O}) \cup \mathcal{D}_\Sigma(cl(\mathcal{O}))$ and every axiom in $\mathcal{D}_\Sigma(cl(\mathcal{O}))$ has a literal $\neg D$ with $Rol(D, cl(\mathcal{O})) \not\subseteq \Sigma$, we have

$$\begin{aligned} &\mathit{Res}_\Sigma(\mathit{cl}_\Sigma(\mathcal{O}), RI_\Sigma(\mathcal{O})) \\ &= \mathit{Res}_\Sigma(RI_\Sigma(\mathcal{O}), RI_\Sigma(\mathcal{O}))\big|_\Sigma. \end{aligned}$$

□

Theorem 109

Theorem 131 *If $\mathcal{O}$ is role isolated for Σ, then $\mathtt{rolE}_\Sigma(\mathcal{O})$ is a role-forgetting for $\mathcal{O}$ and Σ.*

Proof. Assume $\mathcal{O}$ is role isolated for Σ. Recall that by Lemma 128, for any ontology $\mathcal{O}$ and signature $\Sigma \subseteq \mathit{sig}(\mathcal{O})$, we have $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})) \equiv_{\Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})} \mathit{cl}(\mathcal{O})$. We note that, since $\mathcal{O}$ is in normal form, $\mathit{cl}(\mathcal{O})$ is obtained by replacing every literal $\mathsf{Q}r.A$ by some $\mathsf{Q}r.D$, where we also have the axioms $\neg D \sqcup A$. We observe furthermore that, since $\mathcal{O}$ is role isolated, the r-Rule is applicable in $\mathcal{O}$ iff the r-Res rule is applicable on the corresponding normalized axioms, excluding the last premise. As a consequence, we obtain that $\mathit{Res}_\Sigma(\mathit{cl}(\mathcal{O}), \mathit{cl}(\mathcal{O})) = \mathit{cl}(\mathtt{rolE}_\Sigma(\mathcal{O}))$. We obtain $\mathit{cl}(\mathtt{rolE}_\Sigma(\mathcal{O})) \equiv_{\Sigma \cup \mathsf{N_C}} \mathit{cl}(\mathcal{O})$. Since we also have $\mathit{cl}(\mathcal{O}) \equiv_{\Sigma \cup \mathsf{N_C}} \mathcal{O}$, we obtain that $\mathtt{rolE}_\Sigma(\mathcal{O})$ is a role forgetting of $\mathcal{O}$ for Σ. □

Theorem 114

Let Σ be a signature. For X being a role name or a concept, we define $\mathtt{copy}_\Sigma(X)$ by structural induction.

1. $\mathtt{copy}_\Sigma(A) = A$, if $A \in (\mathsf{N_C} \cap \Sigma) \cup \mathsf{N_D}$;
2. $\mathtt{copy}_\Sigma(r) = r$, if $r \in \mathsf{N_R} \cap \Sigma$;
3. $\mathtt{copy}_\Sigma(B) = \overline{B}$ if $B \in \mathit{sig}_\mathsf{C}(\mathcal{O}) \setminus \Sigma$, where $\overline{B}$ is fresh;
4. $\mathtt{copy}_\Sigma(r) = \overline{r}$ if $r \in \mathit{sig}_\mathsf{R}(\mathcal{O}) \setminus \Sigma$, where $\overline{r}$ is fresh;
5. $\mathtt{copy}_\Sigma(C_1 \sqcup C_2) = \mathtt{copy}_\Sigma(C_1) \sqcup \mathtt{copy}_\Sigma(C_2)$;
6. $\mathtt{copy}_\Sigma(C_1 \sqcap C_2) = \mathtt{copy}_\Sigma(C_1) \sqcap \mathtt{copy}_\Sigma(C_2)$;
7. $\mathtt{copy}_\Sigma(\mathsf{Q}r.C) = \mathsf{Q}\mathtt{copy}_\Sigma(r).\mathtt{copy}_\Sigma(C)$,

8. $\mathtt{copy}_\Sigma(\neg C) = \neg\mathtt{copy}_\Sigma(C)$.

We further define $\mathit{cl}^{ex}(\mathcal{O}) = RI_\Sigma(\mathcal{O}) \cup \mathit{cl}(\mathcal{O}_D)$, where

$$\mathcal{O}_D = \{D \equiv \mathtt{copy}_\Sigma(C_D) \mid D \in \mathit{sig}(\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))) \cap \mathsf{N_D}\}. \quad (5.4)$$

To prove Theorem 114, we need the following two lemmas.

Lemma 132 *We have* $\mathcal{O} \equiv_{\Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})} \mathit{cl}^{ex}(\mathcal{O})$.

Proof. Since $RI_\Sigma(\mathcal{O}) \subseteq \mathit{cl}^{ex}(\mathcal{O})$, and $\mathcal{O} \equiv_{\Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})} RI_\Sigma(\mathcal{O})$ (by Theorem 104), we have for every axiom α s.t. $\mathit{sig}(\alpha) \subseteq \Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})$ and $\mathcal{O} \models \alpha$, also $\mathit{cl}^{ex}(\mathcal{O}) \models \alpha$. We thus only need to show the other direction.

Let α be s.t. $\mathit{sig}(\alpha) \subseteq \Sigma \cup \mathit{sig}_\mathsf{C}(\mathcal{O})$, and assume $\mathit{cl}^{ex}(\mathcal{O}) \models \alpha$ but $\mathcal{O} \not\models \alpha$. Then, also $\mathit{cl}(\mathcal{O}) \not\models \alpha$, and there is a witnessing model $\mathcal{I}$ of $\mathit{cl}(\mathcal{O})$ s.t. $\mathcal{I} \not\models \alpha$. Based on $\mathcal{I}$, we construct a model $\mathcal{I}^{ex}$ of $\mathit{cl}^{ex}(\mathcal{O})$ and also show that $\mathcal{I}^{ex} \not\models \alpha$, and thus $\mathit{cl}^{ex}(\mathcal{O}) \not\models \alpha$. A contradiction!

$\mathcal{I}^{ex}$ is defined as follows.

1. $A^{\mathcal{I}^{ex}} = A^\mathcal{I}, r^{\mathcal{I}^{ex}} = r^\mathcal{I}$ for all $A, r \in \mathit{sig}(\mathcal{O}) \cap \mathit{sig}(cl^{ex}(\mathcal{O}))$;
2. $\overline{r}^{\mathcal{I}^{ex}} = r^{\mathcal{I}^{ex}}$, $\overline{B}^{\mathcal{I}^{ex}} = B^{\mathcal{I}^{ex}}$ for all introduced role names $\overline{r}$ and introduced concept names $\overline{B}$.
3. $D^{\mathcal{I}^{ex}} = (C_D)^{\mathcal{I}^{ex}}$ for every definer $D \in \mathit{sig}(\mathit{cl}^{ex}(\mathcal{O})) \cap \mathsf{N_D}$.

Since $\mathcal{I}$ is a model of $\mathcal{O}$, by the item 3 above and the definition of C_D, we know $\mathcal{I}^{ex}$ is compatible with all axioms in $RI_\Sigma(\mathcal{O})$ and $\mathcal{O}_D$. Therefore, $\mathcal{I}^{ex}$ is compatible with all axioms in $cl(\mathcal{O}_D)$ and thus a model of $cl^{ex}(\mathcal{O}_D)$. Moreover, because $A^{\mathcal{I}^{ex}} = A^\mathcal{I}$ and $r^{\mathcal{I}^{ex}} = r^\mathcal{I}$ for all $A, r \in \Sigma \cup \mathit{sig}(\mathcal{O})$, we have $\mathcal{I}^{ex} \models \mathit{cl}^{ex}(\mathcal{O})$ and $\mathcal{I}^{ex} \not\models \alpha$. A contradiction. □

Lemma 133 $\mathit{cl}^{ex}(\mathcal{O}) \equiv_\Sigma \mathit{gm}_\Sigma(\mathcal{O})$.

Proof. Assume $\Sigma^{ex} = \Sigma \cup \mathit{sig}(cl(\mathcal{O}_D))$. Then $\mathit{sig}_\mathsf{R}(cl(\mathcal{O}_D)) = \mathit{sig}_\mathsf{R}(\mathcal{O}_D) \subseteq \Sigma^{ex}$. Since $RI_\Sigma(\mathcal{O})$ is role isolated for Σ, we have $\mathit{cl}^{ex}(\mathcal{O}) = RI_\Sigma(\mathcal{O}) \cup \mathit{cl}(\mathcal{O}_D)$ is role isolated for Σ^{ex}. Moreover, we observe the following:

1. $\mathtt{rolE}_{\Sigma^{ex}}(\mathit{cl}^{ex}(\mathcal{O})) = \mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})) \cup cl(\mathcal{O}_D)$.
   This is because $\mathit{sig}_R(cl(\mathcal{O}_D)) \subseteq \Sigma^{ex}$ and $cl(\mathcal{O}_D)$ does not contain axioms of the form $\neg D_1 \sqcup \ldots \sqcup \neg D_n$, then the r-Rule is only applicable on $RI_\Sigma(\mathcal{O})$.
2. $\mathtt{conE}_{\Sigma^{ex}}(\mathtt{rolE}_{\Sigma^{ex}}(\mathit{cl}^{ex}(\mathcal{O}))) = \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \cup cl(\mathcal{O}_D)$.
   This is because $\mathit{sig}_C(cl(\mathcal{O}_D)) \subseteq \Sigma^{ex}$, then the A-rule is only applicable on $\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))$.

By Theorem 109 and 112, we have

$$\mathit{cl}^{ex}(\mathcal{O}) \equiv_{\Sigma^{ex}} \mathtt{rolE}_{\Sigma^{ex}}(\mathit{cl}^{ex}(\mathcal{O})),$$

$$\mathtt{rolE}_{\Sigma^{ex}}(cl^{ex}(\mathcal{O})) \equiv_{\Sigma^{ex}} \mathtt{conE}_{\Sigma^{ex}}(\mathtt{rolE}_{\Sigma^{ex}}(cl^{ex}(\mathcal{O}))).$$

It follows from these observations that

$$\begin{aligned} cl^{ex}(\mathcal{O}) &\equiv_{\Sigma^{ex}} \mathtt{conE}_{\Sigma^{ex}}(\mathtt{rolE}_{\Sigma^{ex}}(cl^{ex}(\mathcal{O}))) \\ &= \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \cup cl(\mathcal{O}_D) \\ &\equiv_\Sigma \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \cup \mathcal{O}_D. \end{aligned}$$

Moreover, we have

$$\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \cup \mathcal{O}_D \equiv_\Sigma gm_\Sigma(\mathcal{O})$$

because $gm_\Sigma(\mathcal{O})$ can be obtained by applying over $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \cup \mathcal{O}_D$ the following three operations:

1. Replace all occurrences of the definers D in $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ by $\mathtt{copy}_\Sigma(C_D)$ and remove tautologies.
2. Replace every newly introduced concept and role name $\overline{B}$ and $\overline{r}$ with B and r, respectively. Note that $\overline{B}, B, \overline{r}, r \notin \Sigma$ by the definition of $\mathtt{copy}_\Sigma$.
3. Apply exhaustively the translations $C_1 \sqsubseteq \neg C_2 \sqcup C_3 \Rightarrow C_1 \sqcap C_2 \sqsubseteq C_3$ and $C_1 \sqsubseteq \mathsf{Q} r.\neg C_2 \sqcap C_3 \Rightarrow C_1 \sqcap \overline{\mathsf{Q}} r.C_2 \sqsubseteq C_3$, where $\overline{\exists} = \forall$ and $\overline{\forall} = \exists$.

These operations produce a new ontology that is Σ-inseparable to the input ontology. In conclusion, since $\Sigma \subseteq \Sigma^{ex}$, we have $cl^{ex}(\mathcal{O}) \equiv_\Sigma \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \cup \mathcal{O}_D \equiv_\Sigma gm_\Sigma(\mathcal{O})$. This completes the proof. □

Theorem 134 *Let $gm_\Sigma(\mathcal{O})$ be the ontology obtained from $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ by*

- *deleting subsumed axioms,*
- *replacing each definer D by C_D, and*
- *exhaustively applying the two transformations below*

$$C_1 \sqsubseteq \neg C_2 \sqcup C_3 \;\Rightarrow\; C_1 \sqcap C_2 \sqsubseteq C_3$$
$$C_1 \sqsubseteq \mathsf{Q} r.\neg C_2 \sqcap C_3 \;\Rightarrow\; C_1 \sqcap \overline{\mathsf{Q}} r.C_2 \sqsubseteq C_3,$$

where $\overline{\exists} = \forall$ and $\overline{\forall} = \exists$.

Then, $gm_\Sigma(\mathcal{O})$ is a general module for $\mathcal{O}$ and Σ.

Proof. By Lemma 132 and 133, we have

$$\mathcal{O} \equiv_\Sigma cl^{ex}(\mathcal{O}) \equiv_\Sigma gm_\Sigma(\mathcal{O}),$$

which proves the theorem. □

Proposition 116

In the following, we use $|\mathcal{O}|$ to denote the number of axioms in $\mathcal{O}$. We then have $|\mathcal{O}| \leq \|\mathcal{O}\|$.

Proposition 135 *For any ontology $\mathcal{O}$ and signature Σ, we have $\|gm_\Sigma(\mathcal{O})\| \leq 2^{O(\|cl(\mathcal{O})\|)}$. On the other hand, there exists a family of ontologies $\mathcal{O}_n$ and signatures Σ_n s.t. $\|\mathcal{O}_n\|$ is polynomial in $n \geq 1$ and $\|gm_{\Sigma_n}(\mathcal{O}_n)\| = n \cdot 2^{O(\|cl(\mathcal{O}_n)\|)}$.*

Proof. We first show the upper bound. Our construction ensures that for every axiom $\alpha = \mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))$, we can find a sequence of axioms $\beta_1, \ldots, \beta_n \in \mathit{cl}_\Sigma(\mathcal{O})$ such that α is obtained from $\bigsqcup_{1 \leq i \leq n} \beta_i$ by removing all literals that contain a role $r \notin \Sigma$. Because there are at most exponentially many subsets of $\mathit{cl}_\Sigma(\mathcal{O})$, this limits the number of possible inferred axioms to exponentially many. We obtain

$$\begin{aligned} |\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))| &\leq 2^{|\mathit{cl}_\Sigma(\mathcal{O})|} \\ &\leq 2^{\|\mathit{cl}(\mathcal{O})\|} \\ \|\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))\| &\leq |\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))| \cdot \|cl(\mathcal{O})\| \\ &\leq 2^{\|\mathit{cl}(\mathcal{O})\|} \cdot \|\mathit{cl}(\mathcal{O})\|. \end{aligned}$$

Similarly, for every axiom $\gamma = \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$, we can find a sequence of axioms $\alpha_1, \ldots, \alpha_n \in \mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))$ such that γ is obtained from $\bigsqcup_{1 \leq i \leq n} \alpha_i$ by removing all literals of the form A or $\neg A$ with $A \notin \Sigma$. As shown above, each α_k is obtained from $\bigsqcup_{1 \leq i \leq n_k} \beta_i^k$, for some $\beta_1^k, \ldots, \beta_{n_k}^k \in \mathit{cl}_\Sigma(\mathcal{O})$, by removing all literals that contain a role name $r \notin \Sigma$. We obtain that γ is obtained from $\bigsqcup_{1 \leq k \leq n, 1 \leq i \leq n_k} \beta_i^k$ by removing all literals L such that (i) L contains a role name $r \notin \Sigma$, or (ii) L is of the form A or $\neg A$ with $A \notin \Sigma$. We obtain

$$\begin{aligned} |\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))| &\leq 2^{|\mathit{cl}_\Sigma(\mathcal{O})|} \\ &\leq 2^{\|\mathit{cl}(\mathcal{O})\|} \\ \|\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))\| &\leq |\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))| \cdot \|\mathit{cl}(\mathcal{O})\| \\ &\leq 2^{\|\mathit{cl}(\mathcal{O})\|} \cdot \|\mathit{cl}(\mathcal{O})\|. \end{aligned}$$

For every definer $D \in \mathit{sig}(\mathit{cl}(\mathcal{O}))$, we have $|C_D| < \|\mathit{cl}(\mathcal{O})\|$. Taking that the lengths of axioms in $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ are bound by $\|\mathit{cl}(\mathcal{O})\|$, we obtain that for every axiom $\alpha \in \mathit{gm}_\Sigma(\mathcal{O})$, we have $|\alpha| \leq \|\mathit{cl}(\mathcal{O})\|^2$. Consequently, we have

$$\begin{aligned} \|\mathit{gm}_\Sigma(\mathcal{O})\| &\leq |\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))| \cdot \|\mathit{cl}(\mathcal{O})\|^2 \\ &\leq 2^{\|\mathit{cl}(\mathcal{O})\|} \cdot \|\mathit{cl}(\mathcal{O})\|^2 \\ &< 2^{3 \cdot \|\mathit{cl}(\mathcal{O})\|}. \end{aligned}$$

Note that for the last step, we use that $n^2 < 2^{2n}$ for all integers $n \geq 0$. This establishes the upper bound.

We continue to show the lower bound. For any integer $n \geq 0$, we define the ontology $\mathcal{O}_n$ to contain the following axioms:

1. $Z_1 \sqcap Z_2 \sqcap \ldots \sqcap Z_n \sqsubseteq \bot$ (1 axiom of length $n+1$)
2. $X_i \sqcup Y_i \sqsubseteq Z_i$ for all $1 \leq i \leq n$ (n axioms of length 3),
3. $\top \sqsubseteq A_1 \sqcup \exists s.X_1$, $\top \sqsubseteq \overline{A_1} \sqcup \exists s.Y_1$ (2 axioms of length 4), and
4. $\top \sqsubseteq A_j \sqcup \forall s.X_j$, $\top \sqsubseteq \overline{A_j} \sqcup \forall s.X_j$ for $2 \leq j \leq n$ ($2n-2$ axioms of length 4).

As signature, we define $\Sigma_n = \{A_j\}_{j=1}^n \cup \{\overline{A_j}\}_{j=1}^n$.

Normalizing $\mathcal{O}_n$ introduces the definers D_i^X and D_i^Y with $C_{D_i^X} = X_i$ and $C_{D_i^Y} = Y_i$. In particular, this gives $2n$ additional axioms of length 2 each, so that we obtain $|\mathit{cl}(\mathcal{O}_n)| = 5n+1$ and

$$\begin{aligned} \|\mathit{cl}(\mathcal{O}_n)\| &= n + 1 + 3n + 2 \cdot 4 + (2n-2) \cdot 4 + 2n \cdot 2 \\ &= 16n + 1 \end{aligned}$$

We continue to compute the sizes of the other axiom sets computed.

- $\mathcal{D}_{\Sigma_n}(\mathit{cl}(\mathcal{O}_n))$ consists of axioms of the form:

$$\neg D_1^* \sqcup \ldots \sqcup \neg D_n^*, \quad \text{where for } 1 \leq i \leq n, \quad D_i^* \in \{D_i^X, D_i^Y\}.$$

  We have $|\mathcal{D}_{\Sigma_n}(\mathit{cl}(\mathcal{O}_n))| = 2^n$ and $\|\mathcal{D}_{\Sigma_n}(\mathit{cl}(\mathcal{O}_n))\| = n \cdot 2^n$.

- $\mathtt{rolE}_{\Sigma_n}(RI_{\Sigma^n}(\mathcal{O}_n))$ consists of the axioms in $\mathcal{D}_{\Sigma_n}(\mathit{cl}(\mathcal{O}^n))$ and axioms of the form:

$$\neg A_1^* \sqcup \ldots \sqcup \neg A_n^*, \quad \text{where for } 1 \leq i \leq n, \quad A_i^* \in \{A_i, \overline{A_i}\}.$$

  We obtain that $|\mathtt{rolE}_{\Sigma_n}(RI_{\Sigma_n}(\mathcal{O}_n))| = 2^{n+1}$ and $\|\mathtt{rolE}_{\Sigma_n}(RI_{\Sigma_n}(\mathcal{O}_n))\| = n \cdot 2^{n+1}$.

- Finally, since no definer occurs positively anymore, the axioms in $\mathcal{D}_{\Sigma_n}(\mathit{cl}(\mathcal{O}_n))$ are removed, so that we obtain $|\mathtt{conE}_{\Sigma_n}(\mathtt{rolE}_{\Sigma_n}(RI_{\Sigma_n}(\mathcal{O}_n)))| = 2^n$ and

$$\|\mathtt{conE}_{\Sigma_n}(\mathtt{rolE}_{\Sigma_n}(RI_{\Sigma_n}(\mathcal{O}_n)))\| = n \cdot 2^{n+1}.$$

As a final result, we obtain $|\mathit{gm}_{\Sigma^n}(\mathcal{O}_n)| = 2^n$ and $\|\mathit{gm}_{\Sigma^n}(\mathcal{O}_n)\| = n \cdot 2^{n+1}$.

To summarize, we defined a sequence of ontologies $\mathcal{O}_n$ with signatures Σ_n s.t. $\|\mathit{cl}(\mathcal{O}_n)\| = 16n + 1$ and $\|\mathit{gm}_{\Sigma^n}(\mathcal{O}_n)\| = n \cdot 2^{n+1}$. This establishes the second claim of the proposition. □

Proof of Propositions 118 and 117

To simplify the proofs of Propositions 118 and 117, we define the operator `defE` which applies the definer substitution explicitly.

Definition 136 *Let $\mathcal{O}$ be an ontology that contains definers D, for which C_D is defined. Then, the* definer substitution *on $\mathcal{O}$ is the ontology $\mathtt{defE}(\mathcal{O})$ that is obtained from $\mathcal{O}$ by replacing each definer D by the corresponding concept C_D.*

Since Proposition 118 relies on a simpler situation than Proposition 117, namely where the input is normalized, it is more convenient to start with it, before the more complex situation of Proposition 117.

Proposition 137 *Let $\mathcal{O}$ be an ontology in normal form and $\mathcal{M} = \mathit{gm}_\Sigma(\mathcal{O})$. Then, $\mathit{gm}_\Sigma(\mathcal{M})$=$\mathcal{M}$.*

Proof. We first make a general observation on the effect definers have when computing $\mathit{gm}_\Sigma(\mathcal{O})$ for normalized ontologies $\mathcal{O}$. First, since $\mathcal{O}$ is normalized, only concept names occur under role restrictions, which means that for every definer D introduced, we have $C_D \in \mathsf{N_C}$, and the only negative occurrence of D is in an axiom of the form $\neg D \sqcup C_D$. In case $C_D \notin \Sigma$, this means that previously eliminated concept names get reintroduced by the definer substitution when computing $\mathit{gm}_\Sigma(\mathcal{O})$, but they can only occur in two ways: 1) as negative literals in axioms of the form $\neg C_D \sqcup C$, or 2) in literals of the form $\mathsf{Q}r.C_D$. This also means that $\mathit{gm}_\Sigma(\mathcal{O})$ remains normalized. This means in particular that $\mathcal{M}$ contains no role name $r \notin \Sigma$, and all concept names $A \in \mathit{sig}_{\mathsf{C}}(\mathcal{M}) \setminus \Sigma$ occur either negatively or under role restrictions.

Let $RI_\Sigma(\mathcal{M})$ be the role isolated form for Σ and $\mathcal{M}$. We observe that $\mathit{sig}_{\mathsf{R}}(\mathcal{M}) \subseteq \Sigma$, since no role name outside of Σ is introduced to $\mathcal{M} = \mathit{gm}_\Sigma(\mathcal{O})$ when substituting definers D with their corresponding concepts C_D. We obtain that $RI_\Sigma(\mathcal{M}) = \mathit{cl}(\mathcal{M})$.

By definition of gm_Σ, we thus have

$$\mathit{gm}_\Sigma(\mathcal{M}) = \mathtt{defE}(\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(\mathit{cl}(\mathcal{M})))).$$

We show that $\mathit{gm}_\Sigma(\mathcal{M}) = \mathcal{M}$ using the following two results.

1. $\texttt{rolE}_\Sigma(\mathit{cl}(\mathcal{M})) = \mathit{cl}(\mathcal{M})$.

   Since $\mathit{sig}_\mathsf{R}(\mathcal{M}) \subseteq \Sigma$, there is no role name to be eliminated by the operator $\texttt{rolE}$. This means that $\texttt{rolE}_\Sigma(\mathit{cl}(\mathcal{M})) = \mathit{cl}(\mathcal{M})$.

2. $\texttt{defE}(\texttt{conE}_\Sigma(\mathit{cl}(\mathcal{M}))) = \mathcal{M}$.

   First, we show that $\mathcal{M} \subseteq \texttt{defE}(\texttt{conE}_\Sigma(\mathit{cl}(\mathcal{M})))$. Let $c \in \mathcal{M}$. Then, c is of the form

   $$\neg B_1 \sqcup \ldots \sqcup \neg B_k \sqcup \mathsf{Q}_1 r_1.A_1 \sqcup \ldots \sqcup \mathsf{Q} r_n.A_n \sqcup C_1 \tag{5.5}$$

   where (i) $B_i \in \mathit{sig}_\mathsf{C}(\mathcal{O}) \setminus \Sigma$ for $1 \leq i \leq k$, (ii) $A_i \in \mathit{sig}_\mathsf{C}(\mathcal{O}) \setminus \Sigma$ for $1 \leq i \leq n$, and (iii) $\mathit{sig}(C_1) \subseteq \Sigma$.

   We show that $c \in \texttt{defE}(\texttt{conE}_\Sigma(\mathit{cl}(\mathcal{M})))$. By the definition of $\mathcal{M} = \mathit{gm}_\Sigma(\mathcal{O})$, we have the following results.

   - $c \in \mathcal{M}$ must be obtained from an axiom $c_1 \in \texttt{conE}_\Sigma\Big(\texttt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{O})\big)\Big)$ by replacing every definer D by the concept C_D. Then c_1 is of the form

     $$c_1 = \neg D_1 \sqcup \ldots \neg D_k \sqcup \mathsf{Q}_1 r_1.D'_1 \sqcup \ldots \mathsf{Q}_n r_n.D'_n \sqcup C_1,$$

     where $C_{D_i} = B_i$ for $1 \leq i \leq k$, and $C_{D'_j} = A_j$ for $1 \leq j \leq n$.

     By our construction, since otherwise c_1 is deleted through $\texttt{conE}_\Sigma$, D_1, …, D_k must also occur positively in $\texttt{conE}_\Sigma\Big(\texttt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{O})\big)\Big)$. Consequently, there are k axioms

     $$c_2, \ldots, c_{k+1} \in \texttt{conE}_\Sigma\Big(\texttt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{O})\big)\Big)$$

     that are of the forms

     $$\begin{aligned} c_2 &= C_2 \sqcup \mathsf{Q}'_1 r'_1.D_1, \\ &\vdots \\ c_{k+1} &= C_{k+1} \sqcup \mathsf{Q}'_k r'_k.D_k. \end{aligned}$$

   - c_1 must be obtained by applying A-rules on $k+1$ axioms $c'_1, \ldots, c'_{k+1} \in \texttt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{O})\big)$ that are of the forms

     $$c'_1 = \neg B_1 \sqcup \ldots \sqcup \neg B_k \sqcup \mathsf{Q}_1 r_1.D'_1 \sqcup \ldots \mathsf{Q}_n r_n.D'_n \sqcup C_1, \tag{5.6}$$
     $$c'_2 = \neg D_1 \sqcup B_1 \tag{5.7}$$
     $$\vdots \tag{5.8}$$
     $$c'_{k+1} = \neg D_k \sqcup B_k. \tag{5.9}$$

     Modulo renaming of definers,[1] we have

     (a) $c'_1 \in cl(\mathcal{M})$ by normalizing c;

[1] Note that it is in principle possible that $\mathit{cl}(\mathcal{M})$ contains more definers than $\mathit{cl}(\mathcal{O})$, since occurrences of role restrictions can be multiplied. However, this does not affect the following argument, since all definers get replaced by the same concept names again.

(b) $c'_2, \ldots, c'_{k+1} \in cl(\mathcal{M})$ by normalizing the axioms in

$$\texttt{defE}(\{c_2, \ldots, c_{k+1}\}) \subseteq \mathcal{M}.$$

Therefore, we can assume $\{c'_1, c'_2, \ldots, c'_{k+1}\} \subseteq \mathit{cl}(\mathcal{M})$. We obtain $c \in \mathit{gm}_\Sigma(\mathcal{M})$ by repeating the process of generating $c \in \mathit{gm}_\Sigma(\mathcal{O})$ from

$$c'_1, c'_2, \ldots, c'_{k+1}.$$

As a result, we obtain that $\mathcal{M} \subseteq \texttt{defE}(\texttt{conE}_\Sigma(cl(\mathcal{M})))$.

Furthermore, we have $\texttt{defE}(\texttt{conE}_\Sigma(\mathit{cl}(\mathcal{M}))) \subseteq \mathcal{M}$, since all the axioms in $\mathit{cl}(\mathcal{M})$ are of the forms (5.6) – (5.9), and we cannot obtain axioms other than c in (5.5) after applying the operators conE and defE. Consequently, we have $\mathcal{M} = \texttt{defE}(\texttt{conE}_\Sigma(\mathit{cl}(\mathcal{M})))$.

From 1 and 2, it follows that $\mathcal{M} = \mathit{gm}_\Sigma(\mathcal{M})$. □

Proposition 138 *Let $(\mathcal{M}_i)_{i\geq 1}$ be the sequence of ontologies defined by (i) $\mathcal{M}_1 = \mathit{gm}_\Sigma(\mathcal{O})$ and (ii) $\mathcal{M}_{i+1} = \mathit{gm}_\Sigma(\mathcal{M}_i)$ for $i \geq 1$. Then, we have*

$$\mathcal{M}_i \subseteq \mathcal{M}_{i+1} \text{ for } i \geq 1.$$

Moreover, there exists $i_0 \geq 0$ s.t. $\mathcal{M}^k = \mathcal{M}^{i_0}$ for all $k \geq i_0$.

Proof. Assume $\mathcal{M}_0 = \mathcal{O}$. We first show that every axiom $c \in \mathcal{M}_i = \mathit{gm}_\Sigma(\mathcal{M}_{i-1})$ is also in $\mathcal{M}_{i+1}$ for all $i \geq 1$.

Any axiom $c \in \mathit{gm}_\Sigma(\mathcal{M}_{i-1})$ is obtained from some axiom

$$c_d \in \texttt{conE}_\Sigma\Big(\texttt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{M}_{i-1})\big)\Big)$$

by replacing every definer D by the corresponding concept C_D. Then, $\mathit{sig}(c_d) \subseteq \Sigma \cup \mathsf{N_D}$ by the definitions of $\texttt{rolE}_\Sigma$ and $\texttt{conE}_\Sigma$. There are two different cases.

1. c_d does not contain negative definers. Then, c_d and c are of the forms

$$c_d = C_1 \sqcup \mathsf{Q}_1 r_1.D_1 \sqcup \ldots \sqcup \mathsf{Q}_n r_n.D_n \tag{5.10}$$
$$c = C_1 \sqcup Q_1 r_1.C_{D_1} \sqcup \ldots \sqcup \mathsf{Q}_n r_n.C_{D_n}. \tag{5.11}$$

where, $\mathit{sig}(C_1) \subseteq \Sigma$, $r_i \in \Sigma$ and $\mathsf{Q}_i \in \{\exists, \forall\}$ for $1 \leq i \leq n$.

Then, $c \in \mathit{gm}_\Sigma(\mathcal{M}_i)$ because of the following observations, which hold modulo renaming of definers.

- $c_d \in \texttt{rolE}_\Sigma(RI_\Sigma(\mathcal{M}_i))$ because $c_d \in RI_\Sigma(\mathcal{M}_i)$ and $\mathit{sig}(c_d) \subseteq \Sigma \cup \mathsf{N_D}$,
- $c_d \in \texttt{conE}_\Sigma(\texttt{rolE}_\Sigma(RI_\Sigma(\mathcal{M}_i)))$ because $c_d \in \texttt{rolE}_\Sigma(RI_\Sigma(\mathcal{M}_i))$, $\mathit{sig}(c_d) \subseteq \Sigma \cup \mathsf{N_D}$ and c_d does not contain negative definers, and
- $\texttt{defE}(\{c_d\}) = \{c\}$ by definition.

2. c_d contains negative definers. Then, c_d and c are respectively of the forms

$$c_d = \neg D_1 \sqcup \ldots \neg D_n \sqcup C'_d \tag{5.12}$$
$$c = \neg C_{D_1} \sqcup \ldots \neg C_{D_n} \sqcup C', \tag{5.13}$$

where C'_d, C' do not contain negative definers, and C'_d and C' are of the forms as in (5.10) and (5.11).

In this case, normalizing c in $\mathcal{M}_i$ produces axioms in $cl(\mathcal{M}_i)$ that are different from c_d. However, we can still show that $c \in \mathcal{M}_{i+1} = gm_\Sigma(\mathcal{M}_i)$. We only consider the case where $n = 1$, that is, there is only one negative definer in c_d. The case for $n > 1$ is shown by repeating the argument step-wise for each definer. We distinguish 4 possible cases based on the syntactical shape of C_{D_1}.

(a) C_{D_1} is of the form A or $\neg A$. This case is proved similarly as for Proposition 118, where we considered the case of normalized ontologies for which C_{D_1} is always of the form $A \in \mathsf{N_C}$.

(b) $C_{D_1} = \mathsf{Q}r.C_1$. We consider only the case where $\mathsf{Q} = \exists$. The other direction is shown in a similar way by just switching the quantifiers. With $\mathsf{Q} = \exists$, we have

$$c_d = \neg D_1 \sqcup C'_d, \qquad c = \neg(\exists r.C_1) \sqcup C'.$$

Normalizing c, we obtain

$$c'_1 = \forall r.D'_2 \sqcup C'_d \in cl(\mathcal{M}_i),$$

where $C_{D'_2} = \neg C_1$. If $r \in \Sigma$, then we have $c \in gm_\Sigma(\mathcal{M}_i)$ as in Case 1. Assume $r \notin \Sigma$. We then make the following observations.

- There exists an axiom

$$c_1 = \neg D_1 \sqcup \exists r.D'_3 \in RI_\Sigma(\mathcal{M}_{i-1})$$

with $C_{D'_3} = C_1$. Here, c_1 is introduced when normalizing the literal $C_{D_1} = \exists r.C_1$.

- There exists an axiom

$$c_2 = C \sqcup \mathsf{Q}r.D_1 \in \mathtt{conE}_\Sigma\Big(\mathtt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{M}_{i-1})\big)\Big)$$

for some C, Q, r because D_1 must also occur positively in $\mathtt{conE}_\Sigma\Big(\mathtt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{M}_{i-1})\big)\Big)$. (Otherwise, c_d would be deleted by conE.)

We have $c_1 \in cl(\mathcal{M}_i)$ (modulo renaming of definers) by normalizing the axiom in $\mathtt{defE}(\{c_2\}) \subseteq \mathcal{M}_i$. Furthermore, we have

$$cl(\mathcal{M}_i) \models D'_2 \sqcap D'_3 \sqsubseteq \bot.$$

This allows us to make the following further observations.

- $c_d \in \mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{M}_i))$ due to $c_d \in \mathtt{conE}_\Sigma\Big(\mathtt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{M}_{i-1})\big)\Big)$ and the following inference with the r-Rule.

$$\frac{\forall r.D'_2 \sqcup C'_d,\ \neg D_1 \sqcup \exists r.D'_3,\ \neg D'_2 \sqcup \neg D'_3}{\neg D_1 \sqcup C'_d}.$$

- $c_d \in \mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{M}_i)))$ since $c_d \in \mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{M}_i))$ and $sig(c_d) \subseteq \Sigma \cup \mathsf{N}_D$;
- $\{c\} = \mathtt{defE}(\{c_d\})$ as follows directly from (5.12) and (5.13).

We obtain that $c \in gm_\Sigma(\mathcal{M}_i, \Sigma) = \mathcal{M}_{i+1}$.

(c) $C_{D_1} = L_1 \sqcap \ldots \sqcap L_n \sqcap \mathsf{Q}_1 r_1.C_1 \sqcap \ldots \sqcap \mathsf{Q}_m r_m.C_m$, where for all $1 \leq i \leq n$, $L_i = A_i$ or $L_i = \neg A_i$ for some $A_i \in \mathsf{N_C}$ and all $1 \leq i \leq n$. Then, normalizing c produces following axiom in $cl(\mathcal{M}_i)$:

$$\neg L_1 \sqcup \ldots \sqcup \neg L_n \sqcup \mathsf{Q}_1^* r_1.D_1' \sqcup \ldots \sqcup \mathsf{Q}_m^* r_m.D_m' \sqcup C',$$

where $C_{D_i'} = \neg C_i$, $\{\mathsf{Q}_i, \mathsf{Q}_i^*\} = \{\forall, \exists\}$, $1 \leq i \leq m$. As in Case (b), we have

- $\neg D_1 \sqcup L_1, \ldots, \neg D_1 \sqcup L_n, \neg D_1 \sqcup \mathsf{Q}_1 r_1.D_1', \ldots, \neg D_1 \sqcup \mathsf{Q}_m r_m.D_m' \in RI_\Sigma(\mathcal{M}_i)$;
- $C \sqcup \mathsf{Q} r.D_1 \in \mathtt{conE}_\Sigma\Big(\mathtt{rolE}_\Sigma\big(RI_\Sigma(\mathcal{M}_{i-1})\big)\Big)$ for some C, Q, r because otherwise c_d is removed in Step 3 using conE.

We obtain that $c \in gm_\Sigma(\mathcal{M}_i)$ using the argument from Case (a) for every L_i and from Case (b) for every D_i.

(d) For the general case, we have $C_{D_1} = \overline{C_1} \sqcup \ldots \sqcup \overline{C_n}$, where each $\overline{C_i}$ is as C_D in Case (c). In this case, we rewrite c as n different axioms.

$$\overline{c_1} = \neg\overline{C_1} \sqcup C', \qquad \ldots, \qquad \overline{c_n} = \neg\overline{C_n} \sqcup C'.$$

For each $1 \leq i \leq n$, we then have $\overline{c_i} \in gm_\Sigma(\mathcal{M}_i)$ as in Case (c).

We obtain in each case that $c \in \mathcal{M}_{i+1}$. As a consequence, we have $\mathcal{M}_i \subseteq \mathcal{M}_{i+1}$, for each $i \geq 1$.

It remains to show that there exists some $i_0 \geq 0$ such that $\mathcal{M}_{i_0} = \mathcal{M}_{i_0+1}$, since all axioms in $\mathcal{M}_i$ are axioms consisting of literals of the from

$$A,\ \neg A,\ \neg C_D,\ Qr.C_D,$$

where C_D is a sub-concept of a concept in $\mathcal{O}$. There exist only finitely many such literals and thus only finitely many such axioms. Consequently, the chain $\mathcal{M}_0 \subseteq \mathcal{M}_1 \subseteq \ldots$ must reach a fixpoint after finitely many steps. □

Theorem 119

Recall the operator Op1, Op2 introduced in Section 5.4.3. For simplicity,

- let $\mathcal{M}_1$ be the ontology obtained by applying the operator Op1, and
- let $\mathcal{M}_2$ be the ontology obtained by applying the operator Op2 on $\mathcal{M}_2$

We prove the correctness of the two operations one after the other. For the first operation, we use a result from [68], which inspired our optimization, and which uses the inference rules shown in Figure 5.10.

Lemma 139 $conE_\Sigma(rolE_\Sigma(RI_\Sigma(\mathcal{O}))) \equiv_\Sigma \mathcal{M}_1$.

Proof. Let $\mathcal{O}_{red}$, $\mathcal{M}_{red}$ be the ontologies obtained by applying the rules in Figure 5.10 on $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$ and $\mathcal{M}_1$ respectively. By [68, Theorem 5], we have

$$\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \equiv_\Sigma \mathcal{O}_{red},$$
$$\mathcal{M}_1 \equiv_\Sigma \mathcal{M}_{red}.$$

Role Propagation (RP):

$$\frac{\bigcup_{j=1}^{m}\{P_j \sqcup C_j\}, \qquad E_0 \sqcup \mathsf{Q}r.D_0, \qquad \bigcup_{i=1}^{k}\{E_i \sqcup \forall r.D_i\}}{(\bigsqcup_{i=0}^{n} E_i) \sqcup \mathsf{Q}r.(\bigsqcap_{j=0}^{m} C_j)},$$

where $P_0 = \bigsqcup_{i=0}^{n} \neg D_i$, for $j > 0$, P_j is a sub-concept of P_0, $\mathsf{Q} \in \{\forall, \exists\}$, and C_0 and C_j do not contain a definer.

Reduction (Red):

$$\frac{\mathcal{O} \cup \{\neg D_1 \sqcup \ldots \sqcup \neg D_n \sqcup C\}}{\mathcal{O}},$$

where C is a general concept expression that does not contain a negative definer and $D_1, \ldots, D_n$ are definer symbols. The RP rule applies before this rule if $\neg D_1 \sqcup \ldots \sqcup \neg D_n$ takes the form of P_0 in the RP rule.

Figure 5.10: Rules RP and Red.

Because the rule conD-Elim used for computing $\mathcal{M}_1$ is a special case of the RP rule, we have $\mathcal{O}_{red} = \mathcal{M}_{red}$. As a consequence, we obtain

$$\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O}))) \equiv_\Sigma \mathcal{M}_1.$$

□

Lemma 140 $\mathcal{M}_1 \equiv_\Sigma \mathcal{M}_2$.

Proof. It is shown in [77, Theorem1], as an easy consequence of Ackermann's lemma, that if the set of all negative occurrencess of a definer D in $\mathcal{O}$ is of the form $\{\neg D \sqcup C_j \mid 1 \leq j \leq n\}$, then we can replace all positive occurrences of D in $\mathcal{O}$ by $\bigsqcap_{j=1}^{n} C_j$ without losing logical consequences that do not involve D. This is exactly what Op2 does. □

Theorem 141 *Let $gm^*_\Sigma(\mathcal{O})$ be the ontology obtained by:*

- *successive application of Op1 and Op2 over $\mathtt{conE}_\Sigma(\mathtt{rolE}_\Sigma(RI_\Sigma(\mathcal{O})))$, and then*
- *application of the steps described in Theorem 114.*

Then, $gm^*_\Sigma(\mathcal{O})$ is a general module for $\mathcal{O}$ and Σ.

Proof. This can be shown almost in the same way as in the proof for Theorem 114. The only difference is that we change the definition of O_D in Equation (5.4) to

$$\mathcal{O}_D = \{D \equiv \mathtt{copy}_\Sigma(C_D) \mid D \in \mathit{sig}(\mathcal{M}_2) \cap \mathsf{N_D}\}.$$

The reason is that we do not need to consider definers eliminated by the rules con-Elim and D-prop anymore. □

Theorem 122

Theorem 142 *Let us define* $dm_\Sigma(\mathcal{O})$ *by*

$$dm_\Sigma(\mathcal{O}) = \{\alpha \in \mathcal{O} \mid \alpha R^* \beta \text{ for some } \beta \in gm^*_\Sigma(\mathcal{O})\}.$$

Then, $dm_\Sigma(\mathcal{O})$ *is a deductive module for* $\mathcal{O}$ *and* Σ.

Proof. Set $\mathcal{M} = dm_\Sigma(\mathcal{O})$. We note that $gm_\Sigma(\mathcal{M}) = gm_\Sigma(\mathcal{O})$, since $\mathcal{M}$ contains exactly the set of axioms that are used to compute $gm_\Sigma(\mathcal{O})$. By Theorem 114, we have $\mathcal{M} \equiv_\Sigma gm_\Sigma(\mathcal{M})$ and $\mathcal{O} \equiv_\Sigma gm_\Sigma(\mathcal{O})$. Putting these observations together, we obtain $\mathcal{M} \equiv_\Sigma \mathcal{O}$. Since also $\mathcal{M} \subseteq \mathcal{O}$, $\mathcal{M}$ is a deductive module of $\mathcal{O}$ for Σ. □

6 - Conclusions and Future Work

In this thesis, we develop three methods for extracting knowledge from Description Logics $\mathcal{EL}^+$, $\mathcal{EL}$, and $\mathcal{ALC}$, respectively. An overview of our contributions is summarized in Figure 6.1. Our methods have been developed to solve two different scenarios. (1) In the first scenario, the users are interested in finding an explanation of a conclusion based on a given ontology. To this end, we developed a method `minH` for computing justifications for $\mathcal{EL}^+$-ontologies; (2) In the second scenario, the users want to analyze what the ontology states about a given set of concept and role names. To this end, we proposed to compute ontology modules. For $\mathcal{EL}$-ontologies, we introduced `ForMod` that allows one to compute two kinds of deductive modules: pseudo-minimal modules and complete modules. This method is also a contribution to computing minimal modules, as (recall Theorem 72) pseudo-minimal modules are indeed minimal modules when the corresponding forest is finite. For $\mathcal{ALC}$-ontologies, we developed GEMO that allows one to compute general modules, a notion that subsumes uniform interpolant and deductive modules. The method GEMO has been extended for the computation of uniform interpolant and deductive modules.

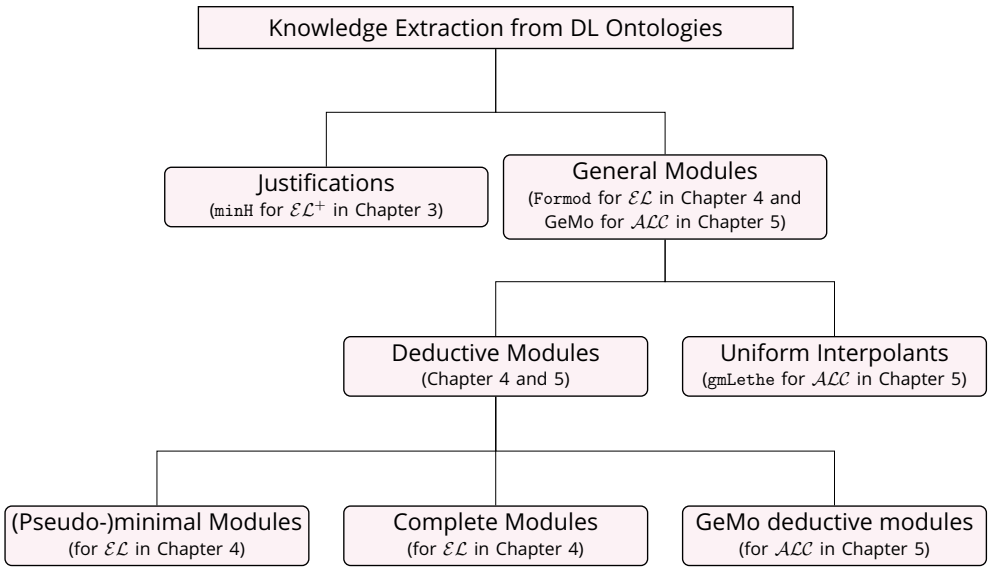


Figure 6.1: Overview of the thesis work

Justifications In Chapter 3, we introduce and investigate a new set of sound and complete inference rules based on a hypergraph representation of $\mathcal{EL}^+$-ontologies. We design the algorithm `minH` that leverages these inference rules to compute all justifications for a given conclusion. The key to the performance of our method is

that regular paths are used as elementary components of H-paths and this leads to reducing the size of complete sets because (1) the rules are more compact than the standard ones, (2) redundant inferences are captured and eliminated by regular paths (see Section 3.4.2). The efficiency of the algorithm `minH` has been validated by our experiments showing that it outperforms PULi in most cases.

There are still many possible extensions and applications of the hypergraph approach. For instance, to get even more compact inference rules, we could extend the notion of regular path to a more general one that will encapsulate the inference rule $\mathcal{H}_2$ (recall Table 3.2 on page 34) in the same way as regular paths encapsulated in H-rules. Moreover, we could try to apply our approach for other tasks like classification and for computing logical differences [54].

Deductive modules In chapter 4, we presented two deductive modules for $\mathcal{EL}$-ontologies: pseudo-minimal modules and complete modules, and we developed a SAT-based algorithm `ForMod` to compute those modules. This method is based on a novel notion of the forest, which enables to capture all the entailments over a given signature. The experiments on real-world ontologies validate the efficiency of our proposal as well as the quality of our deductive modules compared to `Zoom` [16].

As the next step, we plan to investigate how to generalize our ideas to more expressive ontologies such as $\mathcal{EL}^+$ and $\mathcal{ALC}$. Also, we are interested in investigating such ideas on module notions that are not necessarily deductive modules (e.g., semantic modules [42]).

General modules In chapter 5, we present the first tool GEMO for computing general modules for $\mathcal{ALC}$ ontologies, which can also be used for computing deductive modules and uniform interpolants. Our method is based on a new role isolation process that enables efficient role forgetting and an easy definer elimination. Our method's efficiency and the quality are validated by experiments on real-world ontologies, outperforming LETHE and other methods.

In the future, we are interested in optimizing the concept elimination step to obtain more concise general modules. Also, we plan to generalize our ideas to more expressive ontologies.

7 - Acknowledgement

First and foremost, I want to express my sincere thanks to my supervisor, Professor Nicole Bidoit Tollu. She has been incredibly kind and helpful throughout my Ph.D. studies, providing great assistance with my writing and presentations. She almost helps me review every word in my papers, slides, and thesis, and I am truly grateful for her support.

I also want to give a big thank you to my co-supervisor, Yue MA. She was not only a good supervisor but also a wonderful friend, who has been a tremendous help in my life and career in France. She helps me apply for a bank card, handle conference reimbursements, and many other things. She has also been a warm and welcoming person, inviting me to have dinner at her home many times. I greatly appreciate her generosity and for giving me the opportunity to study in France.

I would also like to thank Professor Patrick Koopmann, who is a wonderful researcher and a kind person. It is a good experience to be able to collaborate with him on the work of computing general modules. He provided valuable assistance with implementation, proof-checking, and revising the paper.

A special thanks go to the members of my thesis jury, especially Professor Meghyn Bienvenu. Their insightful comments and suggestions greatly helped me improve my thesis.

Also, I want to express my gratitude to all my friends and families for their support. I thank Yuting, Junjie, Tianjiao, Guanghui, and all the other friends for all the happy times we spent together, such as climbing mountains, having dinner, preparing food, and skiing.

Finally, I want to say thanks to my girlfriend Yue Su. Since I don't speak French, she has been supporting me a lot in my daily life, such as helping me update visas and find houses. Moreover, she was the one who helped me prepare a wonderful celebration after my defense. I would like to thank her wholeheartedly for her companionship, which made the challenging times during the Covid-19 pandemic much easier and happier. :-)

Bibliography

[1] C. Alrabbaa, F. Baader, S. Borgwardt, P. Koopmann, and A. Kovtunova. Finding small proofs for description logic entailments: Theory and practice. In E. Albert and L. Kovács, editors, *LPAR 2020: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 73 of *EPiC Series in Computing*, pages 32–67. EasyChair, 2020.

[2] M. F. Arif, C. Mencía, A. Ignatiev, N. Manthey, R. Peñaloza, and J. Marques-Silva. Beacon: An efficient SAT-based tool for debugging $\mathcal{EL}^+$ ontologies. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 521–530, 2016.

[3] M. F. Arif, C. Mencía, and J. Marques-Silva. Efficient axiom pinpointing with EL2MCS. In *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*, pages 225–233, 2015.

[4] M. F. Arif, C. Mencía, and J. Marques-Silva. Efficient MUS enumeration of horn formulae with applications to axiom pinpointing. In *International Conference on Theory and Applications of Satisfiability Testing*, pages 324–342, 2015.

[5] G. Ausiello and L. Laura. Directed hypergraphs: Introduction and fundamental algorithms—a survey. *Theoretical Computer Science*, 658:293–306, 2017.

[6] F. Baader, S. Brandt, and C. Lutz. Pushing the EL envelope. In *International Joint Conference on Artificial Intelligence*, pages 364–369, 2005.

[7] F. Baader and F. Distel. Exploring finite models in the description logic. In S. Ferré and S. Rudolph, editors, *Formal Concept Analysis, 7th International Conference, ICFCA 2009, Darmstadt, Germany, May 21-24, 2009, Proceedings*, volume 5548 of *Lecture Notes in Computer Science*, pages 146–161. Springer, 2009.

[8] F. Baader, B. Ganter, B. Sertkaya, and U. Sattler. Completing description logic knowledge bases using formal concept analysis. In M. M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 230–235, 2007.

[9] F. Baader, I. Horrocks, C. Lutz, and U. Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[10] F. Baader and R. Peñaloza. Axiom pinpointing in general tableaux. *J. Log. Comput.*, 20(1):5–34, 2010.

[11] D. Borchmann and F. Distel. Mining of el-gcis. In M. Spiliopoulou, H. Wang, D. J. Cook, J. Pei, W. Wang, O. R. Zaïane, and X. Wu, editors, *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, Vancouver, BC, Canada, December 11, 2011*, pages 1083–1090. IEEE Computer Society, 2011.

[12] D. Calvanese and G. D. Giacomo. Expressive description logics. In F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 178–218. Cambridge University Press, 2003.

[13] J. Chen. *Knowledge Extraction from Description Logic Terminologies*. PhD thesis, Université Paris-Saclay (ComUE), 2018.

[14] J. Chen, G. Alghamdi, R. A. Schmidt, D. Walther, and Y. Gao. Ontology extraction for large ontologies via modularity and forgetting. In *International Conference on Knowledge Capture*, pages 45–52, 2019.

[15] J. Chen, M. Ludwig, Y. Ma, and D. Walther. Evaluation of extraction techniques for ontology excerpts. In *Description Logics*, volume 1193 of *CEUR Workshop Proceedings*, pages 471–482, 2014.

[16] J. Chen, M. Ludwig, Y. Ma, and D. Walther. Zooming in on ontologies: Minimal modules and best excerpts. In *16th International Semantic Web Conference*, pages 173–189, 2017.

[17] P. Cimiano. *Ontology learning and population from text - algorithms, evaluation and applications*. Springer, 2006.

[18] W. Del-Pinto and R. A. Schmidt. ABox abduction via forgetting in $\mathcal{ALC}$. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, pages 2768–2775. AAAI Press, 2019.

[19] K. Donnelly et al. SNOMED-CT: The advanced terminology and coding system for eHealth. *Studies in health technology and informatics*, 121:279, 2006.

[20] S. Feng, M. Ludwig, and D. Walther. Foundations for the logical difference of el-tboxes. In G. Gottlob, G. Sutcliffe, and A. Voronkov, editors, *Global Conference on Artificial Intelligence, GCAI 2015, Tbilisi, Georgia, October 16-19, 2015*, volume 36 of *EPiC Series in Computing*, pages 93–112. EasyChair, 2015.

[21] L. Galárraga, C. Teflioudi, K. Hose, and F. M. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6):707–730, 2015.

[22] G. Gallo, G. Longo, S. Pallottino, and S. Nguyen. Directed hypergraphs and applications. *Discrete applied mathematics*, 42(2-3):177–201, 1993.

[23] W. Gatens, B. Konev, and F. Wolter. Lower and upper approximations for depleting modules of description logic ontologies. In T. Schaub, G. Friedrich, and B. O'Sullivan, editors, *ECAI 2014 - 21st European Conference on Artificial Intelligence*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 345–350. IOS Press, 2014.

[24] W. Gatens, B. Konev, and F. Wolter. Lower and upper approximations for depleting modules of description logic ontologies. In *European Conference on Artificial Intelligence*, pages 345–350, 2014.

[25] B. Glimm, I. Horrocks, B. Motik, G. Stoilos, and Z. Wang. Hermit: An OWL 2 reasoner. *J. Autom. Reason.*, 53(3):245–269, 2014.

[26] B. C. Grau. Privacy in ontology-based information systems: A pending matter. *Semantic Web*, 1(1-2):137–141, 2010.

[27] B. C. Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *Journal of Artificial Intelligence Research*, 31:273–318, 2008.

[28] B. C. Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Modular reuse of ontologies: Theory and practice. *J. Artif. Intell. Res.*, 31:273–318, 2008.

[29] M. Horridge and S. Bechhofer. The OWL API: a java API for OWL ontologies. *Semantic Web*, 2(1):11–21, 2011.

[30] A. Hou. A theory of measurement in diagnosis from first principles. *Artif. Intell.*, 65(2):281–328, 1994.

[31] A. Ignatiev, J. Marques-Silva, C. Mencıa, and R. Peñaloza. Debugging EL ontologies through Horn MUS enumeration.

[32] E. Jiménez-Ruiz, B. C. Grau, U. Sattler, T. Schneider, and R. Berlanga. Safe and economic re-use of ontologies: A logic-based methodology and tool support. In *European Semantic Web Conference*, pages 185–199, 2008.

[33] A. Kalyanpur, B. Parsia, M. Horridge, and E. Sirin. Finding all justifications of OWL DL entailments. In K. Aberer, K. Choi, N. F. Noy, D. Allemang, K. Lee, L. J. B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference,*

ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007, volume 4825 of *Lecture Notes in Computer Science*, pages 267–280. Springer, 2007.

[34] A. Kalyanpur, B. Parsia, E. Sirin, and J. Hendler. Debugging unsatisfiable classes in OWL ontologies. *Journal of Web Semantics*, 3(4):268–293, 2005.

[35] Y. Kazakov and P. Klinov. Goal-directed tracing of inferences in EL ontologies. In *International Semantic Web Conference*, pages 196–211, 2014.

[36] Y. Kazakov, M. Krötzsch, and F. Simancik. ELK reasoner: Architecture and evaluation. In *ORE*, 2012.

[37] Y. Kazakov and P. Skočovskỳ. Enumerating justifications using resolution. In *International Joint Conference on Automated Reasoning*, pages 609–626, 2018.

[38] R. D. King, J. Rowland, S. G. Oliver, M. Young, W. Aubrey, E. Byrne, M. Liakata, M. Markham, P. Pir, L. N. Soldatova, et al. The automation of science. *Science*, 324(5923):85–89, 2009.

[39] B. Konev, M. Ludwig, D. Walther, and F. Wolter. The logical difference for the lightweight description logic el. *J. Artif. Intell. Res.*, 44:633–708, 2012.

[40] B. Konev, M. Ludwig, D. Walther, and F. Wolter. The logical difference for the lightweight description logic EL. *Journal of Artificial Intelligence Research*, 44:633–708, 2012.

[41] B. Konev, C. Lutz, D. Walther, and F. Wolter. Logical difference and module extraction with CEX and MEX. In F. Baader, C. Lutz, and B. Motik, editors, *Proceedings of the 21st International Workshop on Description Logics (DL2008), Dresden, Germany, May 13-16, 2008*, volume 353 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[42] B. Konev, C. Lutz, D. Walther, and F. Wolter. Semantic modularity and module extraction in description logics. In *European Conference on Artificial Intelligence*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 55–59, 2008.

[43] B. Konev, C. Lutz, D. Walther, and F. Wolter. Formal properties of modularisation. In H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors, *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*, pages 25–66. Springer, 2009.

[44] B. Konev, C. Lutz, D. Walther, and F. Wolter. Model-theoretic inseparability and modularity of description logic ontologies. *Artif. Intell.*, 203:66–103, 2013.

[45] B. Konev, D. Walther, and F. Wolter. Forgetting and uniform interpolation in large-scale description logic terminologies. In *21st International Joint Conference on Artificial Intelligence*, pages 830–835, 2009.

[46] P. Koopmann. *Practical uniform interpolation for expressive description logics*. PhD thesis, University of Manchester, UK, 2015.

[47] P. Koopmann. Lethe: Forgetting and uniform interpolation for expressive description logics. *KI-Künstliche Intelligenz*, 34(3):381–387, 2020.

[48] P. Koopmann and J. Chen. Deductive module extraction for expressive description logics. In C. Bessiere, editor, *International Joint Conference on Artificial Intelligence*, pages 1636–1643, 2020.

[49] P. Koopmann and R. A. Schmidt. Forgetting concept and role symbols in ALCH-ontologies. In *Logic for Programming, Artificial Intelligence, and Reasoning - 19th International Conference*, pages 552–567, 2013.

[50] J. Lehmann and J. Völker. *Perspectives on Ontology Learning*, volume 18 of *Studies on the Semantic Web*. IOS Press, 2014.

[51] M. H. Liffiton and K. A. Sakallah. Algorithms for computing minimal unsatisfiable subsets of constraints. *J. Autom. Reason.*, 40(1):1–33, 2008.

[52] M. Ludwig and B. Konev. Practical uniform interpolation and forgetting for $\mathcal{ALC}$ tboxes with applications to logical difference. In C. Baral, G. D. Giacomo, and T. Eiter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014*. AAAI Press, 2014.

[53] M. Ludwig and D. Walther. The logical difference for ELHr-terminologies using hypergraphs. In *European Conference on Artificial Intelligence*, pages 555–560. 2014.

[54] M. Ludwig and D. Walther. The logical difference for $\mathscr{ELH}^{\mathsf{r}}$-terminologies using hypergraphs. In T. Schaub, G. Friedrich, and B. O'Sullivan, editors, *European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 555–560. IOS Press, 2014.

[55] C. Lutz and F. Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In T. Walsh, editor, *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 989–995. IJCAI/AAAI, 2011.

[56] A. Maedche and S. Staab. Ontology learning for the semantic web. *IEEE Intell. Syst.*, 16(2):72–79, 2001.

[57] N. Manthey, R. Peñaloza, and S. Rudolph. Efficient axiom pinpointing in EL using SAT technology. In *Description Logics*, 2016.

[58] N. Nikitina and B. Glimm. Hitting the sweetspot: Economic rewriting of knowledge bases. In P. Cudré-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. X. Parreira, J. Hendler, G. Schreiber, A. Bernstein, and E. Blomqvist, editors, *The Semantic Web - ISWC 2012 - 11th International Semantic Web Conference, Proceedings, Part I*, volume 7649 of *Lecture Notes in Computer Science*, pages 394–409. Springer, 2012.

[59] N. Nikitina and S. Rudolph. (non-) succinctness of uniform interpolants of general terminologies in the description logic EL. *Artificial Intelligence*, 215:120–140, 2014.

[60] N. F. Noy, N. H. Shah, P. L. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. L. Rubin, M. D. Storey, C. G. Chute, and M. A. Musen. BioPortal: ontologies and integrated data resources at the click of a mouse. *Nucleic Acids Res.*, 37(Web-Server-Issue):170–173, 2009.

[61] B. Parsia, N. Matentzoglu, R. S. Gonçalves, B. Glimm, and A. Steigmiller. The OWL reasoner evaluation (ORE) 2015 competition report. *J. Autom. Reason.*, 59(4):455–482, 2017.

[62] R. Peñaloza. Axiom pinpointing. *arXiv preprint arXiv:2003.08298*, 2020.

[63] R. Penaloza and B. Sertkaya. Understanding the complexity of axiom pinpointing in lightweight description logics. *Artificial Intelligence*, 250:80–104, 2017.

[64] A. Petrova, Y. Ma, G. Tsatsaronis, M. Kissa, F. Distel, F. Baader, and M. Schroeder. Formalizing biomedical concepts from textual definitions. *J. Biomedical Semantics*, 6:22, 2015.

[65] A. A. Romero, M. Kaminski, B. C. Grau, and I. Horrocks. Module extraction in expressive ontology languages via datalog reasoning. *J. Artif. Intell. Res.*, 55:499–564, 2016.

[66] D. L. Rubin, O. Dameron, Y. Bashir, D. Grossman, P. Dev, and M. A. Musen. Using ontologies linked with geometric models to reason about penetrating injuries. *Artificial Intelligence in Medicine*, 37(3):167–176, 2006.

[67] S. Rudolph. Exploring relational structures via FLE. In K. E. Wolff, H. D. Pfeiffer, and H. S. Delugach, editors, *Conceptual Structures at Work: 12th International Conference on Conceptual Structures, ICCS 2004, Huntsville,*

AL, USA, July 19-23, 2004. Proceedings, volume 3127 of *Lecture Notes in Computer Science*, pages 196–212. Springer, 2004.

[68] M. Sakr and R. A. Schmidt. Fine-grained forgetting for the description logic $\mathcal{ALC}$. In O. Arieli, M. Homola, J. C. Jung, and M. Mugnier, editors, *Proceedings of the 35th International Workshop on Description Logics (DL 2022)*, volume 3263 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[69] U. Sattler, T. Schneider, and M. Zakharyaschev. Which kind of module should I extract? *Description Logics*, 477:78, 2009.

[70] F. Simancik, Y. Kazakov, and I. Horrocks. Consequence-based reasoning beyond horn ontologies. In T. Walsh, editor, *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 1093–1098. IJCAI/AAAI, 2011.

[71] B. Suntisrivaraporn. Module extraction and incremental classification: A pragmatic approach for ontologies. In S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, editors, *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings*, volume 5021 of *Lecture Notes in Computer Science*, pages 230–244. Springer, 2008.

[72] J. Völker. *Learning expressive ontologies*. PhD thesis, Universität Karlsruhe, 2009.

[73] J. Völker, D. Fleischhacker, and H. Stuckenschmidt. Automatic acquisition of class disjointness. *J. Web Semant.*, 35:124–139, 2015.

[74] K. Wang, Z. Wang, R. W. Topor, J. Z. Pan, and G. Antoniou. Concept and role forgetting in *ALC*{\mathcal {ALC}} ontologies. In A. Bernstein, D. R. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, editors, *The Semantic Web - ISWC 2009, 8th International Semantic Web Conference, ISWC 2009, Chantilly, VA, USA, October 25-29, 2009. Proceedings*, volume 5823 of *Lecture Notes in Computer Science*, pages 666–681. Springer, 2009.

[75] H. Yang, Y. Ma, and N. Bidoit. Hypergraph-based inference rules for computing $\mathcal{EL}^+$-ontology justifications. In *International Joint Conference on Automated Reasoning*, pages 310–328, 2022.

[76] Y. Zhao, G. Alghamdi, R. A. Schmidt, H. Feng, G. Stoilos, D. Juric, and M. Khodadadi. Tracking logical difference in large-scale ontologies: A forgetting-based approach. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 3116–3124. AAAI Press, 2019.

[77] Y. Zhao and R. A. Schmidt. Role forgetting for $\mathcal{ALCOQH}(\nabla)$-ontologies using an Ackermann-based approach. In C. Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, pages 1354–1361. ijcai.org, 2017.

[78] Y. Zhao and R. A. Schmidt. Fame: an automated tool for semantic forgetting in expressive description logics. In *International Joint Conference on Automated Reasoning*, pages 19–27, 2018.

[79] Y. Zhao and R. A. Schmidt. FAME(Q): an automated tool for forgetting in description logics with qualified number restrictions. In P. Fontaine, editor, *Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings*, volume 11716 of *Lecture Notes in Computer Science*, pages 568–579. Springer, 2019.