FOCUS SET SUBONTOLOGIES AND SEMANTIC DIFFERENCES FOR LARGE $\mathcal{ELH}$ ONTOLOGIES

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN THE FACULTY OF SCIENCE AND ENGINEERING

2022

Ghadah Abdulrahman S Alghamdi

Department of Computer Science

Contents

Word Count: 72250

List of Tables

List of Figures

Abstract

FOCUS SET SUBONTOLOGIES AND SEMANTIC DIFFERENCES FOR LARGE $\mathcal{ELH}$ ONTOLOGIES

Ghadah Abdulrahman S Alghamdi

A thesis submitted to The University of Manchester for the degree of Doctor of Philosophy, 2022

Ontologies have been widely used in a variety of fields to serve as sources for formally structured knowledge. The Systemized Nomenclature of Medicine – Clinical Terms (SNOMED CT) ontology, among others, is widely used in the (bio-)medical domain, as it provides a comprehensive multilingual vocabulary for capturing all aspects of electronic health records and clinical knowledge, resulting in a very large and complex ontology. Ontology extraction methods enable efficient use of such large ontologies by splitting them into interconnected smaller parts.

This thesis presents a novel method for extracting focus set subontologies from $\mathcal{ELH}$ ontologies for a set of symbols selected by the user. The resulting subontologies satisfy the requirements sought by SNOMED CT users, including providing complete semantics for the description of input symbols while being concise and conforming to SNOMED CT modelling principles. The findings show that, in comparison to locality-based modularisation and uniform interpolation, the resulting subontologies satisfy the requirements of SNOMED CT in terms of size, encapsulating the entire semantics of the definitions solely for the set of input symbols, and retaining the original ontology's structure.

This thesis also investigates the computation of semantic differences between extracted subontologies using a combination of our notion of subontologies and the uniform interpolation-based semantic difference method in order to constrain the development of such differences to a particular subdomain of the ontology identified by the user. The findings demonstrate that our method is capable of revealing differences in the meaning of focus concept definitions associated with a particular subdomain of the ontology, where some of these differences would not have been generated without this focused approach to tracking semantic differences between ontologies.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I would like to offer my deep gratitude to my supervisor, Renate A. Schmidt, for her advice and assistance throughout my PhD journey. You have been an invaluable resource and have never hesitated to provide me with information and suggestions that I must explore in order to complete the project.

I would like to express my gratitude to my sponsor, the Saudi Ministry of Education, as represented by the Royal Embassy of Saudi Arabia Cultural Bureau in the United Kingdom, for financially supporting me during the project.

I would also like to thank the people I have had the pleasure of working with and learning from during my PhD programme, particularly Yongsheng Gao of SNOMED International, who was helpful during our meetings with his creative ideas. Also, Yizheng Zhao, Jieying Chen, and Warren Del-Pinto, with whom I had the pleasure of working and from whom I have gained a lot of knowledge.

I would also like to express my gratitude to Haifa Alrdahi for her kindness and invitation to stay with her for around two months during a time when I was in desperate need.

Thank you also to my colleagues Ruba, Kiana, Sen, David, and Mostafa for their insightful comments on my project during our seminars.

Last but not least, I would like to express my deepest gratitude to my mother for believing in me and my father for providing the support I needed during my difficult times.

Chapter 1

Introduction

In computer science, ontologies are a type of knowledge representation that is used to identify and establish a topic's vocabulary in order to facilitate information exchange. Ontologies convey knowledge through the use of axioms, which are logical statements that precisely describe the meaning of concepts and roles within a domain. Ontologies are typically constructed using description logics (DL); a family of knowledge representation languages that are composed of several decidable fragments of first-order logic, which comes in a variety of expressivity ranges [BCM+07]. Automated reasoning systems can be used to infer implicit information from ontologies.

The Web Ontology Language (OWL) [W3C12] is a family of machine-processable languages developed by the World Wide Web Consortium (W3C), whose semantics are based on the DL languages. The W3C has developed tractable sublanguages (so-called profiles [MGH+12]) of OWL, such as OWL EL, to account for the scalability of reasoning and other services that employ ontologies. OWL EL profile is based on the $\mathcal{ELH}$ family of description logics [Bra04, BBL05].

Ontologies have been widely applied in a range of fields, including Medicine [Spa00a, GFH+03], Biology [WNS+11], Engineering [DdME+08], Energy [BEF+21], and Law [HBBB09]. These ontologies are used for a variety of purposes, including providing a source of formally structured knowledge, and have evolved from being used for a particular purpose to providing a range of services such as decision support and resource management.

In particular, in the (bio-)medical domain, the Systemized Nomenclature of Medicine – Clinical Terms (SNOMED CT) ontology [Spa00b], the National Cancer Institute (NCI) Thesaurus (NCIt) ontology [GFH+03], and the Gene Ontology (GO) [The18] are prominent. These ontologies are materials that have been collaboratively generated,

generally by groups of engineers, using principled design and deployment processes. The increasing complexity and size of these ontologies pose significant challenges to ontology modellers, maintainers, and potential users. For example, the latest version of SNOMED CT (February 2022) consists of 356 813 axioms, 355 235 concepts and 125 roles.

SNOMED CT is widely recognised as one of the most extensive ontology initiatives in the biomedical industry, providing a complete, multilingual vocabulary for capturing all parts of electronic health records and clinical knowledge. The ontology is based on a multihierarchical taxonomy of over 350 000 concpets, describing many subdomains of biomedicine, including acupuncture, infectious diseases, chronic disorders, and pharmaceutical products. The primary goal of SNOMED CT is to standardise clinical terminology globally by linking information from diverse clinical information systems and research data [Inta]. As a result, it is used in 42 countries and by over 5000 affiliate individuals and organisations. The SNOMED CT international edition is managed and maintained by SNOMED International[1] organisation in the UK.

Ontology extraction methods enable efficient use of such massive ontologies by dividing them into interconnected smaller parts. Modularisation is one of these methods; it generates subsets containing the axioms of the original ontology for a given set of symbols. These subsets preserve the original ontology's semantic relationships and syntactic shapes of the axioms [SS20, CMG+22]. Therefore, modularity is relevant in the biomedical domain, and has been extensively studied to determine its potential for supporting novel applications to enhance scientific research [PJC09]. Furthermore, the application scenarios for modularisation in relation to semantic annotation are diverse, ranging from clinical notes for acute care services to medical photographs [WSB11, LBIS12].

Because the majority of information contained in ontologies is implicit and the entities contained inside an ontology may interact in non-trivial ways, comprehending and managing a complex ontology requires the usage of appropriate tool support. Uniform interpolation, a logic-based method for constructing compact representations, can be utilised for this purpose [KWW09, LK14, NR14]. Uniform interpolation aids in the extraction of implicit information from an ontology for a given set of symbols. The resulting information is a constrained view of that ontology that use just the symbols from the specified signature while preserving all the ontology's logical entailments. Uniform interpolation may be used to assist with a variety of ontology-related tasks.

[1] snomed.org

It can be used to extract portions of an ontology for analysis or to remove confidential information, among other tasks.

A range of applications can be satisfied through ontology extraction using modularisation and uniform interpolation [SPS09, GHKS08, KWZ10, KLWW13, CLMW19, MCNK15, LW11, WWTP10]. Such applications include *ontology analysis;* for examining and understanding the semantics of definitions and hierarchy relevant to a specific domain. Another application is *ontology authoring;* allowing extracting standalone ontologies that can be utilised as a starting point or enhancement to existing extendable ontologies, enabling scalability of evolution and maintenance. Additionally, decomposing an ontology into smaller portions simplifies *ontology debugging;* this enables easier ontology repair by ensuring that such portions are consistent and coherent. Moreover, segmenting the ontology into smaller parts facilitates *ontology usability;* allowing content comprehension and context awareness (personalization), as well as *reuse of extracted content*; providing extracted content as a standalone ontology in a format compatible with existing technologies. Furthermore, ontology extraction enhances *computability tasks* by increasing the efficiency of reasoning tasks, such as querying data.

Another significant application is *tracking semantic difference between ontology versions* [KWW08, KLWW12]. Identifying changes between two versions of a document is a common problem in information technology, and it is especially important for text processing and software development. Significant changes are even more important in the context of knowledge representation, because differences in the knowledge recorded by ontologies are typically more significant than syntactic changes. A purely syntactic diff operation is hardly useful to compare ontologies. Therefore, *tracking differences in the meaning* of ontologies that have occurred as a result of ontology revisions or updates regardless of its syntactic structure is typically more important for ontology engineers to keep track of.

Semantic difference computation based on uniform interpolation aid in discovering implicit differences within an ontology's deductive closure [LK14]. Additionally, computing semantic difference via uniform interpolation assists in eliminating symbols that appear in one of the ontology versions but not the other in order to determine whether ontology entailments (consequences) after forgetting such symbols is equivalent in both versions (before and after updates) [LK14]. This semantic difference detection is beneficial for a variety of applications, including those that need the integration or mapping of several ontologies while ensuring that the integrated information

does not overlap [SGW+18].

1.1 Ontology Extraction for SNOMED CT

At present, the large size of SNOMED CT is dealt with by the use of reference sets. Reference sets are domain-specific subsets of concept names that are used to limit querying, searching, and data entry to a subset of the application area, e.g., the ERA-EDTA reference set [NCS+18]. They are carefully created to represent precise concept definitions from the ontology indicating their intended function. Typically, searching for information associated with such reference sets implies utilising the original ontology to query data pertaining to the reference sets in concern. Despite ongoing efforts to enhance storage and querying systems, their performance is impacted by the ontology's size and complexity, necessitating the development of effective methodologies for ontology extraction.

For SNOMED CT users, ontology extraction is relevant for maintenance, development and quality assurance. It is inevitable that editors and content maintainers will focus their editorial work on a subset of the content. As an example from the literature, the work in [OCP16] focuses on remodelling of the bacterial infectious disease subhierarchy in SNOMED CT. Additionally, methods for segmenting the SNOMED CT ontology into several subhierarchies based on a structural analysis of the underlying SNOMED CT hierarchy were also introduced in [OCP17, OGP+15, LS16]. Such methods were developed for the purpose of improving the analysis and debugging of SNOMED CT content.

Currently, SNOMED CT users do not use modularisation or uniform interpolation. Modules are not used by SNOMED CT users since they generally do not meet their specific requirements. On the one hand, the notion of syntactic locality-based modules was established to efficiently generate modules for a given set of terms [GHKS07]. However, when applied to very large ontologies such as SNOMED CT, the resulting modules are quite large, limiting their use and evolution [dSSS09]. On the other hand, other module notions such as semantic modules [KLWW13] and minimal subsumption modules [CLW18] may be too small in size, in the sense that the generated module does not contain the complete semantics of definitions for the given set of concepts, resulting in issues with other criteria such as connectedness [dSSS09]. Moreover, they can be computationally expensive due to the large size of SNOMED CT.

Uniform interpolation, which was evaluated in this thesis for its practicality when

applied to SNOMED CT, has a number of shortcomings that render it unsuitable. One of these limitations is the small size of the resulting uniform interpolants, which may not contain all of the specified symbols' definitions. Additionally, the axioms' structure in the resulting uniform interpolants is notably different from the original ontology, which violates SNOMED CT's modelling guidelines. Another significant issue is the high complexity of generating uniform interpolants, as uniform interpolation is a challenging non-standard reasoning task. A case in point is the computation of uniform interpolants for a very small number of symbols from the very large SNOMED CT. Most real-world signatures such as those found in SNOMED CT's standard reference sets have a small number of symbols in relation to the original ontology's large size. This indicates that when used to such a use case, uniform interpolation must eliminate a large number of symbols from the original ontology (99% of the ontology's symbols), which is computationally infeasible on SNOMED CT.

SNOMED CT users at SNOMED International are looking for extracts that satisfy a number of predefined criteria. The first criterion is the provision of complete definitions for the set of symbols from which a subontology is to be extracted. The second criterion is that the generated subontologies retain the structure of the axioms as specified by the modelling guidelines. A third criterion is that the information contained inside the created subontologies remain brief, i.e., containing only definitions of the concepts of interest and other additional hierarchical relationships that exist between the symbols of such definitions.

One of the two main aims of this thesis is to introduce a new functionality for creating SNOMED CT subontologies that meet the aforementioned criteria. We want subontologies to be used in place of reference sets, which are simply a flat list of concept names. Subontologies, on the other hand, should incorporate all semantic relationships between the concepts in the reference set from the original ontology. Our method should be applicable to ontologies that are expressed in the $\mathcal{ELH}$ language.

1.2 Semantic Difference for SNOMED CT

A real-world ontology, such as SNOMED CT, is regularly updated to reflect domain changes through the inclusion of new concepts, the retirement of obsolete concepts, and the modification of existing concepts. Typically, updating the terminology results in a significant number of changes in a SNOMED CT release. For instance, roughly 192,000 changes were made to the content of SNOMED CT in the January

2015, July 2015, and January 2016 releases [OPEC16]. To track such modifications, the ontology's changes are published as delta release files that allow for the review of changes to the ontology's axioms. The delta files describe the changes that have occurred to the ontology's structure, i.e. if new axioms have been added or deleted. Such changes within the delta files are scattered over the ontology's various subhierarchies. Without the ability to extract certain subdomains from the ontology, it is envisaged that users will be overwhelmed by the volume of changes to analyse. As a solution, the work [OPEC16], provided a method for visualising and segmenting the data contained in delta files. This is to facilitate the user's reading and tracking of changes to SNOMED CT content.

There is presently no technique for tracking differences in the semantics between SNOMED CT versions that are exclusive to a particular subdomain of the ontology. As previously mentioned, uniform interpolation can be used as a means to compute logical differences between ontology versions [LK14]. Moreover, given that our proposed method of generating subontologies adheres to the requirements of SNOMED International, we found that it is a suitable approach to track semantic differences. In this case, the computed semantic differences are focused on particular subdomains of the ontology defined by the given set of terms.

The second main aim of this thesis is to develop a new method for tracking semantic differences between extracted subontologies for a given set of symbols from the ontology versions the user wishes to audit. This method is based on our proposed notion of subontologies suited for SNOMED CT users and utilises the notion of uniform interpolation to compute semantic differences. With the assumption that the input symbols are specific to a certain subdomain of the original ontology, the user will be able to generate the axioms describing the change in meaning within the chosen domain of the ontology.

1.3 Aims and Contributions

In summary, the main focus the thesis is to develop novel functionalities for (i) extracting subontologies and (ii) tracking semantic differences for SNOMED CT and other possible $\mathcal{ELH}$ ontologies.

The introduced methods were evaluated using implemented prototypes, which demonstrate that it is indeed possible to compute subontologies for target ontologies as large as SNOMED CT for real-world signatures, as well as to track semantic differences

within the specified subdomain of the ontology.

The following summarises the main contributions of the thesis:

1. We present a novel method for extracting focus set subontologies for a user-selected set of symbols. This method is capable of resolving the concerns raised in the preceding methods such as modularisation and uniform interpolation in order to generate extracts suitable for users of SNOMED CT.

2. We present a novel method for tracking focus set semantic differences that is based on the combination of our notion of subontologies and the uniform interpolation based semantic difference method in order to limit the development of such differences to a certain subdomain of the ontology that the user identifies.

3. An investigation and evaluation of the combination of modularisation and uniform interpolation to generate uniform interpolants suitable for the use case of real-world signatures. Though not achieving the desired usability requirements, this work was a useful precursor that led to the development of the subontology extraction and the semantic difference computation methods. (See points 1 and 2).

4. As part of the modularisation-uniform interpolation workflow, we developed signature extension and partitioning algorithms that can be used with real-world signatures, such as those included in SNOMED CT's standard reference sets.

5. Evaluation of our method to generate focus set subontologies with related methods namely, locality-based modularisation and uniform interpolation. The findings showed that the resulting subontologies in comparison to those generated by the other methods satisfy the requirements that are relevant to SNOMED CT domain experts in terms of size, capturing the whole semantics of the definitions solely for the set of input symbols, and retaining the original ontology's structure.

6. Evaluation of our method to generate focus set semantic differences focused on specific domains for real-world-signatures provided by a SNOMED CT domain expert. The findings showed that our method was capable of generating differences in the meaning of the definitions of focus concepts associated with a particular subdomain of the ontology, where some of these differences would not be generated without this focused approach to tracking semantic differences between ontologies.

7. In our evaluation of the focus set subontology generator method and the uniform interpolation-modularisation workflow, we have incorporated biomedical ontologies other than SNOMED CT, such as the Gene ontology and NCIt to highlight critical aspects of our methods while utilising these ontologies.

1.4 Published Results

Throughout the course of my PhD research, I actively contributed to several projects, which resulted in the following papers published at international workshops and conferences.

1-1 [SGW+18] Giorgos Stoilos, David Geleta, Szymon Wartak, Sheldon Hall, Mohammad Khodadadi, Yizheng Zhao, Ghadah Alghamdi, and Renate A. Schmidt. Methods and Metrics for Knowledge Base Engineering and Integration. In *Proceedings of the 9th Workshop on Ontology Design and Patterns (WOP 2018) co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 9th, 2018*, volume 2195 of *CEUR Workshop Proceedings*, pages 72–86. CEUR-WS.org, 2018

1-2 [ZAS+19b] Yizheng Zhao, Ghadah Alghamdi, Renate A. Schmidt, Hao Feng, Giorgos Stoilos, Damir Juric, and Mohammad Khodadadi. Tracking Logical Difference in Large-Scale Ontologies: A Forgetting-Based Approach. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3116–3124. AAAI Press, 2019

1-3 [ZAS+19a] Yizheng Zhao, Ghadah Alghamdi, Renate A. Schmidt, Hao Feng, Giorgos Stoilos, Damir Juric, and Mohammad Khodadadi. Tracking Logical Difference in Industrial-Scale Ontologies. In *Proceedings of the 32nd International Workshop on Description Logics, Oslo, Norway, June 18-21, 2019*, volume 2373 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019

The above publications are the result of a collaboration between Babylon Health® and the University of Manchester. The project used a prototype tool built by Yizheng Zhao to compare semantic differences between various versions of

SNOMED CT International edition and country extensions. I contributed to the project by analysing the resulting differences. The analysis involved partitioning the differences based on whether they are stated or inferred in the respective ontologies. The analysis provided Babylon collaborators with a better understanding of the type of differences generated in the Australian extension.

The work [ZAS+19a] is an extended abstract of the work [ZAS+19b].

2-1 [CAS+19a] Jieying Chen, Ghadah Alghamdi, Renate A. Schmidt, Dirk Walther, and Yongsheng Gao. Modularity Meets Forgetting: A Case Study with the SNOMED CT Ontology. In *Proceedings of the 32nd International Workshop on Description Logics*, volume 2373. CEUR-WS.org, 2019

2-2 [CAS+19b] Jieying Chen, Ghadah Alghamdi, Renate A. Schmidt, Dirk Walther, and Yongsheng Gao. Ontology Extraction for Large Ontologies via Modularity and Forgetting. In *Proceedings of the 10th International Conference on Knowledge Capture, K-CAP'19*, pages 45–52. ACM, 2019

The publications are the result of a proof of concept project conducted in collaboration with SNOMED International. The aim was to use the uniform interpolation method and modularity notions to generate abstracts for different SNOMED CT reference sets while soliciting feedback from the collaborator. My contribution included the development of signature adjustment algorithms, the execution of experiments utilising the workflow, namely with the systems LETHE and UI-FAME, the analysis of the results, and the authoring of parts of the manuscripts.

The work [CAS+19a] includes our initial results of the work in [CAS+19b].

Chapter 4 of this thesis contains a more detailed presentation of the evaluations in [CAS+19b].

3-1 [LLA+21a] Zhao Liu, Chang Lu, Ghadah Alghamdi, Renate A. Schmidt, and Yizheng Zhao. Tracking Semantic Evolutionary Changes in Large-Scale Ontological Knowledge Bases. In *CIKM'21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 1130–1139. ACM, 2021

3-2 [LLA+21b] Zhao Liu, Chang Lu, Ghadah Alghamdi, Renate A. Schmidt, and Yizheng Zhao. Tracking Semantic Evolutionary Changes in Large-Scale Ontological Knowledge Bases. In *Proceedings of the 34th International Workshop on Description Logics (DL 2021) part of Bratislava Knowledge September*

(BAKS 2021), Bratislava, Slovakia, September 19th to 22nd, 2021, volume 2954 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021

The papers were the outcome of a collaboration with Nanjing University on the development of a tool for tracking semantic differences across versions of $\mathcal{ELH}$ ontologies as a continuation of the research started with Babylon Health®. I conducted the experiments of generating focused semantic differences related to the General Dentistry reference set, wrote the results and provided overall feedback on the paper. My part in [LLA+21a] is included and expanded in Chapter 6.

The work [LLA+21b] is a short version of the work [LLA+21a].

4-1 [ASDG21a] Ghadah Alghamdi, Renate A. Schmidt, Warren Del-Pinto, and Yongsheng Gao. Upwardly Abstracted Definition-Based Subontologies. In *Proceedings of the 34th International Workshop on Description Logics (DL 2021) part of Bratislava Knowledge September (BAKS 2021), Bratislava, Slovakia, September 19th to 22nd, 2021*, volume 2954 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021

4-2 [ASDG21b] Ghadah Alghamdi, Renate A. Schmidt, Warren Del-Pinto, and Yongsheng Gao. Upwardly Abstracted Definition-Based Subontologies. In *K-CAP'21: Knowledge Capture Conference*, pages 209–216. ACM, 2021

The above publications are the result of a continuation of our collaboration with SNOMED International on extracting focused extracts for their reference sets. I devised a novel method for generating focus set subontologies in this research. The resulting subontologies are more beneficial and relevant to SNOMED International. I wrote the manuscripts, developed the system and conducted the experiments.

The work [ASDG21a] includes our initial results of the work in [ASDG21b].

Chapter 5 is an extended version of the paper [ASDG21b].

1.5 Thesis Overview

The following gives a summary and synopsis of each chapter of this thesis.

- Chapter 2 provides background material about ontologies and description logics. Our review highlights description logics used in biomedical ontologies. Additionally, we define the various standard reasoning services that can be employed

with description logic-based ontologie and cover the ontology tools that were used in this thesis. Furthermore, we provide a comprehensive overview of the medical ontology SNOMED CT as well as the identification of significant real-world signatures (standard reference sets) that are actively employed in application contexts. Finally, we discuss additional ontologies that were used in our method's evaluations, including the NCIt and Gene ontologies.

- Chapter 3 covers the state-of-the-art approaches for ontology extraction and difference computation. We review, among other ontology extraction methods, ontology modularity and uniform interpolation methods and point out their shortcomings related to our research problem. In terms of difference computation methods, our review focuses on a method for computing semantic differences using uniform interpolation, among other methods.

- Chapter 4 is the first main chapter of the thesis. In it, we present our workflow to assess the practicality of computing uniform interpolants which is based on pre-processing stages including signature adjustment and modularisation steps. We evaluate the workflow with real-world ontologies, namely SNOMED CT and NCIt.

- The most important chapter of the thesis is Chapter 5. In it, we introduce a novel method for extracting *focus set subontologies* from large $\mathcal{ELH}$ ontologies. We evaluated the quality of our subontologies and compared them to relevant methods in the area, namely locality-based modularisation and uniform interpolation in terms of different proposed quality measures. The findings showed that the subontologies are more concise than locality-based modules and uniform interpolants while providing the complete semantics of the given symbol definitions.

- In Chapter 6 we present a method for computing a new notion of semantic differences named *focus set semantic differences*. We evaluated the method using several SNOMED CT reference sets in order to produce focused semantic differences relating to such reference sets. We demonstrate the method's outcomes by illustrating how modellers might use it to compute semantic differences.

- Finally in Chapter 7, we draw conclusions regarding the results of our methods introduced in this thesis. We discuss potential applications of focus set subontologies and semantic differences as well as some ideas for future work.

Chapter 2

Ontologies and Description Logics

Ontologies are a type of knowledge representation used in computer and information sciences to identify and establish the vocabulary of a topic of interest in order to enable information exchange. In this chapter, we establish the foundations for description logics ontologies and the central background notions that will be used throughout the thesis.

In Section 2.1, we define the fundamental concepts of popular description logics languages. We discuss the Web Ontology Language (OWL) in Section 2.2, which is a standard machine-processing language based on description logics. In Section 2.3 we go through the tools we used during the project of the thesis. In Section 2.4, we discuss the medical ontology SNOMED CT, which is the target ontology for our methods. Lastly, in Section 2.5 we discuss some other ontologies in the field of biomedical science that were used in our evaluations in Chapters 4, 5, and 6.

2.1 Description Logics

Description logics (DLs) are decidable fragments of first order logic (FOL), well suited for expressing domain knowledge [BN07]. The knowledge of a domain is expressed by first establishing domain concepts and then utilising these concepts to specify the properties of objects and individuals found in the domain. DLs enable a variety of reasoning tasks, including satisfiability of knowledge bases, axiom entailment, and instance retrieval [BN07].

We now define the description logic $\mathcal{SHOIQ}$, which is the union of all previously considered classical description logics.

Concept descriptions are defined inductively in DLs using a set of three constructors that are countably infinite and mutually disjoint, beginning with a set of concept names $\mathsf{N_C}$, a set of role names $\mathsf{N_R}$, and (potentially) a set of instance names $\mathsf{N_I}$. The union of such sets form the signature of an ontology $\mathcal{O}$. The signature $\text{sig}(\xi)$ is a set of concept and role names that occur in ξ, where ξ is any syntactic object or ontology. The signature $\text{sig}_C(\xi)$ is a set of concept names that occur in ξ.

A concept is an expression of one of the following forms, where $A \in \mathsf{N_C}$, C, C_1 and C_2 are any concepts, R is any role, n and m are natural numbers with $n \geq 1$, $m \geq 0$, respectively, and $a_1, \ldots, a_n \in \mathsf{N_I}$.

$$\top \mid \bot \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2,$$

$$\exists R.C \mid \forall R.C \mid \ \geq nR.C \mid \ \leq nR.C,$$

$$\{a_1, \ldots, a_n\}$$

The concepts in the first row are concept descriptions, which are referred to from left to right as top, bottom, negation of the concept C, conjunction of the concepts C_1 and C_2, and disjuntion of the concepts C_1 and C_2. The concepts in the second row are role restrictions which are referred to from left to right as existential restrictions, universal restrictions, $\geq$-number restrictions and $\leq$-number restrictions.

The axioms in DL ontologies are categorised into three different types:

- TBox, which contains terminological information about concepts and their relationships, describing how a knowledge base is conceptualised,
- RBox, representing role axioms, and
- ABox, which encapsulates knowledge of facts or assertions about individuals that fall under a particular concept.

A TBox $\mathcal{T}$ is a finite set of axioms consisting of the following forms:

- $C_1 \sqsubseteq C_2$ (concept inclusion axiom)
- $C_1 \equiv C_2$ (concept equivalence axiom)

where C_1 and C_2 are any concepts. The axiom $C_1 \equiv C_2$ is a short hand for the two concept inclusion axioms $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$.

An RBox $\mathcal{R}$ is a finite set of role axioms, which consists of the following forms:

- $R \sqsubseteq S$, (role inclusion axiom)
- $R \equiv S$, (role equivalence axiom)
- $\mathsf{trans}(R)$ (role transitivity axiom)

where R and S are any roles. $R \equiv S$ is a short hand of the two role inclusion axioms $R \sqsubseteq S$ and $S \sqsubseteq R$.

An ABox $\mathcal{A}$ is a finite set of axioms of the following form:

- $C(a)$ (concept assertion axiom)
- $R(a, b)$ (role assertion axiom)

Definition 1 (Ontology $\mathcal{O}$)**.** *An ontology $\mathcal{O}$ is the union of a TBox $\mathcal{T}$, an ABox $\mathcal{A}$, and an RBox $\mathcal{R}$.*

In the remainder of the thesis, the terms *TBox* and *ontology* are used interchangeably.

The semantics of description logics is defined in terms of an *interpretation* $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, where $\Delta^{\mathcal{I}}$ is the *domain of the interpretation* (a non-empty set), and $\cdot^{\mathcal{I}}$ denotes the *interpretation function*, which assigns to every concept name $A \in \mathsf{N_C}$ a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and to every role name $r \in \mathsf{N_R}$ a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The interpretation function $\cdot^{\mathcal{I}}$ is inductively extended to concepts as shown in Table 2.1.

Table 2.1: Concept constructors for the $\mathcal{ALC}$ DL

Constructor	Syntax	Semantics
Bottom	$\bot$	$\emptyset$
Top	$\top$	$\Delta^{\mathcal{I}}$
Negation (Complement)	$(\neg C)^{\mathcal{I}}$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
Conjunction (intersection)	$(C \sqcap D)^{\mathcal{I}}$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Disjunction (union)	$(C \sqcup D)^{\mathcal{I}}$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Existential restriction	$(\exists r.C)^{\mathcal{I}}$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y.(x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
Universal restriction	$(\forall r.C)^{\mathcal{I}}$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y.(x, y) \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$

Let $\mathcal{I}$ be an interpretation, an axiom α is *true* (i.e., $\mathcal{I} \models \alpha$) when the following conditions hold:

- α as a general concept inclusion $C \sqsubseteq D$ is *true* in $\mathcal{I}$ iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$.
- α as a general concept equivalence $C \equiv D$ is *true* in $\mathcal{I}$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$.

- α as a role inclusion $r \sqsubseteq s$ is *true* in $\mathcal{I}$ iff $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$.
- α as a role equivalence $r \equiv s$ is *true* in $\mathcal{I}$ iff $r^{\mathcal{I}} \equiv s^{\mathcal{I}}$.
- α as a role transitivity $\mathsf{trans}(R)$ is *true* in $\mathcal{I}$ iff $R^{\mathcal{I}}$ is transitive.
- α as a concept assertion $C(a)$ is *true* in $\mathcal{I}$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$.
- α as a role assertion $R(a, b)$ is *true* in $\mathcal{I}$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$.

$\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$, written as $\mathcal{I} \models \mathcal{O}$, iff every axiom in $\mathcal{O}$ is *true* in $\mathcal{I}$. An axiom α is a *logical entailment* of an ontology $\mathcal{O}$, written as $\mathcal{O} \models \alpha$, iff α is true in every model $\mathcal{I}$ of $\mathcal{O}$.

Given a signature Σ, an interpretation $\mathcal{I}$ is restricted to symbols in Σ denoted as $\mathcal{I}|_{\Sigma}$ when $\Delta^{\mathcal{I}|_{\Sigma}} = \Delta^{\mathcal{I}}$ and $X^{\mathcal{I}|_{\Sigma}} = X^{\mathcal{I}}$ for all $X \in \Sigma$.

2.1.1 Different DLs and DLs for Biomedical Ontologies

We obtain different description logics by limiting the operators that can be used to construct concepts and roles. Table 2.2 summarises the operators that are allowed in the description logics $\mathcal{EL}$, $\mathcal{ALC}$ and $\mathcal{S}$. A description logic is typically denoted by prefixes that specify the additional permitted operators for one of these description logics, as in $\mathcal{ELH}$ for $\mathcal{EL}$ with role hierarchies. Table 2.3 shows the allowed constructors that can be used as extensions for the description logics names shown in Table 2.2.

Table 2.2: Description logics families

Description Logics Name	Constructors
$\mathcal{EL}$	A, $C \sqcap D$, $\exists R.C$ and $\top$
$\mathcal{ALC}$	$\mathcal{EL} + \forall R.C + \neg C$
$\mathcal{S}$	$\mathcal{ALC} + \mathsf{trans}(R)$

Table 2.3: Some extensions that are allowed for a description logic

Extension Name	Constructors	Description
$\mathcal{H}$	$R \sqsubseteq S$, $R \equiv S$	Role hierarchies
$\mathcal{O}$	$\{a_1, \ldots, a_n\}$	Nominals
$\mathcal{I}$	r^-	Inverse roles
$\mathcal{F}$	$\leq 1R.\top$	Functional role restrictions
$\mathcal{Q}$	$\geq nR.C, \leq nR.C$	Qualified number restrictions

An $\mathcal{ELH}$ ontology $\mathcal{O}$ is called an $\mathcal{ELH}$ terminology if its TBox consists of the following form of axioms:

- $A \sqsubseteq C$ (primitive concept axiom, where A is a primitive concept name)
- $A \equiv C$ (defined concept axiom, where A is a defined concept name)

where $A \in \mathsf{N_C}$, $r, s \in \mathsf{N_R}$ and C is an $\mathcal{EL}$-concept, with A appearing not more than once on the left-hand side of an axiom. In primitive concept axioms $A \sqsubseteq C$, C specifies the *necessary* conditions for the concept A, while in defined concept axioms $A \equiv C$, C specifies the *necessary and sufficient* conditions of the concept A.

An $\mathcal{ELH}$ ontology is normalised iff it only contains axioms of the forms $A \sqsubseteq C$, $A \sqsubseteq \exists r.C$, $r \sqsubseteq s$ and $\exists r.C \sqsubseteq A$, where $A \in \mathsf{N_C}, r, s \in \mathsf{N_R}$ and C is an $\mathcal{EL}$-concept.

An $\mathcal{ELH}$-TBox $\mathcal{O}$ is in clausal form if every axiom in $\mathcal{O}$ is in clausal form. In clausal form, an axiom is referred to as a *clause*. The clausal form of an axiom is a disjunction of literals of the form A, $\neg A$, $\exists R.C$ or $\forall R.C$ where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$ and C is in *negation normal form*. A concept is in negation normal form when the negation operator is applied only to concept symbols.

Figure 2.1 shows transformation rules of axioms into clausal forms [Zha18]. Rules a. to g. can be used to transform concepts into negation normal forms. The rule h can be used to distribute disjunctions over conjunctions. Such rules are standard clausal form transformation rules where the left-hand side of the $\Longrightarrow$ relation is logically equivalent to its right-hand side [Zha18].

Let $\mathcal{O}$ be an $\mathcal{ELH}$-TBox and $A, B_i \in \mathsf{N_C}$ $(1 \leq i \leq n)$ be atomic concepts in $\mathcal{O}$. We say that A *directly depends on* B $(A \prec B)$ iff the clausal form of $\mathcal{O}$ includes a clause in which A occurs positively and B negatively (or vice versa). We say that A *depends on* B_n iff there is a chain of $A, B_1, \ldots, B_n$ such that $A \prec B_1 \prec \ldots \prec B_n$. $\mathcal{O}$ is *acyclic* if there is no concept name in $\mathcal{O}$ that depends on itself; otherwise it is *cyclic*.

Biomedical ontologies such as SNOMED CT and NCIt share similar modelling features. One of the main features is the elementary concepts that form the skeleton of such ontologies. These elementary concepts are referred to as *natural kinds* according to [Rec95]. Natural kinds are defined in terms of primitive concept subsumption axioms. They lack of sufficient conditions to be fully defined. Such concepts are rather recognised than inferred [Rec95]. For example, the concept *Normal anatomy (body structure)* is a natural kind in SNOMED CT, which is expressed by a primitive concept axiom as being subsumed by *Body structure*, which is another natural kind.

Negation normal form transformation rules

- De Morgan's Laws:
  a. $\neg(C \sqcup D) \implies \neg C \sqcap \neg D$
  b. $\neg(C \sqcap D) \implies \neg C \sqcup \neg D$
- Duality between $\exists$- and $\forall$- restrictions:
  c. $\neg\exists r.C \implies \forall r.\neg C$
  d. $\neg\forall r.C \implies \exists r.\neg C$
- Duality between $\top$ and $\bot$:
  e. $\neg\top \implies \bot$
  f. $\neg\bot \implies \top$
- Double negation elimination:
  g. $\neg\neg C \implies C$

Distributivity law

h. $C \sqcup (D_1 \sqcap D_2) \implies C \sqcup D_1, C \sqcup D_2$

Figure 2.1: Rules used to transform axioms into clausal forms [Zha18]

Definitions, with both necessary and sufficient conditions ($A \equiv C$), are critical in biomedical ontologies because they assist in determining which concepts are classified under them in the concept hierarchy. On the other hand, concepts described by necessary conditions only ($A \sqsubseteq C$) affect how a concept is classified, but have no effect on which concepts can be classified under them [Rec95].

2.1.2 Standard Reasoning Services

Among the several reasoning services provided by standard reasoners, we discuss the following [BHLS17]:

- **Satisfiability**: a concept C is satisfiable in $\mathcal{O}$ when there exists at least one model $\mathcal{I}$ of $\mathcal{O}$ that is non-empty ($C^{\mathcal{I}} \neq \emptyset$). When $C^{\mathcal{I}} = \emptyset$ for all models of $\mathcal{O}$ then C is unsatisfiable.
- **Entailment**: an axiom α is entailed by an ontology $\mathcal{O}$, denoted as $\mathcal{O} \models \alpha$, if in every model $\mathcal{I}$ of $\mathcal{O}$ we have that $\mathcal{I} \models \alpha$.
- **Consistency:** an ontology $\mathcal{O}$ is consistent when there is a model $\mathcal{I}$ that satisfies

every axiom in $\mathcal{O}$, i.e., $\mathcal{I} \models \mathcal{O}$.

- **Realisation:** a given instance a is entailed by an ontology $\mathcal{O}$ when $\mathcal{O} \models C(a)$ where C is any concept.

- **Classification**: is the task of computing all of the subsumption axioms between atomic concepts, deriving a concept hierarchy $\mathcal{H}$.

A concept hierarchy $\mathcal{H}$ is a finite set of subsumption axioms $A \sqsubseteq B$ such that $\mathcal{O} \models A \sqsubseteq B$, where A and B are concept names in sig($\mathcal{O}$). A concept hierarchy is a partial order of inclusion relations $A \sqsubseteq B$.

The transitive closure of the concept hierarchy $\mathcal{H}$ is the set of all subsumption axioms representing all possible paths between concept names in $\mathcal{H}$. For example, the transitive closure of the concept hierarchy $\{A \sqsubseteq B, B \sqsubseteq C\}$ is $\mathcal{H}$ itself and the axiom $A \sqsubseteq C$, which was formed under the transitive closure from A to C.

In $A \sqsubseteq B$, the concept A is the subconcept (i.e., child concept) of B, while the concept B is the superconcept (i.e., the parent concept) of A.

2.2 OWL: The Web Ontology Language

The Web Ontology Language (OWL) is a machine-processing oriented language developed by the World Wide Web Consortium (W3C) [W3C12]. The language was extended to a second version (OWL 2) [GHM+08], whose semantics are built on description logics.

OWL documents are expressed in a format built on the XML standards for objects, referred to as Resource Description Framework (RDF). The RDF language allows query languages like SPARQL to retrieve and manipulate data from ontologies. The abbreviation SPARQL represents SPARQL Protocol and RDF Query Language [PS08].

Depending on the semantics expressed by OWL, different variants of OWL 2 exist; we will discuss two of them below:

- OWL 2 EL: based on $\mathcal{EL}$++ DL, which has PTime worst case complexity for reasoning tasks [BBL05].

- OWL 2 DL: built on top of the $\mathcal{SROIQ}$ DL. For several inference services, the complexity of processing OWL 2 DL reaches 2NExpTime [HKS06].

In addition to the semantics of classical description logics, OWL builds on DLs with concrete domains to enable the description of concrete properties of concepts and instances, such as age, weight, height, and temperature [Baa00, BH91, Lut99, Lut02]. To make use of concrete domains, OWL includes what are known as data properties, which assign a concrete value of a datatype to a class or individual [MH08b]. The OWL specification accepts datatypes from the XML Schema Datatypes specification[1], the RDF specification[2], and the specification of plain literals[3].

OWL is widely utilised and has quickly become the standard for ontology building and data sharing. Several ontology editors such as Protégé, process inputs and outputs as ontologies in the OWL/XML (.owl) or RDF formats [GHM+08]. In fact, since the January 2019 version of SNOMED CT, the *stated relationships* file, written in a SNOMED CT special format, was replaced by the *OWL expression types* file including definitions specified in OWL expressions for concepts, and relationships of SNOMED CT [Intb]. This was for the purpose of allowing proper data representation when feeding data into DL reasoners, as well as OWL expression validation and ontology building.

Two types of axioms are allowed in OWL 2: logical axioms and annotation axioms (non-logical) [GHM+08]. Logical axioms are assertions that are affirmed to be true under well-defined semantics, enabling inference of axioms that are true but are not explicitly stated, referred to as entailments (or consequences). On the other hand, annotation axioms are analogous to comments in programming languages. They allow for the addition of additional (and optional) information about the ontology's concepts, roles or axioms.

OWL ontologies can be written and serialised in a variety of syntaxes, including RDF/XML (the primary exchange format), OWL/XML, Functional, Manchester, and Turtle.

The OWL language specification uses the terms classes, object properties, and individuals to refer to description logic concepts, roles, and instances.

2.3 Ontology Tools Used in this Thesis

This section describes the tools that were used throughout the project of the thesis.

[1] `http://www.w3.org/TR/xmlschema11-2/`
[2] `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`
[3] `http://www.w3.org/TR/rdf-plain-literal-1/`

2.3.1 OWL API

The OWL API is a high-level Application Programming Interface (API) that supports a variety of ontology-related functions, including loading, creating, managing, and saving ontologies [HB09]. Its general-purpose reasoning functionality enables the construction of numerous OWL editors and reasoners by allowing for the use of ontologies in various reasoner implementations [HB09]. Additionally, it supports the more recent second version (OWL 2).

The OWL API is based entirely on the OWL 2 structural standard. This permits the provision of a high-level view without being bound by a specific syntax structure, in contrast to other APIs such as the Jena API [CDD+04], which display class expressions and axioms as RDF triples [HB09]. Additionally, the OWL API can automatically parses all of the syntaxes that it processes, eliminating the need to install a syntax-specific parser [HB09]. This can permit editors based on the OWL API to facilitate conversion between multiple syntaxes.

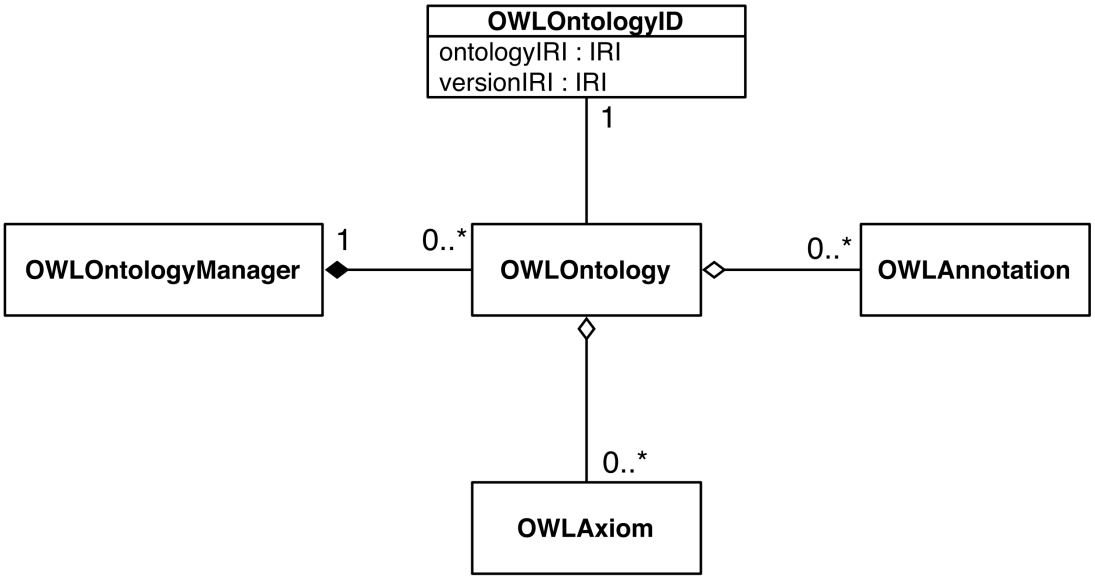

Figure 2.2: OWL API Interfaces [HB09]

The API architecture consists of a set of *interfaces* to manage ontologies, where the main interface is the OWLOntology interface (see Figure 2.2). The OWLOntology interface has a number of OWLAxiom objects that can be used to obtain information about an ontology's axioms, for example, whether an ontology contains an axiom referring to a particular concept. It also contain a number of OWLAnnotation objects in

order to access, create and change an ontology's labels (or annotations).

Each ontology is created, loaded, or changed via the OWLOntologyManager interface. This interface provides a central point for creating, loading, changing and saving ontologies. Moreover, it enables client applications to have a single point of access to ontologies, to implement redirection procedures and other customisations for ontology loading, and to monitor all changes made to any loaded ontologies.

The classes, object properties or individuals of an OWL ontology can be recognised by the IRI, via the OWLOntologyID interface. Internationalized Resource Identifiers (IRIs) are used to identify ontologies and their components. An IRI is an absolute (not relative) address that refers to a specific ontology entity [OWL]. Consider the address: `http://www.owl-ontologies.com/travel.owl#Accommodation`, which refers to the Accommodation class in the travel ontology [Knu].

The OWL API provides an OWLReasoner interface that is used to perform reasoning tasks on ontologies such as consistency checking, computing class or object property hierarchies, and axiom entailment. Incremental reasoning is supported via the OWLReasoner interface, which enables an initialised reasoner to listen for changes as they occur. For example, an ontology may be modified within an ontology editor, and a reasoner should respond to ontology changes as they arrive for further classification or reasoning services.

Different existing reasoners provide libraries to be controlled via the OWL API such as ELK [KKS12b], HermiT [GHM+14] and CEL [BLS06].

The OWL API provides additional capabilities, such as the generation of explanation axioms and modularisation. These explanation axioms are the minimal set of axioms required to hold an entailment. The OWL API supports generating explanation axioms which enables ontology users and developers to easily integrate basic explanation functionality into client applications. Additionally, the OWL API includes a library for extracting syntactic locality-based modules from ontologies [GHKS07], which are modules built on the notion of conservative extensions [GLW06]. Such notion ensures that entailments of an ontology are preserved over a particular signature. (For an overview of the literature on modularisation, see Section 3.1.1).

The OWL API was primarily used to develop the methods in this thesis, specifically the focus set subontology generator in Chapter 5, the modularisation-uniform interpolation workflow in Chapter 4, and the focus set semantic difference generator in Chapter 6. The development was based on the OWL API in order to take advantage of the API's various functionalities, including loading, invoking reasoner functions

bundled with the ELK and Hermit reasoners, and saving the output.

2.3.2 Standard Reasoners

ELK

The ELK reasoner was developed to provide comprehensive reasoning services of OWL EL capabilities, high reasoning performance, and ease of expansion and use [KKS14]. The reasoner is based on goal-directed inference rules and eliminate redundant inferences without violating completeness. It employs a consequence-based reasoning engine that is capable of performing parallel inferences, taking advantage of existing multiprocessor platforms [KKS14]. As an example of the reasoner's performance, it can classify SNOMED CT in less than 5 seconds on a quad core machine. The reasoner supports $\mathcal{ELH}$ TBoxes and ABoxes extended with reflexive roles, transitive roles and role chain axioms.

ELK has been utilised in a range of biomedical applications since its 2011 release, including the development of different ontologies such as, the neuroanatomy of flies [ORM+12] and the Ontology for Biomedical Investigations (OBI) [BBB+16]. Moreover, it has been used for integrating databases of diseases, genes, and medications [HLB+13, HDG12, HHH+12], and validating and querying genetic ontologies [JSH12, The18].

Additionally, the reasoner was employed as a building block in the development of various systems, such as the method proposed in [AKM+17] for learning features from biological knowledge graphs, and the method in [SGH19] which generates vector representations of biological entities in ontologies. Other examples of systems that uses the ELK reasoner are [DBLM15, JBD+19, DBF+14, HAG+16].

The reasoner can be used as a standalone tool, a Java-based library or as a plugin within the well known ontology editor, Protégé [KKS14]. The Java-based library binds the ELK reasoner to the OWL API and can be used as a library managed via the OWL API.[4]

The ELK reasoner is the backbone of our focus set subontology generator (Section 5.2) and the subontologies verifier (Section 5.3).

[4] `https://github.com/liveontologies/elk-reasoner`

HermiT

The HermiT reasoner supports all of the features of the OWL 2 ontology language [GHM+08], including all of the OWL 2 datatypes [MH08b]. Along with concept classification, the reasoner supports object and data property classification, which no other OWL reasoner completely supports [GHM+14].

Additionally, HermiT supports SPARQL query answering and makes use of a variety of optimisations such as *blocking via signature caching*, *individual reuse*, and *core blocking* to ensure that real-world ontologies are processed efficiently [GHM+14]. [5]

HermiT is built on the hypertableau calculus [MSH14]; a hybrid calculus combining resolution and tableau that avoids the nondeterministic behaviour of existing OWL reasoners such as Pellet [SPG+07] and FaCT++ [TH06].

Due to its support for all features of the OWL 2 language, HermiT has been a critical component in the creation of a wide variety of systems that make use of OWL 2 ontologies. Among these systems are the one described in [GEE17]; a recognition system that simulates an occupant's various activities, and the work described in [AKGBGS18], which demonstrated the use of a machine learning method that uses an unsupervised hybrid technique based on inductive and deductive reasoning to detect occupant behaviours and associated potential waste in real time.[6] Additional examples of work that made use of the HermiT reasoner can be found in [HZSBHM15, ZGN+15, HJRS+17, SGD+18].

The HermiT reasoner was used in our evaluation of the upwardly abstracted definitions in Chapter 5 (Section 5.5). Moreover, it was used to check entailments within our focus set semantic difference generator in Chapter 6 (Section 6.2).

[5] Blocking is a tableau reasoning technique used to detect cycles during the construction of ontology models in order to guarantee that only finite model abstractions are created. Blocking in HermiT is optimised using different strategies including *blocking via signature cashing*, *core blocking* and *individual reuse*. *Blocking via signature cashing* is a technique that works on ontologies without nominals and can significantly improve the performance of reasoning tasks involving multiple iterations of the hypertableau algorithm [GHM+14]. On the other hand, *core blocking* reduces reasoning times by identifying core individuals [GHM10]. Such strategy reduces the number of tests needed to identify which individuals that block each other during model construction. *Individual reuse* is an optimisation technique that deals with existential assertions [MH08a]. This technique is a tableau calculus that attempts to nondeterministically reuse individuals from previously generated model individuals to satisfy existential assertions. This aids in minimising the size of the constructed pre-model.

[6] *Unsupervised learning* is a machine learning method, which learns from data without explicit intervention through manual annotation or provision of training data [HS99]. *Inductive reasoning* is a form of reasoning that draws generalisations or behaviours from particular events and observations [CCF05]. *Deductive reasoning* is a style of reasoning that uses general principles or rules (i.e., conclusions) to predict specific outcomes (i.e., future observations) [Cum13].

2.3.3 Other Tools

Protégé

Developed as collaboration between the University of Manchester and Stanford University, the ontology editor, Protégé, provides a variety of services that facilitate the use of ontologies, including navigation of large ontologies, visualisation, components for viewing complex term relationships, and a programming interface for developing terminology-driven applications [Mus15].

Protégé is available as Java-based tool that provides a convenient "plugin" architecture. This plugin environment allows the tool to be extended with a variety of semantic web application services. Examples of these plungins are the *SeMCQ plugin* [TC09] for generating Semantic Multiple Choice Questions (SeMCQs) for any topic of knowledge, *the ProtegeVOWL* [LNB14]; a plugin that provides an accessible and interactive visualisation that is also easily understood by users who are unfamiliar with ontologies, the *Protege-TS plugin* [HS20]; supporting a number of operations that enable users to create and adjust signatures of ontology concept and role names, and the *BioPortal plugin* [NTW+11], which supports the import of entities from ontologies and terminologies stored in the BioPortal repository.

Protégé is also available via the Web as a collaborative ontology editing application [HMT+13]. This web based editor is called "WebProtégé 2". It acts as "Google Docs" environment for ontologies, allowing users to upload their ontologies and collaborate on ontology projects. It includes a variety of tools for collaborative ontology editing, including ontology change tracking, alerting, and issue monitoring. At any moment in time, users can obtain a snapshot of an ontology. Additionally, WebProtégé 2 includes an intuitive user interface for modifying axioms and class constructors, which is well-fitted for biomedical ontologies that often adhere to the lightweight OWL 2 EL profile.

The Protégé editor was mostly utilised to analyse the results of our experiments in Chapters 4, 5 and 6.

OWL Justification Tool

The tool can be used by end-users for explaining entailments in their ontologies, which is available as a Protégé plugin and a Java-library. Generating justification or explanation axioms can be useful for debugging ontologies or understanding the reason of consequences (entailments) of the ontology.

A justification $\mathcal{J}$ of an entailment α is defined as a $\subseteq$-minimal subset of an ontology $\mathcal{O}$ and it is sufficient for α to hold, that is $\mathcal{J} \subseteq \mathcal{O}$, $\mathcal{J} \models \alpha$ and there is no $\mathcal{J}' \subset \mathcal{J}$ such that $\mathcal{J}' \models \alpha$ [KPHS07].

We utilised the OWL Justification tool to generate explanatory axioms for several of our results in Chapter 6's experiments.

2.4 The Medical Ontology SNOMED CT

The College of American Pathologists (CAP) created a committee in 1955 to develop anatomic pathology terminology. Ten years later, the CAP released the first nomenclature for anatomic pathology, titled Systemizes Pathology Nomenclature (SNOP). This vocabulary defined four categories of terms linked with pathology findings: *topography* (affected anatomic site), *morphology* (structural changes associated with disease), *aetiology* (cause of disease), and *function* (physiologic alterations associated with disease). SNOP was the first multi-axial coding system to be employed in the healthcare industry [BG16].

SNOP was expanded in 1975 to form the Systematized Nomenclature of Medicine (SNOMED). This nomenclature had a greater amount of information, describing diseases and procedures. In 1979, a second edition of SNOMED was published, which included a total of seven axes for characterising various medical terminology, including topography, abnormal anatomy, etiology, function, procedure, and occupation [CR80]. By 1993, SNOMED III had been developed and expanded to SNOMED 3.5 in 1998, with revisions to terms totaling 11 axes and 157,000 records [BG16].

CAP then produced a logic-based version of SNOMED 3.5, which was published in 2000 as SNOMED Reference Terminology (RT) [SCC97,Spa00a]. By 2002, SNOMED CT had been produced by the merging of SNOMED RT and Clinical Terms Version 3 (CTV3); a code schema developed in the United Kingdom at the national health care centre [Spa00b, WBB+01]. Following the merger, the first release of SNOMED CT comprised over 300,000 medical terms, which was considered a significant increase. It was initially released in January 2002 and has been upgraded on average every six months since then.

Since 2002, SNOMED CT has been a comprehensive and widely utilised medical ontology covering a wide variety of clinical specialties and requirements.[7] It defines

[7] `https://www.snomed.org/`

standard medical terms used in health records with the goal of facilitating interoperability between health care systems in multiple countries such as Australia, Canada and UK. As a result, various country extensions exist for SNOMED CT that are based on the core (international) edition. These extensions specify additional terms that are country-specific.

SNOMED CT is actively managed, curated and distributed by SNOMED International (aka International Health Terminology Standards Development Organisation (IHTSDO)). The July 2017 version of the international edition of SNOMED CT contained 335 245 concept names, 97 role names and 335 225 logical axioms.

SNOMED CT contains codes, concepts, and definitions for medical concepts that are commonly used in clinical documentation [Spa00b, Spa00a, SCC97]. In the terminology, each concept is identified by a unique concept identifier, and the concepts are related to one another by a set of relations between them. One of the most fundamental relationships is the *is-a* relationship, which connects a specific concept to a more generic concept [Intc].

SNOMED CT consists of an acyclic directed graph. By utilising *is-a* relations, concepts in SNOMED CT form a polyhierarchical structure of relationships. The root concept, called *SNOMED CT Concept*, is the superconcept of all other concepts and all other concepts are subconcept of the root concept [Intc]. More than one *is-a* relationship can exist between SNOMED CT concepts, and it is highly typical for a concept to have several alternative paths that lead to a more generic concept.

A total of 19 *top level concepts* are defined immediately below the root concept. This results in the SNOMED CT generic structure, which has an overall hierarchical structure of 19 concept sub-hierarchies. The deeper one traverses along a concept sub-hierarchy, the more granular the concept becomes. Figure 2.3 shows the top level concepts that are subsumed by the root concept *SNOMED CT concept*. As can be seen in the figure, top-level concepts in SNOMED CT occur immediately under the root concept *SNOMED CT Concept*. The 19 top-level concepts are:

1. Body structure (body structure),
2. Clinical finding (finding),
3. Environment or geographical location (environment / location),
4. Event (event),
5. Observable entity (observable entity),

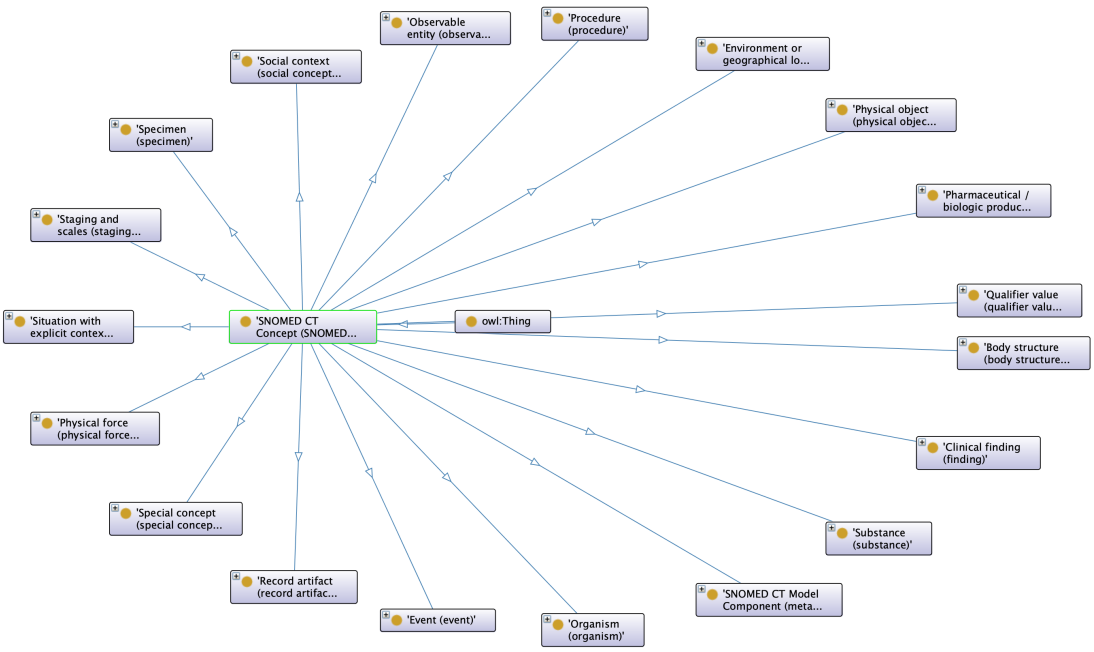

Figure 2.3: The 19 top level concepts beneath the root concept *SNOMED CT concept*, visualised using the OntoGraf plugin in Protégé [Fal].

6. Organism (organism),
7. Pharmaceutical / biologic product (product),
8. Physical force (physical force),
9. Physical object (physical object),
10. Procedure (procedure),
11. Qualifier value (qualifier value),
12. Record artifact (record artifact),
13. Situation with explicit context (situation),
14. SNOMED CT Model Component (metadata),
15. Social context (social concept),
16. Special concept (special concept),
17. Specimen (specimen),

18. Staging and scales (staging scale) and

19. Substance (substance)

Another type of relationships is called *attribute relationships*. Attribute relationships are used to illustrate a characteristic of a concept's meaning, known as a *defining characteristic*. These defining characteristics are analogous to existential restrictions in the context of description logics. The *domain*, attribute, and *range* of the attribute make up attribute relationships. The concept for which the attribute is applicable is the *domain* concept, while the concept that applies to the attribute is the *range* concept [Intd]. Figure 2.4 shows an example of two defining characteristics exist in the definition of the concept *Pneumonitis (disorder)*. Because of the attribute relationships, concepts that belong to different concept sub-hierarchies are interconnected [Intc].

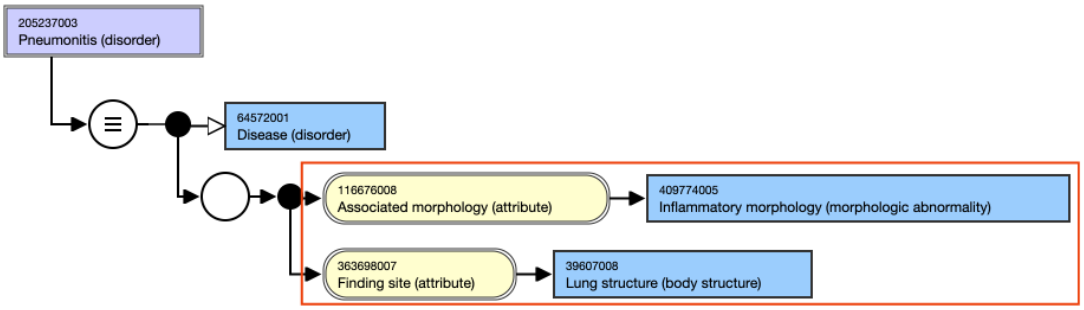

Figure 2.4: Two attribute relationships (defining characteristics) in the definition of *Pneumonitis (disorder)* within the red box. The axiom is viewed using the official SNOMED CT browser.

Attribute relationships in SNOMED CT are grouped using a unique role called *RoleGroup* [SDMW02, CS09]. The use of this role is particularly useful to group attribute relationships that must be be linked with one another. For instance, in the definition of *Tetralogy of Fallot*, the finding site: *right ventricle* is paired with its associated morphology *hypertropy*. As a result, correct inferences and semantic meaning of concept definitions is maintained, specially when a concept relationship contains more than one attribute of the same type such as multiple finding sites and multiple morphologies [SDMW02].

Due to the existence of RoleGroup, nested existential restrictions occur in concept axioms, which can result in expensive computation when inference services are utilised on the ontology. Additionally, the RoleGroup's existence results in existential

Table 2.4: Mapping of SNOMED CT expressions to their corresponding DL expressions

SNOMED CT Form	Description Logics Form
is-a relation	$\sqsubseteq$ (subsumption relation)
Attribute	role
Attribute relationship	Existential restriction ($\exists r.B$)
Primitive concept A	$A \sqsubseteq C$, C is a necessary condition of A
Fully defined concept A	$A \equiv C$, C is necessary and sufficient condition of A

restrictions with a maximum depth of two in every concept description that appears in SNOMED CT [Intd].

Concepts in SNOMED CT have two statuses, they are either *primitive* or *fully defined*. A concept is fully defined if the defining characteristics (attribute relationships) of a concept are sufficient to distinguish its meaning from other concepts, otherwise the concept is primitive [Inte].

The logical profile of SNOMED CT is $\mathcal{ELH}$. As of the January 2018 release, the SNOMED CT terminology has begun to support higher expressive language constructors, including role chains, transitive and reflexive roles, as well as GCIs of the type $C \sqsubseteq A$ where A is a concept name and C is an $\mathcal{EL}$-concept [Intf].

In SNOMED CT, *is-a* relations correspond to *subsumption* relationships in DL form. The *attributes* in SNOMED CT correspond to *roles* in DL form. Table 2.4 summarises the SNOMED CT expressions and their corresponding form in description logics.

The SNOMED CT relationships can be browsed via the SNOMED CT official browser,[8] which is developed and maintained by SNOMED International. The browser allows users to search, navigate, and view terminology content of the international edition and the national extensions used in several SNOMED CT member countries. The SNOMED CT browser is capable of displaying terminological attributes at various levels of detail and makes use of other SNOMED CT standards, including the SNOMED CT compositional grammar for expression generation and the SNOMED CT diagramming standard for concept diagram display. Furthermore, the browser enables search result filtering as well as various user-specific customizations and preferences [Intg].

The SNOMED CT terminology is released every six months by SNOMED International. Three distinct types of release files exist, including [Inth]:

- Full release files: These contain every version of every SNOMED CT component

[8] `https://browser.ihtsdotools.org/`

such as concepts, attributes, and relationships. Among other uses of the full release, it is used for implementations that require a comprehensive history of SNOMED CT's components.

- Snapshot release files: These include the most recent version of each component in SNOMED CT. The snapshopt release is advantageous when only the most recent version of the components is required.

- Delta release files: These contain just changes made to the current release version since the previous one. This is advantageous when an older release must be upgraded to the current one.

Release files are written in a special format called RF2 (release-format 2). Prior to January 2012, the release files were written in the RF1 format. Since then, the release files have been modified to RF2, which is an updated version of the previous format [Inti]. Among the different files that are contained in each release type, we discuss two of these files. The first one is the *stated relationships file*, which has a list of asserted relations that have been authored using an authoring modelling approach. The second file is the *inferred relationships file*. This file contains the inferred view of SNOMED CT, which is constructed mostly through the use of a description logic reasoner [Inth].

The concept definitions and inclusions in SNOMED CT are modelled according to different types of modelling approaches such as (a) the proximal primitive modelling, (b) compose and refine modelling, and (c) maximal modelling [Cha]. The proximal primitive modelling is a type of modelling used to model concept relationships (definitions and inclusions) by assigning the immediate proximal primitive parent and attribute relationships based on their relevance to the concept's defining characteristics. This is an alternative approach to relying on a concept's inheritance and refinement of relevant attributes from immediate, defined super concepts [Intj]. This modelling has been standardised, and is used in the international edition. SNOMED International is refining and implementing a quality assurance project for concept definitions that do not conform to such modelling [Intj]. Our upwardly abstracted definitions are inspired by such modelling to meet SNOMED CT requirements in providing full definitions for the input focus symbols (see Chapter 5).

2.4.1 Normal forms

Normal forms are used to state concept relationships (axioms) that are included in the stated relationships file [Spa01]. After generating the stated relationships file, a description logic reasoner is used to build the inferred view of SNOMED CT, which is then included in the inferred relationship file [Intk]. When a reasoner is applied to concept relationships, it produces a normal form of such relationships that is not identical to the normal form stated initially (while generating the stated relationship file). These normal forms represent the conjunction of concept names and existential restrictions (defining characteristics) of the concept defined by any of the following normal forms. Such normal forms differ syntactically but are logically equivalent.

We now discuss some of these normal forms [Spa01].

Short canonical form. This form is compatible with proximal primitive modelling. The form consists of possible conjunctions of proximal primitive concept names and existential restrictions (defining characteristics). As mentioned previously, all new concepts in the international version of SNOMED CT are modelled using the proximal primitive modelling, which results in the short canonical form.

Long canonical form. This is similar to the short canonical form, however, it allows possible redundant existential restrictions in order to include all of the concept's defining characteristics (existential restrictions). This form is inferred, i.e., it is generated after applying a description logics reasoner in order to include all of the concept's defining characteristics. According to [Spa01], such normal forms that were introduced among others have a number of advantages in the area of SNOMED CT authoring and distribution. The studies in [DSM02, SRP+13] highlight the utility of the long canonical normal form for SNOMED CT query-based applications. Such form can help in eliminating redundant concept representations when querying or retrieving clinical information using the selective retrieval approach described in [DSM02] [HG12].

The long canonical form was adopted in SNOMED CT RF1 release files. However, after January 2012, when RF1 was replaced by RF2, it is no longer utilised in the various relationships files.

Distribution normal form. Another name for this form is the *Necessary Normal Form* as described in [Intk]. This form is used in the *inferred relationships file*. It is generated as a result of applying a description logics reasoner on the stated concept

axioms, which were written in *short canonical form*. The form is similar to the long canonical form, in which it lists all of the defining characteristics of the concept name. Here, the concept is defined in terms of the proximal parent, regardless as to whether it is *defined* or *primitive* [Intk].

2.4.2 Reference sets

Reference sets (or *refsets*) were developed as a means to manage SNOMED CT extensions, data structures, and release formats. Reference sets enable the creation of subsets of SNOMED CT content that are relevant to specific clinical area of information. This increases the clinical utility of SNOMED CT by limiting the recording, retrieval, and processing of information in an electronic health record (EHR) to a specific subset of content [Intl].

Examples of standard reference sets includes the ERA-EDTA refset which contain concepts about renal diseases [VRTG+12] (see below for further information about the ERA refset). The use of the ERA refset is illustrated in [VRTG+12], which demonstrates that a clinical terminology system typically provides a list of code choices each of which has a corresponding meaning in the source ontology. These are the codes that the ERA refset provides. The system allocates a code to the patient in accordance with the patient conditions entered by the doctor.

In most cases, standard reference sets contain concept names that are descendant concepts of one of the SNOMED CT top-level concept hierarchies. For example, as described in [VRTG+12], concepts contained inside the ERA refset were established carefully and have working definitions about renal diseases in the source ontology. Other concepts such as those describing the *finding sites* of renal diseases are not included in the refset. Therefore, concepts in these refsets work as seed symbols for extracting concept definitions (the right hand side) from the source ontology. Additionally, as can be seen from Table 2.5, these refsets often contain a small number of concepts in comparison to the size of concept names in the original ontology (SNOMED CT), which has over 335 000 concept names.

Throughout our experiments in Chapters 4, 5, and 6, we use a number of refsets. Some of these refsets were provided by one of the SNOMED CT leading terminologists, while others were already publicly available but were revised with feedback from the terminologist. These reference sets are listed as below. Table 2.5 summarises our analysis of the refsets (see the first column for the refset name). The second column indicates the version of the refsets that were used. The third column indicates the

proportion of concepts that belongs under a top-level concept hierarchy of SNOMED CT. The last column indicates the size of these refsets, i.e., the number of concepts contained within them.

NHS refsets The National Health Services Digital (NHS Digital) in the UK provides release files, refsets, mappings, and other supporting documentation for users who use SNOMED CT UK Edition in their systems.[9] These are offered in a variety of bundles to assist in identifying the sets of files required [Dig]. The NHS refsets' purpose is to facilitate the focusing on certain set of concepts for a particular requirement, for instance, concentrating on concepts pertaining to ophthalmology clinic operations [nhs].

CSP The Chartered Society of Physiotherapy in the UK has developed a set of SNOMED CT refsets to support physiotherapists. The purpose is to use the refsets to serve as the foundation for their Electronic Health Records (EHR) in any clinical settings [oPa]. The list of refsets is published publicly at [oPb]. The total number of produced refsets is 10, and they cover physiotherapy concepts such as mobility and gait. All of the concepts contained within the ten refsets are classified as *clinical finding* concepts.

Medical conditions The experiments of [ACSDD+19] used a list of medical conditions that affect humans and animals including *heart failure*, *asthma*, *epilepsy*, *glaucoma*, *chronic kidney disease*, *osteoarthritis*, *anaemia*, *arthritis*, *diabetes*, *hypertension*, and *obesity*. The development of these terms was done by mapping SNOMED CT concept identifiers (that correspond to a medical condition) to Unified Medical Language System (UMLS) Concept Unique Identifier (CUI) pairs validated with BMJ Best Practice content [Pra]. The supplementary information of [ACSDD+19]'s experiments includes a list of 19–39 concept names for each medical condition.[10]

ERA-EDTA The European Renal Association-European Dialysis and Transplant Association (ERA-EDTA) registry maintains a subset of concept names about renal diseases. The refset can be used by affiliated registries (renal centres). The concepts in the refset are aligned with international coding standards supported by the World Health Organization International Classification of Diseases (WHO ICD) and SNOMED CT

[9] `https://isd.digital.nhs.uk/trud/users/guest/filters/0/home`
[10] `https://tinyurl.com/medical-conditions-signature`

[VRTG+12]. All of the concepts within this refset reside under the *clinical finding* concept hierarchy.

MRI The list of concept names about Magnetic Resonance Imaging (MRI) branches from the *Procedure* concept hierarchy in SNOMED CT [Intm]. In 2009, among other lists related to imaging modalities, the MRI list was adopted as the national standard for the UK and has since been mandated for the country's Diagnostic Imaging Dataset (DID) [Eng] [GAB]. All of the concepts within this refset are under the *Procedure* concept hierarchy.

GPFP The General Practitioner/Family Practitioner (GPFP) reference set contains SNOMED CT concept names pertaining to two data categories that are often used in electronic health records for general/family practice, including (i) Reasons For Encounter (RFEs) and (ii) health issues [Intn]. Concepts in this refset belong to four main concept hierarchies in SNOMED CT, namely *Clinical finding*, *Event*, *Procedure* and *Situation with explicit context*. We analysed the number of concepts from January 2016 to July 2017 versions of this refset to determine the percentage of concepts falling into each of these four primary concept hierarchies. Our analysis shows that one of the concepts in the GPFP refset belongs to the *Body structure* concept hierarchy in each refset version (as shown in Table 2.5).

ICNP Diagnoses The International Classification of Nursing Practice Diagnoses (ICNP Diagnoses)[11] refset contains concepts that map to those in the nursing diagnoses (problems) refset, which was created to enhance terminology standardisation and interoperability in health information systems for nursing diagnoses [Into]. Concepts within this refset belong to two main categories of SNOMED CT including *Clinical finding* and *Situation with explicit context*. Table 2.5 shows how many concepts belong to the main concept hierarchies in versions January 2016 through to July 2017.

ICNP Interventions The International Classification of Nursing Practice Interventions (ICNP Interventions) refset has the same function as the ICNP-Diagnosis refset [Intp]. As documented in [Intp], concepts within this refset only belong to the *Procedure* hierarchy in SNOMED CT. However, our analysis revealed that 4% out of

[11] `https://www.icn.ch/what-we-do/projects/ehealth-icnptm/about-icnp`

the total number of concepts in this refset belong to the *Situation with explicit context* concept hierarchy (see Table 2.5).

GD A group of SNOMED International Dentistry (SIG) members based in the United States collaborated to develop a set of diagnostic terms that are standarised and could be simply understood by all dental doctors and simple to adopt for electronic records system vendors. The General Dentistry (GD) set of diagnostic terms is intended to offer dentistry with a set of terms that encompasses the majority of dental care [Intq].

Table 2.5: SNOMED CT Refsets

SNOMED CT Refset	Version	Top-Level Concept Hierarchy	Size
ERA-EDTA	201707	Clinical findings (100%)	177
MRI	201707	Procedure (100%)	590
General Dentistry	201707	Clinical findings (89.38%) Body structure (5.31%) Procedure (0.88%) Qualifier value (0.88%) Observable entity (0.44%)	226
GPFP	201607	Clinical findings (86.44%) Procedure (10.79%) Situation with explicit context (1.55%) Event (1.20%) Body Structure (0.02%)	4330
	201607	Clinical findings (86.44%) Procedure (10.79%) Situation with explicit context (1.55%) Event (1.20%) Body Structure (0.02%)	4330
	201701	Clinical findings (86.43%) Procedure (10.79%) Situation with explicit context (1.55%) Event (1.20%) Body Structure (0.02%)	4327
	201707	Clinical findings (86.60%) Procedure (10.65%) Situation with explicit context (1.53%) Event (1.19%) Body Structure (0.02%)	4374
ICNP	201601	Procedure (99.38%) Situation with explicit context (0.62%)	486
	201607	Procedure (99.38%) Situation with explicit context (0.62%)	485
	201701	Procedure (99.38%) Situation with explicit context (0.62%)	486
	201707	Procedure (98.68%) Situation with explicit context (1.32%)	529
ICNP-Interventions	201601	Procedure (96.43%) Situation with explicit context (3.57%)	953
	201607	Procedure (96.43%) Situation with explicit context (3.57%)	953
	201701	Procedure (96.43%) Situation with explicit context (3.57%)	953
	201707	Procedure (95.96%) Situation with explicit context (4.04%)	991

2.4.3 SNOMED CT Versioning

SNOMED International keeps the SNOMED CT international (core) edition up to date with frequent updates. A standard SNOMED CT release is published every year on the last day of January and July. Interim releases are occasionally distributed in specific months in response to specific occurrences; for example, the COVID-19 situation resulted in the March 2020 interim release, which incorporates updates relating to COVID-19-specific terminology [Intr]. Likewise, country members' extensions of international editions are updated on a regular basis by their respective organisations. SNOMED CT is updated on a regular basis for a variety of reasons, including maintaining the content current in light of new research, new healthcare approaches, and other changes in clinical practise. Another purpose is to broaden the formal definitions of primitive concepts in order to ensure that they are sufficiently defined, and to remedy flaws or inconsistencies that may have been discovered by the SNOMED CT community [Ints].

One of the release files of SNOMED CT is the *Delta* release (see Section 2.4 for an overview about the different types of releases in SNOMED CT). It is useful for recognising changes made to the current release since the prior one, such as what has been added or altered. In terms of reading changes that has been lost from the current release since the previous one, it is not possible to read changes of relationships that have been lost using the delta release alone. However, using the full release, one can generate the changes for a specific date range [Intt]. The delta release can be used to determine which concepts were active in a previous release but are no longer active in the current one. If a concept is not active in the current release, it is not used in the current release. Additionally, the delta release assists in determining whether a concept remains a primitive concept or has evolved into a defined concept (sufficiently defined) (or vice versa) in the current release.

The changes in the delta files are based on a *structural difference* rather than a semantic difference [OCP17]. (See Section 3.2 for an overview about ontology diffing methods).

2.5 Other Ontologies

During the evaluations of our methods, we have used biomedical ontologies other than SNOMED CT including the Gene ontology and the NCIt ontology described below.

Gene Ontology The Gene Ontology (GO) is a significant project in the biological domain aimed at standardising the representation of concepts related to genes and gene products across all organisms [DS17]. Three main functional aspects of gene products are defined in GO including: biological processes, molecular functions and cellular components. The main usage of the gene ontology is to associate gene products to their functions using the knowledge within the ontology via gene annotations.

The latest version of GO is expressed in $\mathcal{ELHI}$ extended with transitive, reflexive roles and role chain axioms.

The developers and other users of GO created subset/slim files from the gene ontology [The]. Each subset is a flat list of entities that are specific to certain species or organisms. In basic areas of biology, GO subsets are useful for addressing particular research needs. For example, if a researcher is solely interested in a particular field of biology, such as ageing or fruit development, they may not require terms from other branches and may concentrate exclusively on those terms.

NCIt the National Cancer Institute (NCI) Thesaurus (NCIt) ontology is a thesaurus of biomedical terminology covering different cancer-related information [GFH+03].[12] The ontology is hierarchically organised into 19 distinct sub-domains related to cancer research, e.g., neoplastic diseases, molecular abnormalities, genes, and cancer vaccines [HJ15]. The thesaurus is updated on a monthly basis, and provides a range of vocabularies including cancer terms, a drug dictionary and genetic terms. The latest release 22.01e (2022 January) includes more than 23 000 axioms. The DL expressivity of the NCIt is $\mathcal{SH}$ ($\mathcal{ALC}$ with role hierarchies and transitive roles), though more than 99.9% axioms are formulated in $\mathcal{ELH}$.

Other biomedical ontologies expressed in $\mathcal{EL}$ are available through well-known repositories such as the Open Biomedical Ontologies (OBO) Foundry and the BioPortal repository.[13][14] The Chemical Entities of Biological Interest (ChEBI) ontology is one example of these ontologies.

CheBI [HOD+16] is an ontology of chemical entities with biological relevance that includes a large number of manually curated data items. In the ontology, there are three primary concept hierarchies: *chemical entity*, *role*, and *subatomic particle*. The concept hierarchy for *chemical entities* contains concepts that are categorised according to their shared structural characteristics. The *role* concept hierarchy contains concepts

[12] `https://ncit.nci.nih.gov`
[13] `https://obofoundry.org/`
[14] `https://www.bioontology.org/`

classified according to their biological and chemical activities. The *subatomic particle* concept hierarchy classifies concepts of particles that are smaller than atoms. The logical profile of the CheBI ontology is $\mathcal{ELHI}^+$ (October 2021 version), which is $\mathcal{EL}$ extended with role hierarchies, inverse roles, and transitive roles.

Another example of a biomedical ontology is the Sequence Ontology (SO), which is a controlled structured vocabulary for the various components of a genomic annotation to aid in the exchange, analysis, and management of genomic data [ELM+05]. The logical profile of the Sequence ontology is $\mathcal{ELHI}$.

Chapter 3

Ontology Extraction and Difference Computation

In this chapter, we review state of the art ontology engineering methods in the extraction and diffing areas. The review is mainly focused on the extraction area, since it is the main focus of this thesis.

3.1 Extraction Methods

Ontology extraction is the process of extracting a segment, or a module from an ontology according to preset criteria. Because of the variety of knowledge representation methodologies and the required qualities of the resulting extracts, the literature on ontology extraction is rich with such methods. The approaches to generating ontology extracts vary according to whether the extract is intended to ensure proper knowledge encapsulation or not [SS20]. Without logical guarantees, module extraction and decomposition techniques frequently rely on some type of syntactic traversal of the ontology's class hierarchy [NM09, SK04, SR06]. In this section, we discuss notions of knowledge extraction that preserve the knowledge of the original ontology within the extract for a given set of input symbols. Several notions in the literature exist that meet the knowledge preservation criteria. Among these notions, we focus in this section on three notions of modularisation as well as the notion of uniform interpolation.

3.1.1 Modularisation

Module extraction computes a subset of the original ontology that captures all the consequences over the input signature. The below definition defines generally what a module means. The definition applies to OWL 2 ontologies, i.e., those that are expressed in $\mathcal{SROIQ}$.

Definition 2 (Module [Tsa12])**.** *Let $\mathcal{O}$ be an OWL 2 ontology and Σ be a signature. A subset $\mathcal{M}$ of the ontology is a* module *of $\mathcal{O}$ for Σ if $\mathcal{M} \models \alpha$ iff $\mathcal{O} \models \alpha$ for every axiom α with $sig(\alpha) \subseteq \Sigma$.*

Several approaches exist to extracting modules. Among these approaches are locality-based modularisation, semantic modularisation and minimal subsumption modularisation. The latter two notions are defined according to Σ-inseparability notion (see Definition 5). The condition in Definition 2 applies to the three module notions.

Locality Based Modularisation (LBM)

Locality-based modules are built on the notion of conservative extensions [GLW06], which guarantees the preservation of ontology entailments over a given signature.

To identify which axioms to include in the module for a given set of symbols Σ, the method determines locality of axioms in the input ontology with respect to the input signature Σ. Locality of axioms is defined as follows.

Definition 3 (Locality [Tsa12])**.** *An axiom α is called $\top(\bot)$-*local *w.r.t a signature Σ if replacing all named entities in $sig(\alpha)\backslash\Sigma$ with $\top$ (respectively $\bot$) makes that axiom a tautology. An axiom α is called a* tautology *if it is* local *w.r.t $sig(\alpha)$. An axiom α is called global if it is non-local with w.r.t. $\emptyset$.*

Determining the locality of axioms can be done by checking the entailment of α by the empty ontology i.e., $\emptyset \models \alpha$. This check can be done *semantically* by using a reasoner or *syntactically* by parsing the ontology axioms [GHKS08]. Syntactic locality attempts to simulate the entailment check by exploring the axiom structure to determine locality [GHKS08].

Locality-based modules are defined as follows.

Definition 4 (Locality-based module [GHKS08])**.** *Let $\mathcal{O}$ be an OWL 2 ontology and Σ a signature set. We say that $\mathcal{M} \subseteq \mathcal{O}$ is a* locality-based module *of $\mathcal{O}$ w.r.t Σ when all the axioms in $\mathcal{O}\backslash\mathcal{M}$ are local w.r.t. the signature $\Sigma \cup sig(\mathcal{M})$.*

The syntactic type of Locality-Based Modularisation (LBM), which we referred to as SLBM, is available as part of the OWL API tool [HB09] and is frequently used to extract subsets of an ontology for further localized, and frequently easier, processing with other tools such as reasoning, querying, retrieval, ontology mapping, reuse and import [JGS+08, DTPP09, HS18, SSZ09, vDQCF18, QMMAN+18, ACSDD+19].

LBM accepts ontologies expressed in OWL 2. The method takes as input an ontology $\mathcal{O}$ and a seed signature Σ, and supports computing three distinct module types: bottom ($\bot$), top ($\top$), and nested ($\bot\top*$) modules. In essence, the method internally extends Σ to cover the upward (downward) views of Σ until it reaches the top (bottom) symbol in $\mathcal{O}$, as specified by the $\bot$ ($\top$) types. Specifically, the $\bot$-type of LBM computes a module that includes all super symbols of the input seed signature Σ. This results in the upward view of the symbols in Σ from the input ontology $\mathcal{O}$. The $\top$-type is the total opposite of the $\bot$-type. It generates a module containing all sub symbols associated with the symbols in the input seed signature Σ. The $\bot\top*$-type is an iteration of the $\top$ and $\bot$ types. For a given set of seed signature Σ the method computes a module by first computing a $\top$-module, then from the resulting $\top$-module it computes a $\bot$-module. The iteration stops when the size of the module of the previous round ($\mathcal{M}_i$) is equal to the size of the module of the next round ($\mathcal{M}_{i+1}$). This iteration results in a module that is smaller in size than modules generated for the $\bot$-($\top$)-types [VKP+13].

The following example illustrates the three different locality-based modularisation types.

Example 1. *Let*

$$\begin{aligned}\mathcal{O} = \{&A \sqsubseteq B_1,\\ &B_1 \sqsubseteq B_2,\\ &B_2 \sqsubseteq B_3,\\ &A_2 \sqsubseteq A,\\ &A_3 \sqsubseteq A_2,\\ &A_4 \sqsubseteq A_3\}.\end{aligned}$$

and $\Sigma = \{A\}$. *Extracting a* $\bot$*-module from* $\mathcal{O}$ *for* Σ *gives the following* $\bot$*-module*

($\mathcal{M}_\bot$), where the module contains axioms that represent all the upward relations (super concepts) of the concept A in Σ.

$$\begin{aligned}\mathcal{M}_\bot = \{&A \sqsubseteq B_1,\\ &B_1 \sqsubseteq B_2,\\ &B_2 \sqsubseteq B_3\}.\end{aligned}$$

In contrast, extracting a $\top$-module from $\mathcal{O}$ with the same signature yields the following $\top$-module ($\mathcal{M}_\top$), in which the axioms represent all the downward relations (sub concepts) of the concept A in Σ.

$$\begin{aligned}\mathcal{M}_\top = \{&A_2 \sqsubseteq A,\\ &A_3 \sqsubseteq A_2,\\ &A_4 \sqsubseteq A_3\}.\end{aligned}$$

Extracting the nested type results in an empty module. This is as a result of the iteration that begins with the extraction of an x-module where $x \in \{\top, \bot\}$. In the second iteration, an x-module is extracted, which is the opposite type of the module from the first iteration. Because the module from the second iteration was computed from the module of the first iteration which lacked the required concepts to compute a meaningful/non-empty module, the module from the second iteration is empty.

Algorithm 1 shows how a locality-based module is computed [SSZ09]. Computing a locality-based module starts with an empty module. Then, the algorithm checks if every axiom in the input ontology $\mathcal{O}$ is non-local w.r.t. to the current signature (Lines 4–7). In each repetition of this step, the signature Σ is augmented with the signature of the identified non-local axiom α (Line 7). Those non-local axioms α are added to the module $\mathcal{M}_x$. The iteration stops when the signature reaches a fixpoint (Line 8). Depending on the type of locality check chosen $x = \top$ or $x = \bot$, the generated module type is either a $\top$-module ($\mathcal{M}_\top$) or a $\bot$-module ($\mathcal{M}_\bot$). As previously mentioned, the step of determining locality in Line 5 can be performed syntactically or semantically.

Computing nested locality-based modules is done using Algorithm 2 [SSZ09]. The algorithm calls Algorithm 1 recursively to compute a nested locality-based module ($\mathcal{M}_\star$). The recursive function terminates when the size of the current nested module is equal to the size of the module from the previous call.

Algorithm 1 ComputeLocalityBasedModule($\mathcal{O}$, Σ)

Input: An OWL 2 ontology $\mathcal{O}$, Signature Σ
Output: Locality-based x-module $\mathcal{M}_x$ where $x \in \{\bot, \top\}$

1: $\mathcal{M}_x := \emptyset$, $\Sigma_0 := \emptyset$
2: **repeat**
3: $\quad \Sigma_0 := \Sigma$
4: $\quad$ **for** $\alpha \in \mathcal{O}$ **do**
5: $\quad\quad$ **if** $\alpha \notin \mathcal{M}_x$ and is not local w.r.t $\Sigma \cup \text{sig}(\mathcal{M}_x)$ **then**
6: $\quad\quad\quad \mathcal{M}_x := \mathcal{M}_x \cup \{\alpha\}$
7: $\quad\quad\quad \Sigma := \Sigma \cup \text{sig}(\alpha)$
8: **until** $\Sigma \neq \Sigma_0$
9: **return** $\mathcal{M}_x$

Algorithm 2 ComputeNestedLocalityBasedModule($\mathcal{O}$, Σ)

Input: An OWL 2 ontology $\mathcal{O}$, Signature Σ
Output: Locality-based $\top\bot*$-module $\mathcal{M}_\star$

1: $\mathcal{M}_\star :=$ ComputeLocalityBasedModule($\mathcal{O}, \Sigma$)
2: **if** $|\mathcal{M}_\star| \neq |\mathcal{O}|$ **then**
3: $\quad \mathcal{M}_\star :=$ ComputeNestedLocalityBasedModule($\mathcal{M}_\star, \Sigma$)
4: **return** $\mathcal{M}_\star$

The semantic type of LBM is excessively expensive to compute since it necessitates the employment of a reasoner capable of dealing with OWL 2 expressivity. SLBM, on the other hand, is less expensive to compute because it is based on the syntactic structure of the ontology and generates the module without utilising a reasoner. The underlying algorithm of SLBM runs in polynomial time in the size of the ontology [VKP+13].

Due to the fact that extracting syntactic locality-based modules is based primarily on the ontology's syntactic structure, extracting nested $\bot\top*$-modules may result in the loss of some of the related axioms to the set of input terms. For instance, if the input ontology states the axioms in the form (i) $A \sqsubseteq B \sqcap C$ or equivalently stated as (ii) $A \sqsubseteq B$, $A \sqsubseteq C$, the resulting nested module in the second case (ii) might exclude either $A \sqsubseteq B$ or $A \sqsubseteq C$ because B and C are not syntactically linked via $\sqcap$.

LBM retains the original forms of axioms in the resulting modules. However, for the purpose of extracting definitions of concepts in Σ, modules computed contain a significant number of symbols not in the seed signature [KLWW13, VKP+13]. Very

low average precision rate (1.14%) was obtained in a research examining module extraction using SNOMED CT, performed for usability purposes [LBIS12].

While the practicality of the SLBM method makes it ideal for use in ontology engineering scenarios [vDQCF18, QMMAN+18, ACSDD+19], evaluations have revealed that when used with large ontologies such as SNOMED CT, it generates extremely large modules [LBIS12, RU15, KK14].

In [vDQCF18], LBM was used to produce modules for the purpose of comparing their method's axioms to those in locality-based modules. Their approach generates axioms based on lexical regularities found in concept annotations. These lexical regularities are a set of patterns with a shared lexical structure. The size of the concepts in the created module "congenital" is 64.21% larger than the size of the concepts in the seed signature, while the size of the concepts in the other module "chronic" is 94.9% larger. This demonstrates the vast size of the modules, which contain a large number of symbols that are not within the range of the symbols of interest (the seed signature). The authors emphasised that smaller modules are preferable for their analysis since it necessitates manual inspection of the axioms specified by their method and those included in the module.

The work [QMMAN+18] makes use of OWL ontologies to augment Clinical Practice Guidelines (CPGs) by developing ontologies for the clinical domain of Computer-Interpretable Guidelines (CIG). These ontologies can be used in conjunction with electronic health records (EHR). To produce these ontologies, an alignment technique is used to automatically identify ontological terms relevant to the clinical domain of CIG, and then locality-based modules are computed for the identified terms. According to the authors, the generated modules may be unmanageable for the purpose of analysis due to their massive size.

The work [ACSDD+19] generated modules from SNOMED CT using the semantic locality-based $\bot$-type modularisation for 11 well-known medical conditions that impact both humans and animals. Employing the modules, they queried for phrases extracted from unstructured free-text using their deep learning approach. This is to get reliable knowledge about the diagnosis and treatment of well-known medical conditions affecting humans and animals. Their experiments to extract $\bot$-modules showed that the size of the concepts contained within the module has been significantly increased in contrast to the number of concepts contained in the seed signature. For example, the size of the concepts contained within the module extracted for the anaemia seed signature has been raised by 99.85% in comparison to the seed signature's size (34

concepts).

Due to the efficiency of SLBM, it is desirable as a way to optimise the implementations of a variety of methods, such as the LETHE tool for generating uniform interpolants [Koo20] and the OWL justification tool for generating subsets of the ontology representing explanation axioms [Hor11], as well as in [RGHJ13, Tsa12] for reasoning on ontologies.

Given that SLBM is primarily used by ontology engineers for its practicality, we use it as a primary method in our evaluation experiments in Chapters 4 and 5.

Semantic Modules

Two types of Σ-inseparability relations are defined as follows. In the following definitions, $\mathcal{L}$ ranges over $\mathcal{EL}$, $\mathcal{ELH}$, $\mathcal{ELH}^r$, and $\mathcal{ALC}$.

Definition 5 (Σ-inseparability [KLWW09, SSZ09])**.** *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two $\mathcal{L}$ ontologies and Σ be a signature*

1. *$\mathcal{O}_1$ and $\mathcal{O}_2$ satisfy the subsumption inseparability relation if for every $\mathcal{L}$-entailment α such that $sig(\alpha) \subseteq \Sigma$ we have $\mathcal{O}_1 \models \alpha$ if and only if $\mathcal{O}_2 \models \alpha$. In this case $\mathcal{O}_1$ and $\mathcal{O}_2$ are called* subsumption inseparable *and we write $\mathcal{O}_1 \equiv_\Sigma^{\sqsubseteq} \mathcal{O}_2$.*

2. *$\mathcal{O}_1$ and $\mathcal{O}_2$ satisfy the model inseparability relation* mod *if*

$$\{\mathcal{I}|_\Sigma |\ \mathcal{I} \models \mathcal{O}_1\} = \{\mathcal{J}|_\Sigma |\ \mathcal{J} \models \mathcal{O}_2\}$$

In this case $\mathcal{O}_1$ and $\mathcal{O}_2$ are called model inseparable *and we write $\mathcal{O}_1 \equiv_\Sigma^{mod} \mathcal{O}_2$.*

The formalisation of semantic modules uses the model-theoretic inseparability relation $\equiv_\Sigma^{\mathsf{mod}}$ [KLWW09, KLWW13]. Three different semantic modules exists and are defined in the following definition.

Definition 6 (Semantic modules [KLWW09, KLWW13])**.** *Let $\mathcal{O}$ be an $\mathcal{L}$-TBox and let Σ be a signature. Then $\mathcal{M}_S \subseteq \mathcal{O}$ is*

- *a* plain Σ-module *of $\mathcal{O}$ if $\mathcal{M}_S \equiv_\Sigma^{mod} \mathcal{O}$,*
- *a* self-contained Σ-module *of $\mathcal{O}$ if $\mathcal{M}_S \equiv_{\Sigma \cup sig(\mathcal{M}_S)}^{mod} \mathcal{O}$, and*
- *a* depleting Σ-module *of $\mathcal{O}$ if $\mathcal{O} \backslash \mathcal{M}_S \equiv_{\Sigma \cup sig(\mathcal{M}_S)}^{mod} \emptyset$.*

Algorithm 3 ComputeSemanticModule($\mathcal{O}$, Σ)

Input: An $\mathcal{ELI}$ terminology $\mathcal{O}$, Signature Σ
Output: Semantic module $\mathcal{M}_{\mathcal{S}}$

1: $\mathcal{M}_{\mathcal{S}} := \emptyset$
2: $\Sigma := \Sigma \cup \text{sig}(\mathcal{M}_S)$
3: $\mathcal{M}_{\mathcal{S}} := \mathsf{ApplyRules}(\mathcal{O}, \Sigma \cup \text{sig}(\mathcal{M}_{\mathcal{S}}))$

Removing a depleting module computed for a signature from an ontology, results in an ontology that has no information about the module's signature. If an $\mathcal{ELI}$ ontology (with inverse roles) does not contain trivial concept definitions (see Theorem 29 [KLWW13]), the notions of self-contained Σ-module and depleting Σ-module coincide. From $\mathcal{ELI}$-terminologies [KLWW13], the MEX tool[1] extracts minimal depleting and self-contained semantic modules.

The algorithm implemented within the MEX tool is presented in Algorithm 3 [KLWW13]. The main objective of this algorithm is identifying axioms that have direct or indirect *dependencies* among the symbols in the signature Σ [KLWW13]. The notion of Σ-dependency is defined as follows.

Definition 7 (Direct or indirect Σ-dependencies [KLWW13])**.** *Let $\mathcal{O}$ be an $\mathcal{L}$-TBox, Σ a signature, and $A \in \Sigma$. We say that A has a* direct or indirect Σ-dependency *in $\mathcal{O}$ if* $\mathsf{depend}(A) \cap \Sigma \neq \emptyset$*. We say that $\mathcal{O}$ contains a* direct or indirect Σ-dependency *when there is an $A \in \Sigma$ that has a direct or indirect Σ-dependency in $\mathcal{O}$.*

The set $\mathsf{depend}(A)$ is defined as $\mathsf{depend}(A) = \{X | A \prec^+ X\}$ where $\prec^+$ is the transitive closure relation of $\prec$. This set contains all symbols used in the definitions or inclusions of A in $\mathcal{O}$ [KLWW13].

Algorithm 3 utilises rules to identify axioms with Σ-dependencies. These rules are implemented recursively (Algorithm 4). The rules applied by the LocateAxiomDependencies method (Algorithm 5) identify axioms with Σ-dependencies, where Σ is the set $\Sigma \cup \mathsf{sig}(\mathcal{M}_S)$. Those identified axioms with such a dependency are added to the output, which is the semantic module $\mathcal{M}_S$.

With respect to the size of semantic modules, these are considered to be smaller in size than locality-based modules, which meets the requirement for conciseness in our problem, but the generated modules retain a range of symbols that are undesirable [CLW18].

[1] https://cgi.csc.liv.ac.uk/~konev/software/

Algorithm 4 ApplyRules($\mathcal{O}$, $\Sigma \cup \text{sig}(\mathcal{M}_S)$)

Input: An $\mathcal{ELI}$ terminology $\mathcal{O}$, Signature $\Sigma \cup \text{sig}(\mathcal{M}_S)$

1: LocateAxiomDependencies($\mathcal{O}$, $\Sigma \cup \text{sig}(\mathcal{M}_S)$)
2: **if** LocateIndirectAxiomDependency($\mathcal{O}$, $\Sigma \cup \text{sig}(\mathcal{M}_S)$) **then**
3: ApplyRules($\mathcal{O}$, $\Sigma \cup \text{sig}(\mathcal{M}_S)$)

Algorithm 5 LocateAxiomDependencies($\mathcal{O}$, $\Sigma \cup \text{sig}(\mathcal{M}_S)$)

Input: An $\mathcal{ELI}$ terminology $\mathcal{O}$, Signature $\Sigma \cup \text{sig}(\mathcal{M}_S)$
Output: Semantic module $\mathcal{M}_S$

1: **for** $\alpha \in \mathcal{O}$ **do**
2: $A :=$ GetSubConceptInChosenAxiom(α)
3: depend(A) $:=$ ComputeDepend(A)
4: **if** depend(A) $\cap \Sigma \neq \emptyset$ **then**
5: $\mathcal{M}_S := \mathcal{M}_S \cup \{\alpha\}$
6: $\Sigma := \Sigma \cup \text{sig}(\mathcal{M}_S)$

Minimal Subsumption Modules

The work on minimal subsumption modules introduced in [CLMW17,CLW18,KC17] preserve subsumption queries. In the following definition, $\mathcal{L}$ ranges over $\mathcal{EL}$, $\mathcal{ELH}$, $\mathcal{ELH}^r$, and $\mathcal{ALC}$.

Definition 8 (Subsumption modules [CLW18])**.** *Let $\mathcal{O}$ be an $\mathcal{L}$-TBox and let Σ be a signature. A subset $\mathcal{M}_{\sqsubseteq}$ of $\mathcal{O}$ is called an* $\mathcal{L}$-subsumption module of $\mathcal{O}$ w.r.t. Σ *iff for all $\mathcal{L}$-inclusions α with sig$(\alpha) \subseteq \Sigma$ it holds that $\mathcal{O} \models \alpha$ iff $\mathcal{M}_{\sqsubseteq} \models \alpha$. $\mathcal{M}_{\sqsubseteq}$ is called a* minimal subsumption module of $\mathcal{O}$ w.r.t. Σ *iff for any $\mathcal{M}'_{\sqsubseteq} \subsetneq \mathcal{M}_{\sqsubseteq}$, $\mathcal{M}'_{\sqsubseteq}$ is not a subsumption module of $\mathcal{O}$ w.r.t. Σ.*

An $\mathcal{L}$-inclusion is referred to as any axiom in the ontology $\mathcal{O}$ for some logic $\mathcal{L}$ where $\mathcal{O}$ is an $\mathcal{L}$-TBox.

Algorithms to compute approximated minimal subsumption modules are provided in [CLW18]. Algorithm 6 shows the basic principle underlying the computation of minimal subsumption modules, which depends on removing an axiom from the input ontology $\mathcal{O}$ only when the logical difference with respect to the input signature Σ between the ontology before and after removing the axiom is empty. If this is the case, then the axiom can be safely deleted from the ontology, and the remaining axioms that lead to such a logical difference will remain in the returned module $\mathcal{M}_{\sqsubseteq}$.

Algorithm 6 ComputeMinimalSubsumptionModule($\mathcal{O}$, Σ)

Input: An $\mathcal{ELH}^r$ terminology $\mathcal{O}$, Signature Σ
Output: Semantic module $\mathcal{M}_{\sqsubseteq}$

1: $\mathcal{M}_{\sqsubseteq} := \mathcal{O}$
2: **for** $\alpha \in \mathcal{O}$ **do**
3: **if** the logical difference restricted to Σ between $\mathcal{O}$ and $\mathcal{M}_{\sqsubseteq} \backslash \{\alpha\}$ is empty **then**
4: $\mathcal{M}_{\sqsubseteq} := \mathcal{M}_{\sqsubseteq} \backslash \{\alpha\}$
5: **return** $\mathcal{M}_{\sqsubseteq}$

Optimisations were taken into account when computing minimal subsumption modules according to the algorithms detailed in [CLW18]. The condition in Line 3 of Algorithm 6 can give exponentially many subsumption modules $\mathcal{M}_{\sqsubseteq}$ of the ontology $\mathcal{O}$. As stated by the authors in [CLW18], the interest is in computing subsumption modules that are smallest. The detailed algorithms in [CLW18] show procedures for computing all smallest minimal subsumption modules.

Deciding the preservation of subsumption queries in minimal subsumption modules can be expensive [CLW18]. The algorithm for computing minimal subsumption modules for $\mathcal{ELH}^r$-terminologies runs in exponential time, making it impractical for real-world applications.

Minimal subsumption modules are often substantially smaller than semantic modules, as demonstrated by empirical evaluations [CLW18]. The resulting minimal subsumption modules had axioms ranging of size 50–118 for a 100 input signature consisting of randomly selected 50 concept names and all role names from SNOMED CT [CLW18]. This is in contrast to the semantic modules, which had axioms of size 401–720 for the same input signature. This demonstrates that minimal subsumption modules of SNOMED CT are at least six times smaller than the semantic modules [CLW18].

Relationship Between the Three Module Notions

The following example demonstrates the differences between the three modularisation notions: nested locality-based modules, semantic modules and minimal subsumption modules. To aid with presentation, the concept names A, A_1, A_2, B, X, and Y are used to abbreviate SNOMED CT concept names as follows:

Example 2.

A *Mesoblastic_nephroma*

A_1 *Neoplasm_uncertain_whether_benign_or_malignant*

A_2 *Complex_mixed_AND/OR_stromal_neoplasm*

B *Neoplasm*

X *Neoplasm_and/or_hamartoma*

Y *Tumor*

Let $\Sigma = \{A, B\}$ *and*

$$\begin{aligned}\mathcal{O} = \{&A \sqsubseteq A_1 \sqcap A_2,\\ &A_1 \sqsubseteq B,\\ &A_2 \sqsubseteq B,\\ &B \sqsubseteq X\}.\end{aligned}$$

There are two minimal subsumption modules of $\mathcal{O}$ *w.r.t.* Σ*. According to Algorithm 6, such minimal subsumption modules are computed by first initialising the variable* $\mathcal{M}_{\sqsubseteq}$ *with the ontology* $\mathcal{O}$*. Then the procedure starts iterating through every axiom* α *in* $\mathcal{O}$*. In each iteration, the algorithm checks if there is a logical difference between* $\mathcal{O}$ *and* $\mathcal{M}_{\sqsubseteq}\backslash\{\alpha\}$ *that is entailed by* $\mathcal{O}$ *but not by* $\mathcal{M}_{\sqsubseteq}\backslash\{\alpha\}$*. If there is no logical difference, then the axiom* α *is removed from* $\mathcal{M}_{\sqsubseteq}$*, otherwise, the axiom is kept in the module* $\mathcal{M}_{\sqsubseteq}$*. The order of the axioms that were chosen during the iteration can lead to a different minimal subsumption module. Applying Algorithm 6 to compute minimal subsumption modules from* $\mathcal{O}$ *using* Σ *leads to the following module:*

$$\begin{aligned}\mathcal{M}_{\sqsubseteq_1} = \{&A \sqsubseteq A_1 \sqcap A_2,\\ &A_1 \sqsubseteq B\},\end{aligned}$$

and changing the order of axioms during the iteration (Algorithm 6) leads to another minimal subsumption module, which is the following module:

$$\begin{aligned}\mathcal{M}_{\sqsubseteq_2} = \{&A \sqsubseteq A_1 \sqcap A_2,\\ &A_2 \sqsubseteq B\}.\end{aligned}$$

Neither $\mathcal{M}_{\sqsubseteq_1}$ nor $\mathcal{M}_{\sqsubseteq_2}$ is sufficient to preserve the entailment $A \sqsubseteq B$ that only uses symbols in Σ. The semantic module $\mathcal{M}_S$ and nested locality-based module $\mathcal{M}_\star$ of $\mathcal{O}$ w.r.t. Σ are each

$$\begin{aligned}\mathcal{M}_S = \mathcal{M}_\star = \{&A \sqsubseteq A_1 \sqcap A_2,\\ &A_1 \sqsubseteq B\\ &A_2 \sqsubseteq B\}.\end{aligned}$$

Algorithm 3 computes semantic modules by identifying axioms that contain direct or indirect Σ-dependencies in the ontology $\mathcal{O}$. In accordance with Algorithm 5 (LocateAxiomDependencies), we assume that a selected axiom in the iteration through the axioms of the ontology $\mathcal{O}$ is $\alpha := B \sqsubseteq X$. Then, we construct the set depend(B) $:= \{X\}$ and test its intersection with $\Sigma \cup sig(\mathcal{M}_S) := \{A, B\}$ (see Definition 7). This test returns an empty set, indicating that the axiom $B \sqsubseteq X$ will not be added to the final module $\mathcal{M}_S$. Following the same procedure for the remaining axioms in $\mathcal{O}$ yields the aforementioned semantic module $\mathcal{M}_S$.

By using the method ComputeNestedLocalityBasedModule (Algorithm 2), the nested locality-based module is computed from the ontology $\mathcal{O}$ for Σ recursively. This recursive function stops when the current nested locality-based module has the same size as the module from the preceding call. Assuming the initial call to ComputeLocalityBasedModule computes a $\bot$-module for Σ from $\mathcal{O}$. This produces the bottom module $\mathcal{M}_\bot$, which is identical to $\mathcal{O}$. Then, the second call to ComputeLocalityBasedModule computes a $\top$-module from the $\bot$-module computed by the first call; this returns the $\top$-module = $\{A \sqsubseteq A_1 \sqcap A_2, A_1 \sqsubseteq B, A_2 \sqsubseteq B\}$. Then, a third call is made to ComputeLocalityBasedModule to compute a $\bot$-module from the preceding $\top$-module. Here, the resultant $\bot$-module has the same size as the $\top$-module, therefore the recursion ends (Line 3 of Algorithm 2). This results in the nested locality-based module $\mathcal{M}_\star$ mentioned above.

For

$$\mathcal{O}' = \{A \sqsubseteq A_1 \sqcap A_2, \tag{3.1}$$

$$A_1 \sqsubseteq B, \tag{3.2}$$

$$A_2 \sqsubseteq B, \tag{3.3}$$

$$B \equiv Y\}, \tag{3.4}$$

the nested locality-based module w.r.t. Σ is $\mathcal{O}'$ itself. However, the minimal subsumption modules and semantic module of $\mathcal{O}'$ w.r.t. Σ coincide with corresponding those of $\mathcal{O}$, respectively. Note that in SNOMED CT, the terms Neoplasm *(B) and* Tumor *(Y) are considered as synonyms and have the same identifier. To emphasise the distinction between semantic and nested modules, the axiom $B \equiv Y$ is included in this example.*

The semantic (minimal depleting) module of an $\mathcal{ELI}$-terminology for a signature Σ is always a subset of the respective nested locality-based module for the same signature Σ [KLWW13]. When the terminology lacks concept definitions of the form $A \equiv C$ (cf. Proposition 38 in [KLWW13]), a nested locality-based module corresponds to a semantic minimal depleting module. As demonstrated in Example 2, the nested locality-based module is equal to the semantic module when $\mathcal{O}$ does not contain the axiom $B \equiv Y$. In contrast to the semantic module notion, which is defined using a model-theoretic inseparability relation, the minimal subsumption module notion is defined in terms of entailment queries. This notion of minimal subsumption modularisation produces modules that are subsets of the semantic module computed for the same signature [Che18].

Both semantic and nested locality-based modules produce a unique subset of a given TBox in terms of a signature. On the other hand, as previously mentioned, there may be multiple, potentially exponentially many, minimal subsumption modules for a terminology (see Example 6 in [CLW18]).

As a property of semantic modules and nested locality-based modules, the following proposition is stated in [CAS+19b]:

Proposition 9 ([KLWW09, SSZ09])**.** *Let $\mathcal{L}$ be a description logic language ranges over $\mathcal{EL}$, $\mathcal{ELH}$, $\mathcal{ELH}^r$, and $\mathcal{ALC}$. Let $\mathcal{O}$ be an $\mathcal{L}$-TBox and let Σ be a signature. Additionally, let $\mathcal{M}_S$ be a semantic module and $\mathcal{M}_\star$ a nested locality-based module of $\mathcal{O}$ w.r.t. Σ. Then for all $\mathcal{L}$-inclusions α with $sig(\alpha) \subseteq \Sigma$, it holds that $\mathcal{O} \models \alpha$ iff $\mathcal{M}_S \models \alpha$ and $\mathcal{O} \models \alpha$ iff $\mathcal{M}_\star \models \alpha$.*

In the remainder of the thesis, we denote nested locality-based modules, semantic modules and minimal subsumption modules by $\mathcal{M}_\star$, $\mathcal{M}_S$ and $\mathcal{M}_\sqsubseteq$, respectively.

3.1.2 Uniform Interpolation (Forgetting)

Uniform interpolation (or forgetting) extracts inferred information from an ontology based on a predefined set of concept and role symbols (the input signature Σ). This results in a restricted view of the input ontology that uses only symbols from the specified

signature while preserving all of the ontology's logical entailments [LR94, LLM03, EIS+06, KWW09, KWW09, WWTP10, KWZ10, KS13a, NR14, ZS18b].

Uniform interpolation can be used for a variety of ontology applications, which include, *logical (semantic) difference:* facilitating the detection of possible semantic differences between different versions of ontologies [KWW08, KLWW12].
Ontology analysis: when a developer wants to analyse a large and complex ontology, it is preferable have smaller fragments to assist the analysis process [WWTP10].
Information hiding: when an ontology is to be published but a portion of it must remain hidden from the public, this portion can be erased by forgetting its associated symbols [GM14].
Ontology summary: providing a focused extract that aids in ontology comprehension [LW11, WWTP10]. Additionally, it can be utilised to reveal implicit, hidden relationships between symbols, which aids in the comprehension and maintenance of ontologies.

A definition of uniform interpolation that is applicable to any logic $\mathcal{L}$ -TBox is as follows [CAS+19b].

Definition 10 (Uniform Interpolation)**.** *Let $\mathcal{O}$ be an $\mathcal{L}$-TBox and let Σ be a signature. A finite set $\mathcal{UI}$ of $\mathcal{L}$-inclusions is an* $\mathcal{L}$-uniform interpolant (UI) of $\mathcal{O}$ for Σ *if the following conditions are satisfied: (i) sig$(\mathcal{UI}) \subseteq \Sigma$, and (ii) for every $\mathcal{L}$-inclusion α with sig$(\alpha) \subseteq \Sigma$, $\mathcal{O} \models \alpha$ iff $\mathcal{UI} \models \alpha$.*

Intuitively, uniform interpolation computes an ontology ($\mathcal{UI}$) which has the same logical entailments of the original ontology $\mathcal{O}$ up to the symbols in Σ.

The following example from [CAS+19b] demonstrates an application of forgetting from SNOMED CT. To make the presentation easier to follow, the concept and role names are abbreviated as follows.

Example 3.

A *Drug_interaction_with_drug (finding)*
A_1 *Drug_interaction (finding)*
A_2 *Drug_or_medicament (substance)*
B *Substance (substance)*
X *Adverse_drug_interaction_with_drug (disorder)*
Y *Adverse_drug_interaction (disorder)*
r *Associated_with (attribute)*

Let

$$\begin{aligned}\mathcal{O} = \{&A \sqsubseteq A_1,\\ &A_2 \sqsubseteq B,\\ &X \sqsubseteq Y \sqcap A \sqcap \exists r.A_2\}\end{aligned}$$

be an ontology and let $\Sigma_1 = \{A_1, X\}$ *and* $\Sigma_2 = \{A_1, B, X, r\}$ *be two sets of signature of interest. Then,* $\mathcal{UI} = \{X \sqsubseteq A_1\}$ *is a uniform interpolant of* $\mathcal{O}$ *for the smaller signature* Σ_1*, while the set containing additionally the axiom* $X \sqsubseteq \exists r.B$ *is a uniform interpolant of* $\mathcal{O}$ *for the larger signature* Σ_2*. The nested locality-based module of* $\mathcal{O}$ *for* Σ_1 *is*

$$\begin{aligned}\mathcal{M}_\star = \{&A \sqsubseteq A_1,\\ &X \sqsubseteq Y \sqcap A \sqcap \exists r.A_2\}\end{aligned}$$

which contains the concept names A, A_2, Y *and the role name* r *that are not in* Σ_1*.*

In some cases, generating uniform interpolants (UIs) for an input set of concept names results in ontologies that are too small to contain full definitions of the symbols in the input set. Hence, generating informative UIs mainly depend on the specified set of input symbols. For example, generating a $\mathcal{UI}$ from $\mathcal{O}$ in Example 3 for Σ_3 = $\{A, A_2\}$ results in an empty $\mathcal{UI}$.

While determining the existence of a uniform interpolant is 2ExpTime-complete for $\mathcal{ALC}$-TBoxes, a uniform interpolant is not always present in $\mathcal{EL}$- and $\mathcal{ALC}$-TBoxes

[LW11]. However, according to [KWZ10], for DL-Lite ontologies, a uniform interpolant always exists. Determining the existence of uniform interpolants for $\mathcal{EL}$-ontologies, is ExpTime-complete [LSW12].

There are tools for computing uniform interpolants for light-weight description logics $\mathcal{EL}$ [LW16, NR14, LSW12] and DL-Lite ontologies [WWTP10] despite the problem's high computational complexity. Additionally, several tools in the literature were developed to generate uniform interpolants from expressive description logics. In the following, we give brief overview about some of them.

LETHE The LETHE tool supports different forgetting algorithms to forget concept and role names of three types of ontologies, including $\mathcal{ALCH}$ [KS13a], $\mathcal{SHQ}$ [KS14] and $\mathcal{SH}$ TBoxes [KS15] and knowledge bases [Koo20]. The tool computes uniform interpolants using a resolution-based method. The tool is available as a standalone application and a Java-based library.

FAME Different versions of the tool exist. The UI-FAME tool computes uniform interpolants by forgetting concept and role names from $\mathcal{ALCH}$ ontologies. The method of the tool is based on hybrid approach that combines both resolution and Ackermann's Lemma [WDL+20]. The tool is available as a standalone application and a Java-based library. An older version of UI-FAME was introduced in [ZAS+19b].

The tool in [ZS18a] performs semantic forgetting for $\mathcal{ALCOIH}$ ontologies, which is a stronger notion of forgetting than uniform interpolation. The tool in [ZS19] also performs semantic forgetting, which works with $\mathcal{ALCOQH}$ ontologies.

$\mathcal{ELH}$ forgetting tool The tool introduced in [LLA+21a] supports forgetting from $\mathcal{ELH}$ ontologies. The underlying method is a hybrid approach of resolution and Ackermann's Lemma.

NUI The NUI tool forgets concept and role names from $\mathcal{EL}$ acyclic terminologies [KWW09]. The tool is based on a recursive algorithm for computing instance Σ-interpolants.

The UI-FAME, LETHE and NUI tools were used as part of our proposed workflow to evaluate the uniform interpolation method with real-world signatures in Chapter 4. The $\mathcal{ELH}$ forgetting tool is used as part of our method to generate focus set semantic differences in Chapter 6.

Algorithm 7 ComputeUniformInterpolants($\mathcal{O}$, Σ)

Input: Ontology $\mathcal{O}$, Signature Σ
Output: Uniform interpolant $\mathcal{UI}$

1: $\mathcal{N}$:= NormaliseOntology($\mathcal{O}$)
2: **for** $r \in \Sigma$ **do**
3: $\quad\mathcal{N}^r$:= r-localisation($\mathcal{N}$, r)
4: $\quad\mathcal{N}^r_d$:= IntroduceDefiners($\mathcal{N}^r$, r)
5: $\quad\mathcal{UI}'$:= $\mathcal{UI}' \cup$ ApplyCombinationRules($\mathcal{N}^r_d, r$)
6: **for** $A \in \Sigma$ **do**
7: $\quad\mathcal{N}^A$:= A-localisation($\mathcal{N}^r_d$, A)
8: $\quad\mathcal{N}^A_d$:= IntroduceDefiners($\mathcal{N}^A$, A)
9: $\quad\mathcal{UI}'$:= $\mathcal{UI}' \cup$ ApplyCombinationRules($\mathcal{N}^A_d$, A)
10: $\mathcal{UI}$:= PostProcess($\mathcal{UI}'$)

On medium-sized ontologies, the LETHE and UI-FAME tools performed well in terms of runtime and success rates [Koo20, WDL+20]. NUI uses a direct generation approach to compute UIs, unlike LETHE and UI-FAME tools which create UIs by forgetting concept and role names that are not present in the input signature. Due to the fact that the problem is generally unsolvable for $\mathcal{ALC}$, the UIs generated by LETHE and UI-FAME are expressed in an extended language employing fresh definer concept names to finitely represent infinite UIs. These definer concept names exist in the resulting UIs when there are cyclic dependencies over the forgetting symbols [KS13c, WDL+20].

Algorithm 7 shows the basic principle to compute uniform interpolants [LLA+21a]. The algorithm takes an input an ontology $\mathcal{O}$ and a signature set Σ. Line 1 normalises the ontology $\mathcal{O}$ into a clausal form. This clausal form is a specialised normal form which makes inference rules used to eliminate concept and role names applicable on the ontology. The output of this step is the normalised form $\mathcal{N}$ of the given ontology. The algorithm eliminates all role names in the input signature Σ (Lines 2–5) before eliminating concept names in Σ (Lines 6–9). The steps of removing role and concept names are similar, however, the combination rules used to eliminate role names differ from those used to eliminate concept names [Koo15, Zha18]. In both types of elimination, the algorithm localises the concept name A or role name r to be forgotten in the normalised ontology $\mathcal{N}$ (Lines 3 and 7 respectively). This step yields the portions of the normalised ontology containing the role name r or concept name A to be forgotten, denoted by $\mathcal{N}^r$ and $\mathcal{N}^A$, respectively. Lines 4 and 8 introduces definer

names on the identified portions ($\mathcal{N}^r$ and $\mathcal{N}^A$). These definer names are introduced to make the forgetting rules in the subsequent steps of the algorithm applicable on the ontology. Lines 5 and 9 apply combination rules on the portions $\mathcal{N}_d^r$ and $\mathcal{N}_d^A$ to forget the concept name A and the role name r, respectively. Additionally, this step involves removing the introduced definer names if possible. The result of forgetting a concept or a role name is added to the uniform interpolant $\mathcal{UI}'$. Line 10 of the algorithm post processes the uniform interpolant $\mathcal{UI}'$ to convert the ontology back to DL form.

Uniform interpolation can compute redundant axioms. We observed such redundancy when we tested the $\mathcal{ELH}$ forgetting tool. The forgetting process derives an axiom that is entailed by the rest of the axioms in the $\mathcal{UI}$. Consider the following example.

Example 4. *Let*

$$\begin{aligned}\mathcal{O} = \{&A \sqsubseteq B,\\ &C \sqsubseteq E,\\ &B \sqsubseteq C,\\ &F \sqsubseteq C,\\ &A \sqsubseteq F\},\end{aligned}$$

and $\Sigma = \{A,\ B,\ C,\ E\}$*. Then, the generated uniform interpolant for* Σ *is:*

$$\begin{aligned}\mathcal{UI} = \{&A \sqsubseteq B,\\ &B \sqsubseteq C,\\ &A \sqsubseteq C,\\ &C \sqsubseteq E\}.\end{aligned}$$

We can see that the $\mathcal{UI}$ *includes* $A \sqsubseteq C$ *as a result of forgetting* F *from* $\mathcal{O}$*. The axiom* $A \sqsubseteq C$ *is entailed by* $\{A \sqsubseteq F,\ F \sqsubseteq C\}$ *in* $\mathcal{O}$*. Also this axiom is entailed by the two axioms* $\{A \sqsubseteq B,\ B \sqsubseteq C\}$ *in the* $\mathcal{UI}$*. Thus, the axiom* $A \sqsubseteq C$ *is redundant and can be removed from the resulting* $\mathcal{UI}$*.*

We expect that is the case with other uniform interpolation tools, including the LETHE and UI-FAME tools, since checking for such redundant axioms is usually expensive because it requires additional reasoning. Thus, forgetting tools tend to leave them in the results.

It has been proved that in the worst case the size of the given uniform interpolants can be exponentially three times larger than the size of the input ontology for $\mathcal{EL}$ ontologies [NR14] and for $\mathcal{ALC}$ ontologies [LW11]. However, empirical evaluations appear to contradict such theoretical findings in [LLA+21b], with the exception of a small number of cases in which the generated uniform interpolants were greater than the input ontology in [KS13b, KS13a].

Uniform interpolation has been empirically evaluated using ontologies from the NCBO BioPortal repository to forget randomly selected symbols. The results for forgetting small number of symbols (5, 10, 50 and 150 concept names) using Koopmann and Schmidt's tool in [KS13b] showed that the computed uniform interpolants were larger than the input ontologies in 10% of the results, and larger than the nested locality-based modules in 24.1% of the results. In the worst-case scenario, the uniform interpolant was 559 times larger than its corresponding nested locality-based module.

Additionally, where a high number of symbols must be forgotten from the corpus, the tool in [KS13a] for forgetting symbols from $\mathcal{ALCH}$ ontologies revealed that the uniform interpolants were larger than the input ontologies in 6.05% of the results.

When the UI-FAME tool was used to forget randomly selected symbols from a sample of $\mathcal{ALCH}$ ontologies taken from the Oxford Ontology repository,[2] the resulting uniform interpolants reveal that as the size of the original ontologies grew, the size of the uniform interpolants shrank [WLZ+20].

While uniform interpolation produces highly precise extracts, which contain only those symbols specified in the input signature, empirical evaluations when forgetting randomly selected symbols in [KS13b] showed that the frequency of cases in which definer symbols are left in the results increased marginally as the number of forgotten concept symbols increased. Moreover, our evaluations in Chapter 4 with real-world signatures demonstrated a precision rate of 100% for all resulting UIs except for a few that contained definer names.

Because the axioms in the UIs are in general rewritten (not in their original form), they are generally unsuitable for certain ontology engineering practises in real-world scenarios, such as integration, import, or reuse. This is given by the fact that altering the structure of ontologies may result in incoherences or changes in their original structure, that could affect applications such as integration [SGSK18, SGW+18].

[2] `https://www.cs.ox.ac.uk/isg/ontologies/`

We discovered in Section 4.4 when we evaluated the computation of uniform interpolants for SNOMED CT NHS refsets, our results contained a considerable number of axioms whose structure was significantly different from the original ontology's structure. As illustrated in Figure 3.1, the uniform interpolant computed using the LETHE tool deviates from the typical structure of SNOMED CT. The structure of the SNOMED CT ontology (July 2017 version) is defined by axioms of the form $A \sqsubseteq C$ or $A \equiv D$, whereas the axioms in the figure are GCI axioms, including one stated in $\mathcal{ALC}$, i.e. having disjunction and universal role restrictions.

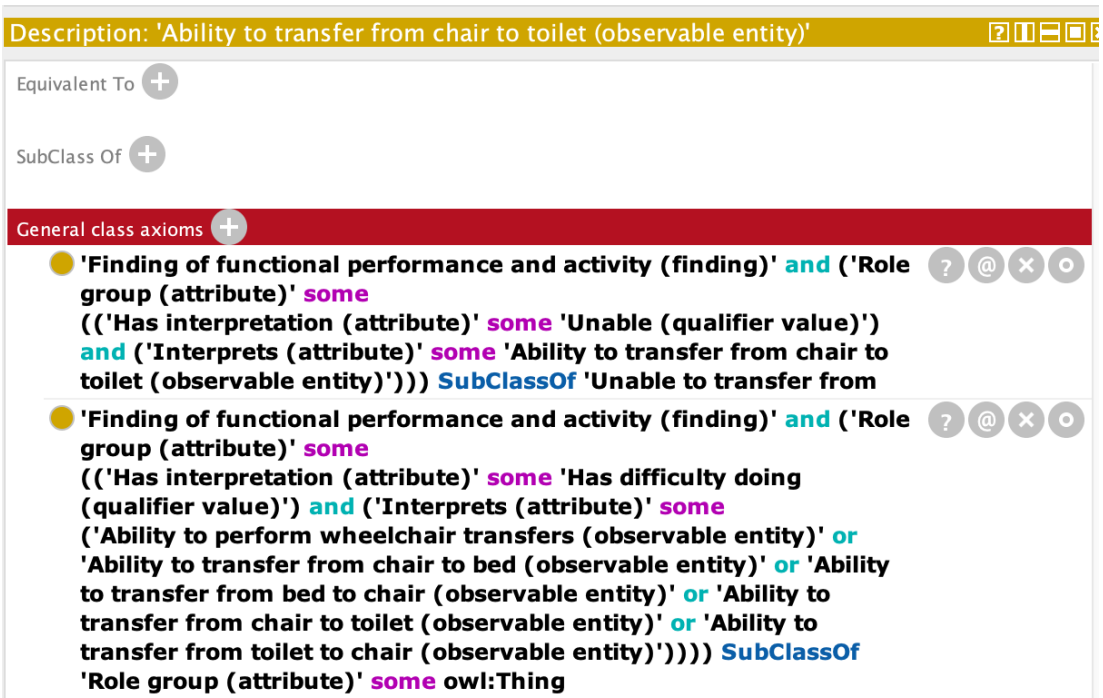

Figure 3.1: Part of a uniform interpolant resulted from using the LETHE tool on the nested-locality based module from SNOMED CT

We found that computing uniform interpolants directly on SNOMED CT for real-world signatures using the LETHE and UI-FAME tools was unfeasible in almost all of our tests due to the computationally expensive reasoning of generating uniform interpolants. As a result, Chapter 4 describes a workflow that makes use of pre-processing via modularisation.

The following proposition is the foundation for our method, which we describe in Chapter 4, for computing uniform interpolants from modules created using locality-based, semantic, or minimal subsumption modularisation methods. The proposition

asserts that computing uniform interpolants from the aforementioned module types is logically equivalent to computing them from the original ontology [CAS+19b].

Proposition 11 ([CAS+19b]). *Let $\mathcal{L}$ be a description logic language ranges over $\mathcal{EL}$, $\mathcal{ELH}$, $\mathcal{ELH}^r$, and $\mathcal{ALC}$. Let $\mathcal{O}$ be an $\mathcal{L}$-TBox and let Σ be a signature. If $\mathcal{UI}$ is a uniform interpolant of $\mathcal{M}$ for Σ where $\mathcal{M}$ is a module ranges over $\mathcal{M}_{\sqsubseteq}$, $\mathcal{M}_S$ or $\mathcal{M}_\star$. then $\mathcal{UI}$ is a uniform interpolant of $\mathcal{O}$ for Σ, where $\mathcal{M}$ is a module extracted from $\mathcal{O}$ for Σ.*

Assuming that $\mathcal{UI}^{\mathcal{O}}$ is a uniform interpolant of $\mathcal{O}$ for Σ, and $\mathcal{UI}^{\mathcal{M}}$ is a uniform interpolant of $\mathcal{M}$ for the same signature Σ where $\mathcal{M}$ ranges over $\mathcal{M}_{\sqsubseteq}$, $\mathcal{M}_S$ or $\mathcal{M}_\star$, $\mathcal{UI}^{\mathcal{O}}$ is equivalent to $\mathcal{UI}^{\mathcal{M}}$ when

- (i) For each axiom α such that $\text{sig}(\alpha) \subseteq \Sigma$, it holds that $\mathcal{O} \models \alpha$ iff $\mathcal{UI}^{\mathcal{O}} \models \alpha$.
- (ii) For each axiom α such that $\text{sig}(\alpha) \subseteq \Sigma$ it holds that $\mathcal{M} \models \alpha$ iff $\mathcal{O} \models \alpha$.

(i) follows from by Definition 10. For $\mathcal{M} = \mathcal{M}_{\sqsubseteq}$ (ii) follows by Definition 8, and for $\mathcal{M} = \mathcal{M}_S$ and $\mathcal{M} = \mathcal{M}_\star$ (ii) follows by Proposition 9. From (i) and (ii), we can conclude that $\mathcal{O} \models \alpha$ iff $\mathcal{UI}^{\mathcal{M}} \models \alpha$, where $\text{sig}(\mathcal{UI}^{\mathcal{M}}) \subseteq \Sigma$. This shows that $\mathcal{UI}^{\mathcal{O}}$ is equivalent to $\mathcal{UI}^{\mathcal{M}}$.

3.1.3 Signature Selection

The signature selection issue has been raised by many researchers in the field of ontology extraction including [Zha18, ZWZ+21, LS16, dSSS07]. The problem revolves around the user's difficulty to select an adequate set of symbols in order to generate appropriate ontology extracts. In this section, we discuss some scenarios of symbols selection using the aforementioned three modularisation notions and uniform interpolation.

Using the notion of uniform interpolation (forgetting) to extract restricted views of the original ontology for a given set of symbols Σ is governed by the intended use case. Generally, if the user is aware of the notion of *forgetting*, they will purposely select a collection of symbols in order to forget them because they are not desired in the resulting extract. On the other hand, the notion of forgetting can be used in the opposite way, where the user will deliberate over which symbols to include in the resulting extract. In each scenario, manually picking such symbols in order to construct an appropriate uniform interpolant is difficult, even more so when dealing with huge ontologies such as SNOMED CT.

When the SLBM is used to produce modules, the size of the resulting module is determined by the input symbol's hierarchy and the relationships between the symbols within the input ontology. In particular, the nested module ($\mathcal{M}_\star$) type, which is typically smaller than the method's $\top$ and $\bot$ types, is very dependent on the symbols chosen and their relationship in the ontology, with the resultant module being empty in some cases. This will be explained further in Chapter 4 (Section 4.1), along with giving a clarifying example.

As with SLBM, the size of semantic ($\mathcal{M}_S$) and minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) is directly related to the connectedness of the symbols in the input ontology. Typically, if the extracted nested locality-based module ($\mathcal{M}_\star$) for a particular seed signature is empty, the semantic and minimal subsumption modules for that seed signature are also empty. This is because minimal subsumption modules and semantic modules are supposed to be equal to or smaller than nested locality-based modules as clarified in [KLWW13] and as stated in [CLW18], which specifies that all minimal subsumption modules are contained in semantic modules.

Concept names in SNOMED CT reference sets can belong to a certain hierarchy in the ontology such as *renal diseases* [VRTG$^+$12]. In these instances, the remainder of the symbols required to compute meaningful uniform interpolates are absent from such reference sets. As a result, it is not straightforward to apply uniform interpolation to generate uniform interpolants from SNOMED CT, because the provided signature is not correctly structured for usage with uniform interpolation.

Utilising SNOMED CT reference sets with SLBM as seed symbols is dependent on the type of SLBM used, with the $\bot$-type resulting in the generation of definitions for concepts contained in the refsets, as well as many axioms describing the upward view of the concepts contained in the refsets. On the other hand, the $\top$-type results in definitions for the concepts contained in the refsets, as well as a large number of additional axioms that describe the downward view of reference set symbols. Nested type modules may fail to generate axioms for certain symbols in reference sets (as will be seen in Section 4.1).

With regards to the use of semantic modules and minimal subsumption modules with SNOMED CT reference sets, the resulting modules are typically smaller or equal to nested-locality based modules. Both minimal subsumption modules and semantic modules are driven by the aim of removing as many symbols as possible that are unrelated to the input signature, a criterion that can be overly rigid and result in incomplete definitions, as is the case with uniform interpolation.

For SNOMED CT users, a method suited for the type of seed symbols found in the SNOMED CT standard refsets is required for extracting subontologies that are meaningful in terms of containing complete definitions for the symbols included in the refsets. For this purpose, we provide two signature adjustment techniques in Chapter 4 that were developed with feedback from a SNOMED CT terminologist. In addition, Chapter 5 presents an extraction approach that generates extracts suitable for the set of input terms provided in standard SNOMED CT refsets.

Related Signature Adjustment Algorithms

There are a few different signature adjustment algorithms described in the literature. The recent work in [ZWZ+21] extends the signature by analysing lexical information related to the input signature Σ in the ontology's metadata. The goal is to determine the significance of Σ-terms and non-Σ-terms in the input ontology using embedding-based computation of relevant metadata. We argue that if the metadata is imprecise and does not adequately describe the formal semantics of relationships in the ontology, the expanded signature can be quite extensive and might include concepts that are irrelevant to the input signature set.

[LBIS12] introduces another approach that employs frequency term analysis to adjust the set of input symbols. The frequency term analysis filters a set of SNOMED CT concepts provided by the user according to their appearance in the MEDLINE Journal. This filtering is limited by a minimum input threshold number provided by the user. Due to its restriction to the MEDLINE journal and the SNOMED CT ontology, this method is not generally suitable for all use cases.

The Protege-TS plugin in [HS20] implements a general functionality that enables a signature to be extended in a variety of ways. Among these several functionalities are UpSet, DownSet, Equivalence, and other features such as General Class Axioms (GCAs) and role relationships between concepts. Additionally, the tool extends the signature iteratively for Concept-Role chains according to a certain depth number. This depth number controls how many concepts and roles are added to the extended signature set, that are in the subsumption chains between concepts connected horizontally via role restrictions.

We propose a more specific algorithm that does signature extension in a cautious way in Chapter 4. In the algorithm the user can specify the depth level of nesting of existential restrictions within the same axiom. We also consider the potential to extend the signature iteratively using an input iteration number, which acts similarly to the

depth number in [HS20].

In addition, a partitioning algorithm is proposed in Chapter 4, which can be used when the input signature is large. The purpose of the algorithm is to produce smaller sets of the input signatures. This partition begins by identifying the nested locality-based module associated for the input signature Σ in order to obtain the module's concept hierarchy $\mathcal{H}$, which is subsequently reduced to the concepts in Σ. This reduction would result, in some cases, in disjoint concept hierarchies $\mathcal{H}_1 \ldots \mathcal{H}_n$, where the signature of each concept hierarchy $\mathcal{H}_i$ belongs to Σ. The signature for each concept hierarchy can then be used to generate extracts or extended further using the algorithm outlined above.

3.1.4 Other Ontology Extraction Approaches

Application Scenarios-based Methods

In this section, we cover ontology extraction studies that were conducted for application scenarios in order to assess their relevance to the approaches described in this thesis.

The work by [NG12] presents a rewriting approach for computing extracts for an input set of symbols. Their requirement is that the generated extracts are small in size (less than the size of minimal modules) while retaining the syntactic structure of the original ontology, making it appropriate for *ontology reuse* applications. While the method attempts to preserve the extract's structure, it does not address the issue of generating complete definitions for the input symbols of interest. Instead, the method is motivated by limiting the resulting extract to the input symbols while preserving the module's structure, which is a compromise between the uniform interpolation result and the result of modularity.

The study in [LBIS12] includes the development of four graph-traversal-based modularisation strategies based on the method described in [SR06]. The study examined the size, precision, and coverage of signature concepts within the created modules generated by the four established heuristics and the locality-based modularisation method. Following the generation of modules using the various approaches, they used a filtering methodology based on the frequency of signature concepts within the MEDLINE database. Although such filtering assisted in increasing the coverage of the modules while decreasing their size, the techniques require pre-processing (matching, ranking, and indexing) and post-processing workloads (ontology reconstruction) [LBIS12].

The proposed method can be costly when used on a large ontology such as SNOMED CT. The study does not address the logical implications of the offered approaches (with the filtering process) on the preservation of the input symbols' entailments in the final extracts. The authors claimed, however, that their methods retain is-a relationships as well as cross-references from the concepts in the signature (attribute relationships). The author stated that additional research is necessary for this logical aspect before extracts derived from their approaches may be employed for reasoning purposes.

[dSM06] focused on a single application scenario: the automatic selection of knowledge components for web page annotation and semantic browsing. The proposed method, which is based on ontology graph traversal, generates suitable modules using a *knowledge selection* mechanism. This selection of knowledge includes two processes: ontology selection and modularisation. Modularisation is used to extract relevant components from ontologies that were initially chosen based on their initial selection. This study demonstrates the effective development of a method that is applicable to a specific application case but also adaptable to other contexts, as the authors claimed.

Definition Extraction Methods

In this section, we review two state of the art methods that extract equivalent concept definitions from $\mathcal{EL}$ ontologies. Such extraction can be done for reasons including, obtaining smaller size of concept definitions, or for obtaining definitions that are implicitly defined within the original ontology.

The work by Nikitina and Koopmann [NK17] tackles the issue of computing minimal equivalent concept definitions from the original concept definitions in $\mathcal{EL}$ ontologies. The method helps in eliminating redundancy from $\mathcal{EL}$ concept definitions by minimising their size. Their argument is that such redundancy complicates ontology maintenance and impairs comprehension. We argue that for application based scenarios some types of redundancy in concept definitions is desirable. For example, the long canonical form of SNOMED CT, which exhibit some redundant concepts in the concept definitions were shown to be very useful for several scenarios of using SNOMED CT [DSM02, HG12] (see Section 2.4.1 for an overview about normal forms in SNOMED CT).

$\mathcal{EL}$ ontologies have the Beth definability property [LPW10]. The Beth definability property states that a concept C is implicitly definable with respect to a signature Σ and an ontology $\mathcal{O}$ if there is some concept D such that the signature of D is a subset

of Σ, and $\mathcal{O} \models C \equiv D$, where $C \equiv D$ is not originally stated in $\mathcal{O}$. In other words, when a logic has such a property, it means that implicit definitions can be derived to be explicitly defined, i.e., stated in the ontology $\mathcal{O}$.

We argue that such a property is too strong for our problem that we are concerned with in this thesis. From the stated definitions of the form $A \equiv C$ (as well as concept inclusions $A \sqsubseteq C$) we are interested in an alternative and logically equivalent definition to the original definition in the original ontology (as will be seen in Section 5.1). This is because our problem is imposed by the requirement of the method, which is to generate an extract that is structurally equivalent to the original ontology in terms of concept definitions and not to derive possibly other implicit definitions that are of the form $A \equiv C$. This is to keep the generated extract minimal (i.e., restricted by definitions stated in the original ontology) and less dispersed.

3.2 Diffing Methods

A large number of tools and methods were developed to detect the differences between two artifacts, (e.g., OWL files). These tools range from approaches that work on the syntactic level, structural to the semantic level. *Syntactic difference* tools include Revision Control System (RCS) [Tic96], Concurrent Versions System (CVS)[3], and Source Code Control System (SCCS) [Roc75]. These tools take into account the order and spacing between the two artifacts' words/characters.

Structural difference tools regard the addition and removal of axioms as differences based on OWL's notion of structural equivalence [OWL]. In this notion, neither the order of operands in conjunctions and disjunctions, nor the order of axioms, is relevant. For example, the axiom $A \sqsubseteq B \sqcap \exists r.C$ is structurally equivalent to the axiom $A \sqsubseteq \exists r.C \sqcap B$. Examples of structural-based difference implementations include OWL Diff [KSK11], Bubastis [MHA+10], and PROMPTDIFF [RNM07].

Semantic difference tools, aim to capture the differences in meaning between ontologies in a way that is independent of how ontologies are articulated. The semantic difference, also referred to as *logical difference*, between two ontologies is defined by the axioms in one that are not entailed by the other, indicating the information gained and information lost between the two versions.

There are a few semantic difference tools available in the field; we will explore some of them in the following, namely the UI-Diff, CEX, and ECCO tools. In our

[3] `savannah.nongnu.org/projects/cvs`

review, we focus mainly on the UI-Diff method (in Section 3.2.1) because it is the foundation for our method in Chapter 6.

3.2.1 Semantic Difference

The notion of semantic difference is used to capture the difference in the meaning of concepts that is independent of how ontologies are represented. Tracking semantic difference is critical for several ontology engineering applications including when merging, integrating, or aligning different ontologies. This is accomplished by determining whether the version in question (**version a**) is safe for integration with another version (**version b**), in the sense that there are no differences in the meaning of terms shared by the two versions [LW14].

ECCO The ECCO tool [GPS12b] employs a hybrid technique that detects both structural and semantic differences . The tool categorises the changes based on whether they are logically effectual or not. Furthermore, the tool provides a thorough view of the axioms representing differences, which are aligned with the axioms that caused them. The semantic difference detection within the tool is primarily dependent on a standard reasoner which can detect differences stated in one version that are not entailed by the other. The employment of a reasoner to detect semantic differences does not aid in the discovery of implicit consequences that vary between versions.

CEX The CEX tool [KLWW08] computes semantic differences for $\mathcal{ELH}$ terminologies. Given two versions of ontologies and a set of signature Σ, the tool computes Σ-entailments using the notion of inseparability. When the two ontologies are Σ-inseparable then the logical difference is empty, otherwise there is a difference between both ontologies. The tool outputs a list of concept names A that are involved in a list of logical differences of the form $A \sqsubseteq C$ or $C \sqsubseteq A$, where these concept names are referred to as *affected terms*, but does not output the entire axiom involving the affected concept.

UI-based Semantic Differences Method

When altering an existing ontology, one of the most significant considerations for an ontology modeller is to ensure that the new modifications do not affect the meaning of *axioms beyond the fragment* under examination. Computing semantic difference using

uniform interpolation assists in *forgetting* symbols of axioms beyond the fragment under examination to ensure that both ontology versions are equivalent prior to and following changes. This assures that the changes are safe [LK14].

Additionally, semantic difference computed using uniform interpolation assists in revealing implicit differences within an ontology's deductive closure [KWW08]. [KWW08] emphasises that the interesting differences across ontologies are those expressed in their shared signature, not in the symbols used in only one of the two ontology versions. The use of uniform interpolation is appropriate in this case since it helps forgetting those symbols that are included in one of the two ontology versions.

As an application of the uniform interpolation method, Zhao et al. [ZAS+19b] presented a tool that computes semantic difference for $\mathcal{ALC}$ ontologies. The tool is based on the UI-FAME tool to track semantic difference between very large ontologies.

Another work presented by Liu el al. [LLA+21a], computes semantic difference based on the uniform interpolation method between $\mathcal{ELH}$ ontologies. They addressed the issue of the tool created in [ZAS+19b] producing more expressive axioms (in $\mathcal{ALC}$) when the input ontology is represented in a less expressive language such as $\mathcal{ELH}$ rather than $\mathcal{ALC}$ with which the tool operates. This discrepancy between the input ontology language and the method's target language can result in false semantic differences.

We now discuss how the uniform interpolation method generates semantic differences [LLA+21a, ZAS+19b].

Through uniform interpolation, the essential differences are generated, which are regarded as stronger entailments under the concerned signature. Thus, it assists in revealing a condensed set of semantic differences that would otherwise be infinite.

The following example, taken from [ZAS+19b], illustrates the generation of finite semantic differences based on the uniform interpolation method.

Example 5 ([ZAS+19b]). *Consider the ontologies:*

$$\begin{aligned}\mathcal{O}_1 = \{&A \sqsubseteq B,\\ &C \sqsubseteq E\},\end{aligned}$$

and

$$\begin{aligned}\mathcal{O}_2 = \{&A \sqsubseteq B,\\ &C \sqsubseteq E,\\ &B \sqsubseteq C,\\ &F \sqsubseteq C,\\ &A \sqsubseteq F\},\end{aligned}$$

with common symbols $\Sigma = \{A, B, C, E\}$*. The semantic difference is the set:*

$$\begin{aligned}\textsf{Diff}(\mathcal{O}_1, \mathcal{O}_2) = \{&B \sqsubseteq C,\\ &A \sqsubseteq C,\\ &B \sqsubseteq E,\\ &A \sqsubseteq E\}.\end{aligned}$$

To compute the UI difference, the symbols that are not in Σ *are forgotten using the uniform interpolation method. This gives the UI:*

$$\begin{aligned}\mathcal{UI}_2 = \{&A \sqsubseteq B,\\ &C \sqsubseteq E,\\ &B \sqsubseteq C,\\ &A \sqsubseteq C\}.\end{aligned}$$

Eliminating the axioms entailed by $\mathcal{O}_1$ *we get the UI-based semantic difference (*UI-Diff*):*

$$\textsf{UI-Diff}(\mathcal{O}_1, \mathcal{O}_2) = \{B \sqsubseteq C, A \sqsubseteq C\}$$

The UI-Diff$(\mathcal{O}_1, \mathcal{O}_2)$ witnesses are the strongest witnesses gained from $\mathcal{O}_1$ to $\mathcal{O}_2$. Using the UI-Diff method to generate witnesses guarantees a finite UI-based semantic difference is found.

The difference between $\mathcal{O}_1$ and $\mathcal{O}_2$ is empty iff $\mathcal{O}_1 \models \mathcal{UI}_2$, where $\mathcal{UI}_2$ is a Σ-uniform interpolant of $\mathcal{O}_2$ computed for $\Sigma \subseteq \text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_2)$. If $\mathcal{O}_1 \not\models \mathcal{UI}_2$ it means that every $\alpha \in \mathcal{UI}_2$ not entailed by $\mathcal{O}_1$ is a *witness*. All such witnesses will be denoted by $\mathcal{W}$. Throughout the thesis, we will use the terms *semantic differences* and *witnesses* interchangeably.

The set of witnesses produced based on the uniform interpolation method is defined as follows [LLA+21a].

Definition 12 (UI-based Semantic Difference / UI-witness)**.** *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be $\mathcal{ELH}$ ontologies. Let Σ be a subset of the shared signature of $\mathcal{O}_1$ and $\mathcal{O}_2$. The* UI-based semantic difference *between $\mathcal{O}_1$ and $\mathcal{O}_2$ is the set* UI-Diff$(\mathcal{O}_1, \mathcal{O}_2)$ *of all $\mathcal{ELH}$-axiom α such that (i) sig$(\alpha) \subseteq \Sigma$, (ii) $\alpha \in \mathcal{UI}_2$ and (iii) $\mathcal{O}_1 \not\models \alpha$, where $\mathcal{UI}_2$ is a Σ-uniform interpolant of $\mathcal{O}_2$. An axiom α satisfying these conditions is a* UI-witness *of a difference in $\mathcal{O}_2$ w.r.t. $\mathcal{O}_1$. All such axioms α are denoted by $\mathcal{W}$.*

Computing UI-Diff($\mathcal{O}_1$, $\mathcal{O}_2$) gives witnesses from $\mathcal{O}_1$ to $\mathcal{O}_2$ representing information gain. In other words, these witnesses are axioms that are entailed by $\mathcal{O}_2$ but not $\mathcal{O}_1$ and are denoted as $\mathcal{W}_2$. Switching the places of $\mathcal{O}_1$ and $\mathcal{O}_2$ in UI-Diff computes information loss $\mathcal{W}_1$, assuming that $\mathcal{O}_2$ is the newer version while $\mathcal{O}_1$ is the older version.

To compute UI-Diff($\mathcal{O}_1$, $\mathcal{O}_2$), two main steps should be performed:

1. Using the UI method, compute $\mathcal{UI}_2$ of $\mathcal{O}_2$ for $\Sigma = \text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_1)$.

2. Using an external reasoner, compute the set $\mathcal{W}_2$, which consists of the axioms $\alpha \in \mathcal{UI}_2$ but $\mathcal{O}_1 \not\models \alpha$.

3.2.2 Discussion

Existing methods for tracking semantic differences are beneficial and can be used to compare ontologies as large as SNOMED CT. A considerable issue with such methods is that the resulting discrepancies might be rather big and difficult to analyse. For example, the case study in [ZAS+19b] demonstrated that the July 2017 International edition of SNOMED CT did not entail over 8 400 axioms from the January 2017 version, whereas the Australian extension (December 2017 version) did not entail nearly 43 000 axioms from the July 2017 International edition. Additionally, when comparing different versions of the NCIt ontology using semantic diff tools such as ECCO and CEX, a considerable number of discrepancies were revealed. For example, a study for comparing the terms affected by the NCIt ontology update (from version 05.06f to version 06.08d) using a range of diffing notions, including those used by CEX and ECCO, a considerable number of affected terms (almost 57 000 terms) was discovered when utilising the ECCO tool [GPS12a].

The UI-Diff method can limit the generation of semantic differences to a smaller number, by specifying which symbols to include in the common signature set when

generating the witnesses. We found that selecting common symbols to generate the set of witnesses is not an easy task. If the user is interested in learning about changes to the definitions of certain concepts of interest and the relationships between their signatures, it is unclear which symbol to include or exclude to generate such differences.

As indicated previously, the CEX semantic difference tool generates a list of affected symbols based on their position relative to the witness axiom. The tool does not output the entire axiom involving the affected symbol. This can make the findings unclear, even more so if the modeller is interested in the reason for a change specified by the axioms.

While the ECCO tool [GPS11] produces comprehensible and well-aligned results, they lack inferred axioms between two versions of ontologies. As a result, ECCO is ineffective in detecting entailment sets that are entailed by one version but not by the other.

As previously mentioned in Section 2.4.3, SNOMED International provides a way to inspect probable modifications in each release of the ontology. Each release includes a set of *delta* files that list the differences of the current version since the prior one [Inth]. The generation of such diff reports is based on a notion of structural difference. Given that these reports list only structural changes, relying on them alone may not capture all of the possible consequences of updating the ontology to determine whether one version of the ontology is semantically different from another. Additionally, these reports are not human-readable and may contain a considerable number of dispersed entries, i.e., those belonging to many sub-hierarchies of the ontology [OCP16]. Moreover, they are unorganised according to a certain scheme, making them difficult to read and analyse.

It is vital for SNOMED CT users to keep track of differences across subdomains as auditing and quality assurance tasks are performed on SNOMED CT subhierarchies that describe specific subdomains, such as *cyst disorders* or *chronic diseases* [Intu]. Currently, none of the aforementioned methodologies are applied in the editorial work of SNOMED CT. One reason for this could be that they do not comply to SNOMED CT modelling.

To address the aforementioned gaps, we present in Chapter 6 a mechanism that combines the UI-Diff method with subontology extraction to generate focused semantic differences. Given that our subontology extraction adheres to SNOMED CT modelling (as discussed in Chapter 5), the proposed method makes use of subontology extraction

to track semantic differences between subdomains represented by the generated sub-ontologies. Additionally, the method presents the resulting differences according to their priority, based on whether the difference is focused or related to another symbol appearing in the concept definitions of focus.

Chapter 4

Modularity Meets Uniform Interpolation

Due to the difficulty of computing uniform interpolants, particularly for large ontologies, the size and complexity of the input ontology have a direct effect on the computation time. Rather than computing uniform interpolants across the entire ontology, this chapter proposes that speed can be gained by computing uniform interpolants across ontology modules.

We assess the practicality of the uniform interpolation method when being used in real-world conditions, i.e., when computing uniform interpolants for real-world signatures such as SNOMED CT reference sets. Furthermore, we assess the feasability of computing uniform interpolants from different module notions discussed in Chapter 3. Moreover, we explore the effect of computing uniform interpolants after adjusting the signature. To achieve that, we developed a workflow that computes uniform interpolants based on pre-processing methods including newly developed signature adjustment techniques (presented in Section 4.2.1), and ontology modularity tools (discussed in Section 3.1.1).

The *workflow consists of the following four stages:*

1. *signature adjustment,*

2. *ontology module extraction,*

3. *forgetting, and*

4. *feedback by domain experts,*

which is evaluated on the SNOMED CT and NCIt ontologies. Our investigation uses three different modularisation approaches (locality-based, semantic and minimal subsumption modularisation) and three forgetting tools (NUI, LETHE and FAME).

A critical component of the workflow is a *signature extension algorithm developed with input from domain experts*, as the shape and quality of ontology modules and UIs are highly dependent on the input signature. It is implausible to expect users to have the theoretical knowledge and modularisation/forgetting tools developers to have domain knowledge in order to select suitable signatures. Additionally, as we will see in Section 4.1, the selection of symbols to compute extracts is highly dependent on the method used and the relationships between the symbols in the input ontology.

We found that UIs computed for SNOMED CT refsets are usually insufficient because of the absence of additional symbols required to obtain meaningful UIs. Therefore, we present a signature extension algorithm for computing suitable input interpolation signatures that makes use of the ontology and existing refsets.

The second stage of the workflow generates modules, which are then used to generate UIs (in the third stage). These UIs are logically equivalent to those computed directly from the original ontology, without the use of modularisation (see Proposition 11 in Section 3.1.2). This shows the correctness of the workflow.

In Section 4.2, we present the workflow. As part of describing the workflow, we describe our signature adjustment algorithms, which include signature extension and partitioning (Section 4.2.1). In Section 4.3, we conduct an experiment to analyse cases of forgetting using the MRI referset, which is a standard refset used by SNOMED CT users, and demonstrate when forgetting becomes difficult to compute. Section 4.4 presents an experimental evaluation of the workflow with real-world ontologies, including SNOMED CT and NCIt. We discuss why ontology modules help to improve the forgetting process in Section 4.5. Finally, we conclude in Section 4.6.

4.1 Signature Selection

In this section, we discuss some scenarios of symbols selection using the locality-based modularisation, semantic modualrisation, minimal subsumption modularisation and uniform interpolation.

In the following, we identify that both: the chosen type of SLBM and the location of the selected signature within the input ontology to generate a module from SNOMED CT, can affect the size of the resulting module.

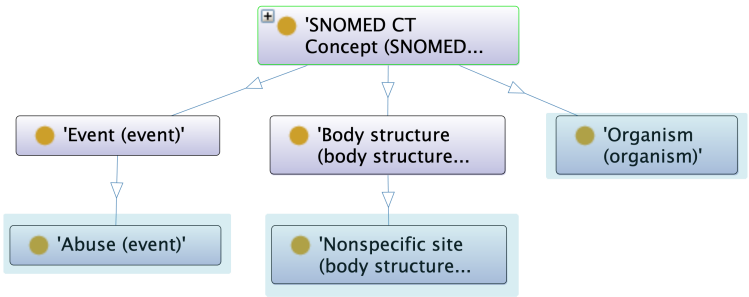

Figure 4.1: Location of symbols in Σ_1 highlighted in blue in SNOMED CT, as seen they are considered near to the root concept *SNOMED CT Concept*

Bottom $\bot$ Modules As previously mentioned, SLBM's $\bot$-type computes a module containing all super symbols in the input seed signature Σ. The resulting module is an upward view of the symbols in Σ from the input ontology $\mathcal{O}$. The size of the resulting module depends on the location of the seed signature in the original ontology that is chosen to compute the module. To be more precise, extracting a bottom module where the seed signature Σ exists in a level near to the root concepts would result in a module that is relatively smaller than a seed signature existing far from the root concepts. For example, extracting a $\bot$-module from SNOMED CT 2017 July version where the seed signature $\Sigma_1 = \{$*Nonspecific site, Abuse, Organism*$\}$ is close to the root concepts, results in a bottom module consisting of five axioms, as illustrated in Figure 4.1. On the other hand, if we choose concepts that are considered to be far from the root concepts, then the resulting module is relatively large. For instance, extracting a $\bot$-module for the set of two concept names in $\Sigma_2 = \{$*Ultrasound treatment to elbow*, *Oral steroid therapy*$\}$ from SNOMED CT 2017 July version, results in a module consisting of 57 logical axioms. Figure 4.2 shows the locations of the concepts in Σ_2 which are considered to be far from the root concept.

Top $\top$ Modules The $\top$-type generates a module that includes all sub symbols associated with the symbols in the input seed signature Σ from the ontology, resulting in the downward view of Σ. Similarly to the $\bot$-type, the location of the seed signature in the original ontology has an effect on the module's size. For example, computing the $\top$-module for Σ_1 results in a module with over 31 000 logical axioms, owing to the proximity of the concepts in Σ_1 to the root concept *SNOMED CT Concept*.

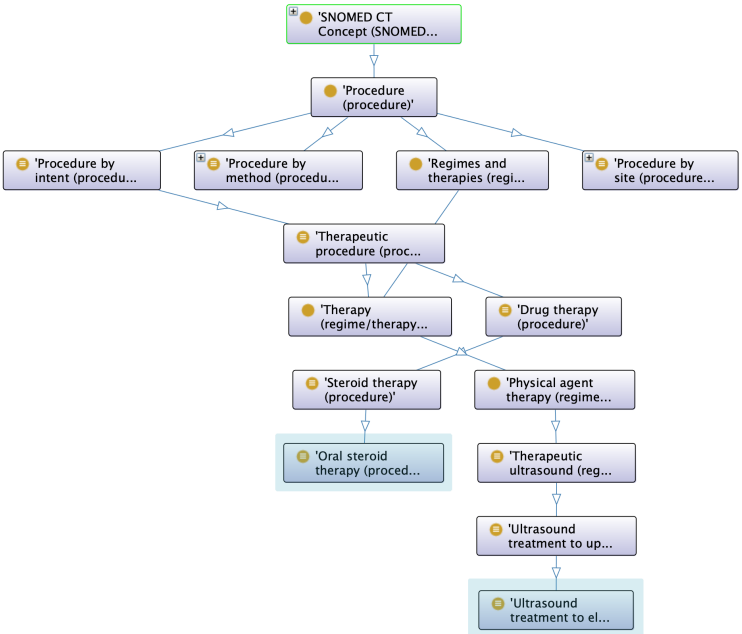

Figure 4.2: Location of symbols in Σ_2 highlighted in blue in SNOMED CT, as seen they are considered far from the root concept *SNOMED CT Concept*

Nested $\top\bot*$ Modules This type of SLBM is an iteration of the $\top$ and $\bot$ types. This iteration produces a module that is smaller than those produced for the $\bot$-($\top$)-types. Following the example of computing a module for Σ_2 from SNOMED CT 2017 July version, the generated $\top\bot*$-module for Σ_2 contains 46 logical axioms, which is smaller than its corresponding $\top$-module, consisting of 57 logical axioms.

In addition to the previous examples, we look at a particular case of signature selection when extracting a nested locality-based module. Assume a SNOMED CT user wishes to extract a nested module from SNOMED CT for a collection of concept

names pertaining to various components of the heart anatomy:

$$\begin{aligned}\Sigma_3 = \{&\textit{Ectopic heart structure 408674005}\\ &\textit{Heart tissue 386108004}\\ &\textit{Structure of transplanted heart 81510007}\\ &\textit{Cardiac internal structure 277712000}\\ &\textit{Cardiac perivalvular structure 118506009}\\ &\textit{Cardiac surface feature 277700004}\\ &\textit{Cardiac wall structure 272657006}\\ &\textit{Coronary artery structure 41801008}\\ &\textit{Structure of cardiac vein 85439003}\\ &\textit{Structure of tricuspid area 33321005}\}\end{aligned}$$

We suppose that the user selected the nested module due to its practicality and lesser size than the other two SLBM variants. This extraction produces an empty module. There are two reasons for this: (1) the method by which nested modularisation generates the module, namely iteration of top and bottom types; and (2) the positions of concept names in the signature within the input ontology (as being a sub or superconcept).

The type of the method affect the generated module because as mentioned previously, the method does iteration of top and bottom types. Looking at this iteration in this example, the method first generates a bottom module, which results in module containing all of the superconcepts of the symbols in Σ_3. Then, the method computes a top module from the generated bottom module, which results in an empty module since no subconcepts of the concepts in Σ_3 existed in the bottom module.

Regarding the second reason, the generation of a nested module is affected greatly by the positions of concepts names in the signature within the input ontology. If such concepts do not exist as both sub (e.g., A) and super concepts (e.g., B) in the input signature where there is a relation between them within the input ontology (such as $A \sqsubseteq B$), then the generated module is going to be empty. In this example, all of the concepts in the input signature exist as subconcepts in the input ontology. Thus, the nested type module is empty.

Additionally, if the concept in the input signature (e.g., $\Sigma = \{A\}$) is a defined concept, i.e., it exists as both a superconcept and a subconcept in $\mathcal{O} = \{A \equiv B \sqcap$

$C\}$, then the nested module is equal to $\mathcal{O}$ in this example, since A exists as both a superconcept and a subconcept in $\mathcal{O}$.

Minimal Subsumption Modules and Semantic Modules Following the previous example, we want to determine the resulting module for Σ_3 by generating a semantic module and a minimum subsumption module using the MEX and minimal subsumption module tools, respectively. Given that the minimal subsumption modules and semantic modules are supposed to be smaller than the nested locality-based module (see Section 3.1.1 for an example illustrating the relationship between the three module notions) then the generated modules for Σ_3 by MEX and the minimal subsumption module tools would also be empty.

Uniform Interpolation Following the example to generate an extract from SNOMED CT using Σ_3 as the input signature, the generated uniform interpolant would also be empty, since the symbols within the input ontology are not interconnected.

As a result of the preceding examples, we can conclude that the size of SLBM modules is very dependent on the symbol location within the input ontology, which can result in extremely big or very small modules. Moreover, there are instances where the generated nested locality-based module is empty, as this notion is influenced by the method's behavior and the specification of the input symbols, including their location and interconnection within the input ontology. This is also true for notions such as semantic and minimal subsumption modularisation, as well as uniform interpolation, because these methods are driven by the aim of generating extracts containing exactly the specified signature's symbols.

As we saw in Section 2.4.2, the type of seed symbols in the SNOMED CT standard refsets usually pertain to a specific subhierarchy of the ontology. The symbols required for extracting meaningful uniform interpolants in terms of including complete definitions for the symbols included in the refsets are typically absent from such refsets. As a result, we suggest two signature adjustment algorithms in Section 4.2.1 to address this problem.

4.2 Workflow

The main aim of the workflow is to investigate the computation of uniform interpolants for real-world signatures derived from very large ontologies. Figure 4.3 illustrates

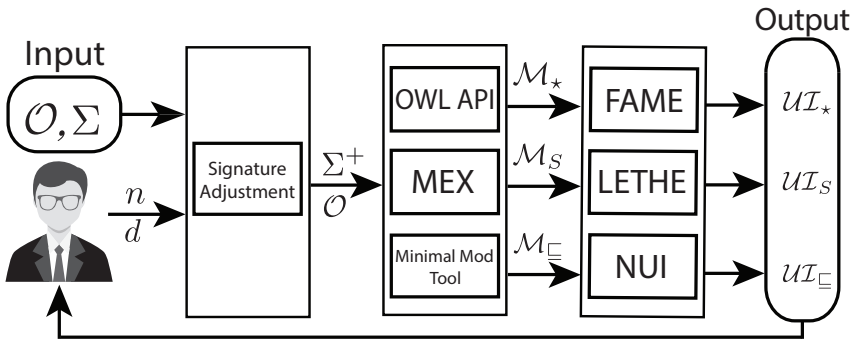

Figure 4.3: Workflow for computing UIs for the adjustment Σ^+ of the signature Σ

the stages of the workflow, which includes: (1) signature adjustment, (2) ontology modularity, (3) forgetting, (4) and evaluation and feedback from a domain expert.

The workflow is fed an ontology $\mathcal{O}$ and an interpolation signature Σ as inputs. The first stage of the workflow modifies the input signature by either extending it or partitioning it. As a result, the interpolation signature Σ^+ is generated. The second stage computes modules for the altered interpolation signature Σ^+ using the input ontology $\mathcal{O}$. This results in the generation of a module $\mathcal{M}$ that is used to compute a uniform interpolant $\mathcal{UI}$ for Σ^+.

To compute modules, we use three different modularisation tools, including the OWL API to compute nested-locality based modules [SSZ09], the MEX tool to compute semantic modules [KLWW08], and the tool that computes minimal subsumption modules [CLW18]. UIs are computed by three different forgetting tools including the NUI [KWW09], UI-FAME [WDL+20] and LETHE [Koo20] tools.

As mentioned in Section 3.1.2, the computation of UIs based on modules for an interpolation signature Σ is logically equivalent to computing UIs from the original ontology for Σ. This shows the correctness of the workflow.

In the following section we present our signature adjustment algorithms.

4.2.1 Signature Adjustments Algorithms

Random signatures or genuine seed signatures [VKP+13, KLWW13] were employed in previous evaluations of modularisation and forgetting methods on ontologies. Such signatures do not reflect how users and developers might use modules and uniform interpolants in real-world scenarios. Real-world signatures are typically:

1. A selection of a few specific concept names that the user is interested in. For example, a list of drug names related to different cancer diseases.

Algorithm 8 SignatureExtension($\mathcal{O}$, Σ, n, d)

Input: Normalised Ontology $\mathcal{O}$, signature Σ, no. of iterations n, role depth d
Output: Σ_d^+

```
1: Σ_d^+ := N_C ∩ sig(O) ∩ Σ
2: while n > 0 do
3:     for A ⊑ C ∈ O with A ∈ Σ_d^+ do
4:         if roleDepth(C) ≤ d then
5:             Σ_d^+ := Σ_d^+ ∪ sig(C)
6:     n := n − 1
```

2. A list of concept names already in use in the domain (a refset), or provided by a domain expert. For example, the set of renal diseases provided by the European Renal Association-European Dialysis and Transplant Association (ERA-EDTA) [NCS+18].

To account for the different types of signatures, we present two signature adjustment algorithms: signature extension and signature partition.

Signature Extension

In real-world scenarios, the question arises as to which axioms should be used and how (and by how much) a signature should be extended. Following discussions with developers from SNOMED International [CASG], we suggest Algorithm 8 for signature extension as a solution. The proposed solution is to use the ontology's axioms to improve the UI's signature in order to acquire more relevant ontology extracts. The algorithm extends the input signature with roles and their target concepts up to a certain level of nesting from the definitions of input concepts. We discovered that extending the signature in this way results in more informative UIs.

Algorithm 8 takes as input an ontology $\mathcal{O}$, a signature Σ, a number of iterations n and a role depth number d. The first step of the algorithm sets the output Σ^+ to the input signature Σ. Then, according to the input iteration number, we start extending the set Σ^+. This extension is done by calling the function $\mathsf{roleDepth}(C)$ below.

$$\mathsf{roleDepth}(C) := \begin{cases} 0 & C \in \mathsf{N_C}; \\ \mathsf{max}(\mathsf{roleDepth}(D), \mathsf{roleDepth}(E)) & C = D \sqcap E; \\ 1 + \mathsf{roleDepth}(D) & C = \exists r.D. \end{cases}$$

Algorithm 9 SignaturePartitioning($\mathcal{O}$, Σ)

Input: Ontology $\mathcal{O}$, Signature Σ
Output: extended signatures $\Sigma_1^+, ..., \Sigma_n^+$

1: $\mathcal{M}_\star :=$ ExtractNestedModule$(\Sigma, \mathcal{O})$
2: $\mathcal{H}' :=$ Reduce$(\Sigma,$ Classify$(\mathcal{M}_\star))$
3: $\langle \mathcal{H}'_1, ..., \mathcal{H}'_n \rangle :=$ Partition$(\mathcal{H}')$
4: **for** $i \in \{1, ..., n\}$ **do**
5: $\quad \Sigma_i :=$ sig$(\mathcal{H}'_i)$
6: $\quad \Sigma_i^+ :=$ SignatureExtension$(\mathcal{M}_\star, \Sigma_i)$

The roleDepth(C) function extends the signature by adding the superconcepts C of a concept name A based on the input role depth number d. The role depth number restricts the extension by the specified level of nesting of existential restrictions that occurs in C. After adding these superconcepts C to the set Σ_d^+, the method iteratively extends the concepts C by the specified number of iterations n.

It is possible to alter Algorithm 8 to work with ontologies other than SNOMED CT by adjusting the iteration number n and role depth d according to the recommendations of domain experts. In conversations with domain experts at IHTSDO and some experimentation, we agreed the values $n = 1$ and $d = 2$ are sufficient for the SNOMED CT setting [CASG].

Signature Partition

Manual examination becomes more difficult for large UIs computed for large signatures. For this purpose, we developed Algorithm 9 to partition the interpolation signature by reducing the number of concept and role names in the signature to a number of signature sets that are closely connected.

We will now go over the algorithm in detail.

Algorithm 9 takes as input an ontology $\mathcal{O}$ and a signature Σ. The first step of the algorithm calls the function ExtractNestedModule$(\Sigma, \mathcal{O})$, which is given by the the OWL API[1] to compute a nested locality-based module $\mathcal{M}_\star$. We only use the method as a quick way to compute the nested module $\mathcal{M}_\star$ of SNOMED CT ontology. When the method Classify$(\mathcal{M}_\star)$ in Line 2 is called, the ELK reasoner is invoked in order to classify $\mathcal{M}_\star$. Following that, we reduce the computed class hierarchy to the symbols included within the input signature. $\mathcal{H}'$ is therefore the classification of the concept

[1] `http://owlapi.sourceforge.net/`

names contained in the input signature. Line 3 contains the function Partition($\mathcal{H}'$), which divides the class hierarchy into unconnected parts. This is accomplished by computing the linked components of a directed graph representation of the class hierarchy. Then we partition $\mathcal{H}'$ by identifying different sets $\mathcal{H}'_i$ of concept inclusions such that every pair of concept names from the same set $\mathcal{H}'_i$ are connected via a chain of concept inclusions from $\mathcal{H}'_i$, and every pair of concept names taken from different sets $\mathcal{H}'_i$ and $\mathcal{H}'_j$ with $(i \neq j)$ are not connected in this sense. As a result, n disjoint sub-hierarchies: $\mathcal{H}_1, \ldots, \mathcal{H}_n$ are computed. The loop from Lines 4–6 first computes the signature Σ_i of the hierarchy $\mathcal{H}'_i$ and then extends Σ_i to Σ_i^+ using the function Signature-Extension($\mathcal{M}_\star, \Sigma_i$) provided in Algorithm 8.

4.2.2 Demonstrating the Benefits of Signature Adjustment

Given that the majority of SNOMED CT reference sets contain concepts belonging to a particular concept hierarchy, such as those under the *Clinical finding* hierarchy (See Section 2.4.2), such sets can be extended using our signature extension algorithm to include concepts describing the locations of clinical findings, which fall under the *Body structure* hierarchy. This is to ensure that legible concept definitions are generated when the uniform interpolation method is utilised, as the uniform interpolation method creates UIs that are restricted by the concepts specified in the input symbols (Σ).

The following example illustrates a situation where the obtained SNOMED CT refset is used to compute UIs with and without the signature extension algorithm. The purpose of the example is to demonstrate the use of the signature extension algorithm in generating a more useful UI; without the algorithm, the resultant UIs would not be meaningful.

Consider a SNOMED CT refset containing concept names about *Lordosis findings*. For illustration, we use four concept names out of the total concept names provided in the refset named Deformity of spine findings.[2] This is a standard SNOMED CT refset provided by the Chartered Society of Physiotherapy in the UK [oPb] (see Section 2.4.2 for an overview about SNOMED CT refsets).

Example 6. *Suppose the input ontology is SNOMED CT and the input signature to*

[2]https://www.csp.org.uk/system/files/csp_snomed_ct_subsets_20160414_v1.pdf

generate a UI is:

$$\Sigma = \{249710008\ \textit{Lordosis accentuated (finding)},$$
$$249711007\ \textit{Flattened lordosis (finding)},$$
$$249712000\ \textit{Lordosis absent (finding)},$$
$$249713005\ \textit{Lordosis reversed (finding)}\}$$

Then, a nested locality-based module is computed from SNOMED CT for Σ. The module is used to compute a uniform interpolant $\mathcal{UI}$ for Σ, which results in an empty ontology because of forgetting symbols that occur on the right hand-side of the definitions of concept names in Σ. Extending the signature gives the set Σ^+:

$$\Sigma^+ = \Sigma\ \cup \{281377003\ \textit{Lordosis finding (finding)},$$
$$53418001\ \textit{Structure of vertebral region of back (body structure)},$$
$$363698007\ \textit{Finding site (attribute)},$$
$$609096000\ \textit{Role group (attribute)}\},$$

The UI for the extended input set Σ^+ is:

$$\mathcal{UI} = \{\textit{Lordosis accentuated} \sqsubseteq \textit{Lordosis finding} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of vertebral region of back}),$$
$$\textit{Flattened lordosis} \sqsubseteq \textit{Lordosis finding} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of vertebral region of back}),$$
$$\textit{Lordosis absent} \sqsubseteq \textit{Lordosis finding} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of vertebral region of back}),$$
$$\textit{Lordosis reversed} \sqsubseteq \textit{Lordosis finding} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of vertebral region of back})\}$$

As seen from the $\mathcal{UI}$ computed for Σ^+, extending the signature helps generating a more informative UI that contain axioms describing the concepts in the input signature Σ. This is because the extended signature includes symbols that occur on the right hand-side of the definitions of concept names in the signature Σ.

After analysing the resulting extended signature and associated UI, Algorithm 8

can be used to make additional improvements. The process can be adjusted to iteratively increase the signature until a useful UI is obtained.

We now summarise the advantages of using Algorithm 9 (signature partition).

If the signature used to compute a UI is large, the resulting UI tend to also be large. This is specifically true when all of the symbols in the input signature occurs in the input ontology. Given this observation, in cases where a refset obtained from domain experts are quite large, the resulting UIs tend to also be relatively large. Large UIs can complicates manual inspection. In such cases, Algorithm 9 helps obtaining relatively small sets of closely related symbols, because it partitions the input signature into smaller sets of concept and role names.

Furthermore, using the workflow, when the input interpolation signature is huge, the generated module tend to also be large, specially in the case of computing nested locality-based modules. This is because of how locality-based modularisation works which computes modules containing lots of symbols outside the range of the input signature. As a result, a significant number of symbols that are not included in the input signature must be forgotten. Therefore, the uniform interpolation tool might not success in giving a UI within a reasonable time-frame. In these situations, the signature partitioning algorithm can be employed to divide the input signature into many smaller sets. The workflow can then be used to compute smaller UIs for the partitioned signature sets.

It is worth noting that producing smaller sets of the interpolation signature is conditional on the relationships between the symbols in the signature inside the input ontology, as the reduction step in Algorithm 9 may not provide a number of disjoint subhierarchies. Even if we end up with disconnected subhierarchies, the amount of symbols contained within a subhierarchy may not be much less than the number of symbols contained within the initial input signature.

4.3 Cases of Difficult Forgetting

This section presents cases in which forgetting becomes computationally difficult.

In this experiment, we tested computing a UI for the Magnetic Resonance Imaging (MRI) refset, provided by the domain expert [Intm]. We begin by utilising the workflow to (a) extend the MRI signature and obtain Σ^+, (b) compute a nested-locality based module for the extended MRI signature Σ^+ and (c) compute a UI for Σ^+ based on the nested module. We found that the workflow did not succeed in computing a

UI in a reasonable amount of time. We conducted an analysis of the locations of the forgetting symbols in the axioms contained within the nested locality-based module. We considered the nested module, to do this analysis as it has the largest size among the three module types. The increased number of axioms can help confirm that specific locations of such forgetting symbols are causing the computational difficulty.

A forgetting signature $\mathcal{F}$ is defined as $\mathcal{F} := \text{sig}(\mathcal{O})\backslash\Sigma^+$, where Σ^+ is the interpolation signature.

We concentrated on GCI axioms of the form $C \sqsubseteq A$ since our testing with LETHE and UI-FAME indicated that forgetting tools demand a considerable amount of computation time when forgetting symbols occur on the left hand side of GCI axioms. We wanted to confirm our observations by analysing the set of forgetting symbols that occur in these GCI axioms within the nested module. To do this, we segmented the forgetting symbols to the following:

- X^{GCI}, which are symbols located in the left hand side of GCI axioms: $C \sqsubseteq A$.
  - $\mathrm{X}^{GCI-atomic}$: This set contains the atomic concepts located on the left hand side of GCI axioms. e.g., B in $B \sqcap C \sqsubseteq A$.
  - $\mathrm{X}^{GCI-complex}$: This set contains the symbols that are filler names for existential restrictions in the left hand side of GCI axioms. e.g., B_2 in $B \sqcap \exists r.B_2 \sqsubseteq A$.
- X^{noGCI}, which are symbols that are not located in the left hand side of GCI axioms: $C \sqsubseteq A$.

Given that the module computed for the extended MRI signature set contains only nested existential restrictions of the form $\exists r.(\exists s.B)$, where B is a concept name, the set $\mathrm{X}^{GCI-complex}$ contains filler names of nested existential restrictions only.

We wanted to determine whether adding the set $\mathrm{X}^{GCI-complex}$ to the interpolation signature Σ^+ would cause the forgetting tool to terminate in a reasonable amount of time. We tried to compute a UI from the nested module that we used to analyse the locations of the forgetting symbols. The interpolation signature to compute the UI is the union of the sets Σ^+ and $\mathrm{X}^{GCI-complex}$. Table 4.1 summarises the computation time when running LETHE and UI-FAME. Using the LETHE tool, we were able to compute a UI with an interpolation signature of $\Sigma^+ \cup \mathrm{X}^{GCI-complex}$. On the other hand, we were unable to generate a UI using UI-FAME.

Table 4.1: Computation times of the forgetting tools LETHE and UI-FAME when computing a UI for $\Sigma^{+} \cup \mathrm{X}^{GCI-complex}$

X set	X set size	Computation Time	Terminate	Tool
$\mathrm{X}^{GCI-complex}$	201	16 m	Yes	LETHE
$\mathrm{X}^{GCI-complex}$	201	N/A	No	UI-FAME

We conclude from the experiment that if the forgetting signature occurs as filler names for nested existential restrictions in the left hand side of GCI axioms of the form $C \sqsubseteq A$, and the frequency of such symbols is more than 28% of the forgetting symbols, then forgetting spends long computation time, and could lead to out of memory exceptions.

Our observation confirms the finding in [KS13b], in which the LETHE tool was used to forget randomly selected symbols from a sample of NCBO ontologies. According to their observation, the largest impact on computation time came when the forgotten concept symbols were encountered in high numbers within role restrictions. In such instances, an inference rule known as the role propagation rule is applied. In these conditions, this rule has to be used more frequently. When the role propagation rule is used, numerous derivations are generated, resulting in timeouts.

4.4 Experimental Evaluation

We assess the workflow against real-world ontologies, including SNOMED CT and NCIt, to compute uniform interpolants for real-world signatures. In Section 4.4.1, we begin by describing the workflow settings. Second, we discuss the data that were utilised in the experiments (Section 4.4.2). Thirdly, in Section 4.4.3, we detail the different experiments that we conducted. The findings of the experiments are then discussed in Sections 4.4.4–4.4.8.

4.4.1 Workflow Settings

We used the OWL API, the MEX tool and the tool in [CLW18] to compute nested modules, semantic modules and minimal subsumption modules, respectively. To perform forgetting we used the three tools: NUI [KWW09], UI-FAME [WDL+20] and LETHE [Koo20]. The experiments with UI-FAME and LETHE were conducted on machines equipped with Intel Xeon CPU E5-2640 v3 running at 2.60GHz with 32GB of RAM. As NUI ran only on 32 bit architectures, its experiments were run on machines

equipped with Intel Xeon 4 Duo CPU at 2.50 GHz and 64GiB of RAM. The workflow can be accessed at `https://tinyurl.com/3wynswdx`. [3]

4.4.2 Data Used

SNOMED CT

The evaluation was performed on the $\mathcal{ELH}$ fragment of SNOMED CT (version Jan 2019), which has 349 763 axioms, 349 430 concept names, and 119 roles. We obtained the $\mathcal{EL}$ fragment of the ontology by removing 129 axioms not in $\mathcal{EL}$ and 20 GCI axioms of the form $C \sqsubseteq A$. We used two sets of real-world signatures, *NHS refsets*[4] and *ERA refset*, as inputs (see Section 2.4.2 for an overview about SNOMED CT refsets).

NHS refsets. We used a sample of 165 signatures from the NHS refsets provided in [nhs], all of which were contained in the SNOMED CT ontology. The signature was extended using Algorithm 8. Following discussions with domain experts, the input iteration number n was determined to be 1 and the role depth number d to be 2 (Σ_2^+, $n = 1$, $d = 2$).

The extended signatures consisted of 2–116 865 concept names and 0–33 role names. We separated the signatures into three different groups according to the number of concept names they contained. The three groups of concepts ranged as (0–150), (150–300) and (300–116 865). This gave clearer insights into the performance of forgetting a range of different sizes of concept names. For each signature, we computed the three types of modules, their precision rate and the UIs using each forgetting tool.

ERA refset. We used the European Renal Association's (ERA) reference set of symbols from SNOMED CT, which was provided by IHTSDO. The ERA reference set includes a list of the main renal diseases for use in renal centres and registries.

We began by splitting the ERA refset using the partitioning algorithm (Algorithm 9) to create 14 small disjoint signatures. Following that, we used Algorithm 8 to extend these signatures to include symbols representing body structure and role names of renal diseases in the ERA refset. The resulting signatures were composed of 5–40 concept names and 0–8 role names.

[3] The results of the NUI tool were generated by Jieying Chen. The UI-FAME tool used in these experiments is the one described in [ZAS+19b], which is an older version of the more recent UI-FAME tool described in [WDL+20].

[4] `https://isd.digital.nhs.uk/trud3/user/guest/group/0/pack/40`

NCIt

We also used the acyclic $\mathcal{EL}$ fragment of the NCIt ontology (201906 version), which has $148\,216$ axioms, $150\,020$ concept names and 94 roles in our evaluation. The real-world signatures were derived from the sets of medications approved by the Food and Drug Administration (FDA) for 41 different types of cancer, which included 2–162 concept names and 0–17 role names.[5] Because the NCIt ontology has axioms where the nesting level of existential restrictions is at most one, the experiments were conducted with extended signatures for which the role depth number d is 1. As with SNOMED CT experiments, a single iteration was chosen (Σ_1^+, $n = 1$, $d = 1$).

4.4.3 Conducted Experiments

We carried out five different experiments to test empirically the following hypotheses:

1. The type of module used to generate a UI has an impact on the computation time and success rates, whether it's a nested locality-based module, a semantic module, or a minimal subsumption module.

2. Module sizes and precision rates differ from large (for nested locality-based modules) to small (for minimal subsumption modules) for the same input signature, depending on the module notion utilised.

3. When computing UIs using the workflow, we expect that the size of the UIs will be similar because the same uniform interpolation signature is used to compute the UI regardless as to which modularisation method is used.

The conducted experiments are the following:

1. **Success rates of forgetting**
   The success rate of a forgetting tool is defined as its capability to return a UI within the specified timeout. This timeout period was one hour for all three forgetting tools. Our experiments showed that success rates barely increased when the timeout was increased to more than one hour. We therefore limited the timeout to one hour.

[5] List of drugs approved for different types of cancer: `https://www.cancer.gov/about-cancer/treatment/drugs/cancer-type`

2. **Computation of precision rates**
   We computed precision rates for the different modules types as well as the resulting UIs. As a definition of the precision rate of a module $\mathcal{M}$ (or a UI) computed for a given signature Σ we used:

$$\mathsf{Precision}(\mathcal{M}, \Sigma) := |((\Sigma \cap \mathrm{sig}(\mathcal{M})) \cup \{\top\})|/|\mathrm{sig}(\mathcal{M})|, \tag{4.1}$$

   which is a slight adaptation from [LBIS12] with $\top$ added in the numerator since every individual is an instance of the $\top$ concept.

3. **Interpolation signature partitioning**

   We used Algorithm 9 to partition the input signature, where the input signature is the ERA refset. This is done in order to create a smaller input signature sets that can be used as input to the forgetting tools in order to successfully compute UIs. Additionally, we report on the success rates and computation times for computing UIs utilising the workflow in this experiment.

4. **Size of computed UIs**

   We compared the size of the generated UIs among the three forgetting tools. The size is defined as the number of logical axioms in the UI. The size comparison was done for the NCIt ontology UIs.

5. **Axioms not in the input ontology's language**

   Because both the LETHE and UI-FAME tools perform forgetting where the target language is $\mathcal{ALC}$, the results might include a number of axioms that are outside the input ontology's language. For this reason, we counted the number of axioms outside $\mathcal{EL}$, the sample language.

Note that in interpreting the results of the tables one should take into account that NUI computes UIs which are formulated in $\mathcal{EL}$, whereas the target logic of LETHE and UI-FAME is $\mathcal{ALC}$. The results are therefore not directly comparable.

The following sections summarise the results from the experiments.

4.4.4 Success Rates of Forgetting

The median forgetting times reported in this section do not include the time required to extract modules of all three types. If forgetting was not finished within the timeout

Table 4.2: Success rate and median computation time (s) of computing UIs using NHS refsets as signatures by UI-FAME on SNOMED CT

Tool		UI-FAME					
$\lvert\Sigma^* \cap \mathsf{N_C}\rvert$	No. of Sig	Success Rate (%)			Med. Time of Forgetting		
		$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
$[0, 150]$	110	75.45	89.09	94.54	0.28	0.28	0.27
$[150, 300]$	13	38.46	46.15	76.92	3 600.00	3 600.00	13.51
$[300, 116\,865]$	42	4.76	11.90	14.28	3 600.00	3 600.00	3 600.00

Table 4.3: Success rate and median computation time(s) of computing UIs using NHS refsets as signatures by LETHE on SNOMED CT

Tool		LETHE					
$\lvert\Sigma^* \cap \mathsf{N_C}\rvert$	No. of Sig	Success Rate (%)			Med. Time of Forgetting		
		$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
$[0, 150]$	110	92.73	100.00	100.00	1.29	1.11	0.96
$[150, 300]$	13	61.54	84.62	92.31	9.72	4.10	3.22
$[300, 116\,865]$	42	14.29	26.20	28.61	3 600.00	3 600.00	3 600.00

period, the running time of the forgetting tool was taken to be one hour. Where the success rate fell below 50%, the median computation time was therefore taken to be one hour.

SNOMED CT - NHS refsets Tables 4.2, 4.3, and 4.4 show the success rates and median computation times of computing UIs using the NHS refsets as signatures by

Table 4.4: Success rate and median computation time(s) of computing UIs using NHS refsets as signatures by NUI on SNOMED CT

Tool		NUI					
$\lvert\Sigma^* \cap \mathsf{N_C}\rvert$	No. of Sig	Success Rate (%)			Med. Time of Forgetting		
		$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
$[0, 150]$	110	97.27	99.09	100.00	0.01	0.01	0.01
$[150, 300]$	13	100.00	100.00	100.00	0.09	0.04	0.02
$[300, 116\,865]$	42	61.90	78.57	84.62	59.05	20.84	16.10

Table 4.5: Computation time (s) of UIs on different modules of NCIt by UI-FAME, LETHE and NUI

Tool	UI-FAME			LETHE			NUI		
$\mathcal{T}$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
Min.	0.12	0.13	0.13	0.62	0.61	0.56	0.03	0.01	0.02
Max.	3 600.00	3 600.00	0.98	31.68	24.39	1.97	0.14	0.08	0.04
Avg.	527.07	88.07	0.30	4.24	1.68	0.86	0.05	0.03	0.03
Med.	0.33	0.21	0.25	1.59	0.74	0.85	0.05	0.03	0.03
Succ.(%)	85.37	97.56	100.00	100.00	100.00	100.00	100.00	100.00	100.00

UI-FAME, LETHE and NUI, respectively. As shown from the results, while computing UIs on small signatures comprising less than 150 concept names, executing the forgetting tools on precomputed minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) allowed for the computation of UIs in less than an hour with a success rate[6] of 100% or near 100% for LETHE and NUI (Tables 4.3, and 4.4). Precomputing semantic modules ($\mathcal{M}_S$) is also competitive for such small signatures, as it increased success rates to 100% for LETHE (Table 4.3) and close to 100% for NUI (Table 4.4).

Due to the fast growing ontology modules for signatures having more than 300 concept names, success rates decreased dramatically with increasing signature size. This impact is noticeable with UI-FAME and LETHE, and to a lesser extend for NUI. When computing UIs based on minimal subsumption modules, UI-FAME's success rate slightly increased to 14.3% (Table 4.2). In comparison to UI-FAME, LETHE increased success rates by 14.33% (Table 4.3). On the other hand, NUI increased success rates substantially (84.62%) (Table 4.4).

NCIt Table 4.5 shows the computation times of forgetting for the UIs of the NCIt ontology. Due to the fact that the NCIt ontology was 50% smaller and less complicated than SNOMED CT (which incorporates nested existential restrictions), success rates were generally greater than in the experiments of SNOMED CT. We found that for all three types of modules, forgetting using LETHE was 100%. On the other hand, UI-FAME performed less well.

In general, computing UIs from minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) was faster, particularly for LETHE and UI-FAME. In the worst-case situation, the time required to compute a UI using LETHE from the nested locality-based module ($\mathcal{M}_\star$) was 7.30 seconds longer than the time required to compute the UI from the semantic module ($\mathcal{M}_S$)

[6]If a tool terminated within 1 hour, we counted it as a successful run.

and 29.71 seconds longer than the time required to compute the UI from the minimal subusumption module. This does not appear to incur a significant computational burden, and as can be seen, the success rates were 100%. On the other hand, computing UIs using UI-FAME based on nested locality-based modules was significantly longer and had lower success rates (85.37%).

4.4.5 Computation of Precision Rates

SNOMED CT - NHS refsets To compute precision rates for the three different module types, we used the formula 4.1. The results indicate that the average precision rates of nested modules ($\mathcal{M}_\star$), semantic modules ($\mathcal{M}_S$), and minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) were approximately 72%, 71%, and 78%, respectively.

In comparison, UIs have a precision rate of 100% (Definition 10), unless definers appear in the results. Seven UIs determined from nested modules contained definer concept names in the case of LETHE. This was also true for one of the semantic modules and minimal subsumption modules. There were no such cases in the ERA refset and NCIt evaluations, i.e., there were zero UIs with definer names.

As mentioned in Section 3.1.2, the appearance of definer concept names in the results returned by LETHE is due to the method it uses. The methods of both LETHE and UI-FAME convert the axioms of the input ontology into normal forms before starting the inference process that is used to compute UIs. Following the completion of the forgetting process, both tools use additional computations in order to eliminate any possible definer concept names from the results that may have been introduced as part of the normalisation. If a definer name cannot be removed from the output, it is left in the output. Definer concept names cannot be removed if they originate with forgetting symbols in cyclic dependencies. Thus, the presence of definer names may imply cyclic axioms exist in the input ontology, where the presence of such definer names aids in the preservation of cyclic relations' entailments. In the case of UI-FAME, the tool simply removes any axiom that contains definer names, resulting in UIs that are under approximated.

NCIt The average precision rates for nested locality-based modules, semantic modules, and minimal subsumption modules were around 49%, 62%, and 85%, respectively.

Table 4.6: Computation time (s) of computing UIs from different modules of SNOMED CT by FAME, LETHE and NUI using 14 signatures obtained as extensions of the ERA refset

Tool	UI-FAME			LETHE			NUI		
$\mathcal{T}$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
Min.	0.10	0.14	0.16	1.45	0.61	0.59	0.03	0.02	0.02
Max.	3 600.00	3 600.00	3 600.00	3 600.00	19.23	1.79	0.14	0.08	0.04
Avg.	2 880.04	960.21	480.23	1 870.11	2.50	0.99	0.05	0.04	0.03
Med.	3 600.00	0.22	0.20	2 194.69	1.24	0.88	0.05	0.04	0.03
Succ.(%)	21.42	78.57	92.86	50.00	100.00	100.00	100.00	100.00	100.00

4.4.6 Interpolation Signature Partitioning

SNOMED CT - ERA refset We computed the three different SNOMED CT modules for each of the 14 signatures. These modules, together with their associated generating signatures, were then used as input for the systems UI-FAME, LETHE, and NUI, which were used to generate UIs. The times required to compute UIs are summarised in Table 4.6 in terms of the minimal, maximal, average, and median timings for all runs, including both successful and unsuccessful cases. The success rate was computed using successful cases that finished within a one-hour time limit.

As shown in Table 4.6, precomputing semantic modules ($\mathcal{M}_S$) and minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) significantly decreased computation time and thus boosted the success rate of generating UIs for LETHE to 100%. In particular, with minimal subsumption modules, LETHE computed UIs in less than 2 seconds for all signatures. In the case of semantic modules, LETHE computed UIs for all signatures in less than 20 seconds. In comparison, nested locality-based modules ($\mathcal{M}_\star$) dropped the success rates to 50% with an average of 32 minutes for LETHE.

In the case of UI-FAME, the success rate was approximately 93% when computing UIs using minimal subsumption modules and 79% when using semantic modules. As we can see, computing UIs from nested-locality-based modules is not viable with UI-FAME, as the success rate fell to 21.42%. On the other hand, the NUI tool successfully computed UIs based on any of the three module notions in terms of computation time and success rate.

We must bear in mind that finding minimal subsumption modules is more computationally expensive. The time required to compute the minimal subsumption modules for the 14 signatures varied between 1 and 15.65 minutes, whereas semantic and nested modules were computed in less than 5 seconds.

Table 4.7: Sizes of UIs from different modules by UI-FAME, LETHE and NUI on the NCIt ontology

Tool	UI-FAME			LETHE			NUI		
$\mathcal{T}$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
Min.	0	0	0	0	0	0	0	0	0
Max.	210	231	232	95	95	93	101	101	103
Avg.	32.94	35.77	48.65	18.46	15.70	18.26	20.18	20.18	22.98
Med.	25.00	23.00	28.00	12.00	10.00	12.00	13.00	13.00	14.50

As mentioned, using semantic modules and minimal subsumption modules instead boosts success rates to 100% for LETHE and NUI, and to about 79% (using semantic modules) and 93% (using minimal subsumption modules) for UI-FAME, as indicated in the final row of Table 4.6. This shows that the use of minimal subsumption modules and semantic modules enable the computation of UIs. This also illustrates that forgetting tools have forgotten less symbols while computing UIs from minimal subsumption modules or semantic modules, as opposed to nested modules, which require a bigger number of symbols to be forgotten.

4.4.7 Size of Computed UIs

Table 4.7 shows the size of UIs computed using the workflow for cancer types drugs signatures from the NCIt ontology. The results indicate that the sizes of the UIs computed by NUI and LETHE were almost the same. However, UIs generated with UI-FAME were 50% larger. This is because UIs computed by UI-FAME had axioms in normalised forms, with a high probability of redundant axioms. By contrast, LETHE improves the resulting UIs by eliminating tautological axioms and redundancies [Koo20]. This reduces the size of the UIs and improves their usability.

Additionally, we can see that the minimum size of UIs is zero, which indicates that certain UIs generated for cancer types drugs signatures from the NCIt ontology were empty. This could be due to the absence of relationships in the modified NCIt ontology version used in our evaluation, in which a number of cyclic axioms and those not expressed in $\mathcal{EL}$ were omitted.

As shown in Table 4.7, the results confirm that the size of UIs computed with LETHE and NUI is similar regardless of module type, where a UI is computed for a given interpolation signature.

Table 4.8: Size of axioms expressed in $\mathcal{ALC}$ in SNOMED CT computed by UI-FAME and LETHE in the NHS refset results

Tool	UI-FAME			LETHE		
$\mathcal{T}$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$	$\mathcal{M}_\star$	$\mathcal{M}_S$	$\mathcal{M}_\sqsubseteq$
Min.	1	1	2	1	1	1
Max.	71	174	428	18799	13201	9533
Avg.	11.78	23.72	47.96	696.12	484.63	308.77
Med.	4.00	8.00	14.50	28.00	48.50	37.00
|UIs with $\mathcal{ALC}$|	18	9	24	43	46	71

4.4.8 Axioms Not in the Input Ontology's Language

Given that the employed tools, LETHE and UI-FAME, are designed for the higher expressive language $\mathcal{ALC}$, while the input ontology language is in $\mathcal{EL}$, we checked whether the results contain axioms expressed in $\mathcal{ALC}$. These axioms include additional constructors that belong to $\mathcal{ALC}$ that are not expressible in $\mathcal{EL}$ including disjunction, universal role restrictions, and negation. UIs created with NUI, on the other hand, did not include such higher expressive constructors as its algorithm is designed to compute UIs for ontologies expressed in $\mathcal{EL}$.

SNOMED CT - NHS refsets Table 4.8 shows the number of axioms that are not in the source language of the input ontology for the UIs computed by UI-FAME and LETHE for SNOMED CT. The results indicate that the UIs generated using LETHE appear to have the most axioms in the $\mathcal{ALC}$ fragment in all of the computed UIs for the three module types. This may be due to the fact that LETHE provided a greater number of UIs than UI-FAME did, increasing the likelihood of such axioms being present in the UIs (see the last raw of Table 4.8). In the case of UIs computed using UI-FAME, the number of axioms expressed in $\mathcal{ALC}$ are least for UIs computed from nested locality-based modules. This could be because fewer UIs were computed in this case. In general, for UIs computed using LETHE, the fewest axioms expressed in $\mathcal{ALC}$ are in UIs based on minimal subsumption modules, whereas most emerge in UIs based on nested locality-based modules. We attribute this to the increased number of axioms that must be forgotten by LETHE from nested locality-based modules, given the larger number of axioms in such modules.

Table 4.9: No. of concept names in different modules of SNOMED CT

	$\#\mathrm{sig}_{\mathsf{N_C}}(\mathcal{M}_\star)$	$\#\mathrm{sig}_{\mathsf{N_C}}(\mathcal{M}_S)$	$\#\mathrm{sig}_{\mathsf{N_C}}(\mathcal{M}_\sqsubseteq)$
Min.	7	6	6
Max.	319	172	95
Avg.	97.9	47.1	24.4
Med.	96.0	41.0	14.5

NCIt We only found one axiom expressed in $\mathcal{ALC}$ in one of the UIs of the NCIt ontology computed using UI-FAME. The UI was based on both semantic modules and minimal subsumption modules. However, all the UIs computed using LETHE based on the three module types did not include $\mathcal{ALC}$ axioms.

4.5 Discussion

It was discovered that neither NUI, UI-FAME, nor LETHE were capable of computing UIs for SNOMED CT and NCIt ontologies without precomputing ontology modules in real-world signatures containing fewer than 35% of the ontology's symbols.

Table 4.10: No. of role names in different modules of SNOMED CT

	$\#\mathrm{sig}_{\mathsf{N_R}}(\mathcal{M}_\star)$	$\#\mathrm{sig}_{\mathsf{N_R}}(\mathcal{M}_S)$	$\#\mathrm{sig}_{\mathsf{N_R}}(\mathcal{M}_\sqsubseteq)$
Min.	0	0	0
Max.	17	15	8
Avg.	5.0	4.0	3.5
Med.	4.0	3.5	3.5

Table 4.11: No. of axioms in different modules of SNOMED CT

	$\#\mathcal{M}_\star$	$\#\mathcal{M}_S$	$\#\mathcal{M}_\sqsubseteq$
Min.	6	2	2
Max.	304	158	64
Avg.	92.7	41.8	14.8
Med.	91.0	38.0	6.5

In the majority of failed cases of our evaluation with the NHS refsets of SNOMED CT, where the signature contained more than 300 concept names, when computing UIs based on any type of module, UI-FAME and LETHE did not end in an acceptable

Table 4.12: No. of axioms in UIs of LETHE for modules in Table 4.11

	$\#\mathcal{UI}_\star$	$\#\mathcal{UI}_S$	$\#\mathcal{UI}_\sqsubseteq$
Min.	(2)	2	2
Max.	(20)	40	42
Avg.	(8.71)	10.43	10.14
Med.	(7.0)	7.5	6.5

length of time, whilst NUI ran out of memory. The results demonstrate that computing minimal subsumption ($\mathcal{M}_\sqsubseteq$) and semantic modules ($\mathcal{M}_S$) in advance can significantly accelerate the process of computing uniform interpolants, particularly for the tools UI-FAME and LETHE.

When utilising the workflow, the results indicate that forgetting on a semantic module should be attempted first because it was the quickest configuration for cases where forgetting succeeds. Alternatively, forgetting on a minimal subsumption module could be attempted, which may result in successful forgetting but may take longer to extract overall, because of the overhead of computing the module.

The remainder of this section discusses why module extraction methods can aid in optimising forgetting tools by examining the detailed statistics given in Tables 4.9–4.12 from the ERA refset experiments on computing UIs from SNOMED CT.

Smaller Module

According to Table 4.11, the size (number of axioms) of minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) was on average 2 times smaller than that of semantic modules ($\mathcal{M}_S$) and 5 times smaller than that of nested modules ($\mathcal{M}_\star$) (even 13 times smaller than that of nested modules according to the median value). The smaller the module, the greater the likelihood that the forgetting tool will succeed to compute UIs.

Fewer Symbols to Forget

As shown in Table 4.9, the number of concept names occurring in minimal subsumption modules ($\mathcal{M}_\sqsubseteq$) was 53% and 25%, respectively, of those occurring in semantic modules ($\mathcal{M}_S$) and nested modules ($\mathcal{M}_\star$). Due to the fact that the interpolation signatures were identical for all modules, forgetting symbols from minimal subsumption modules and semantic modules required the fewest symbols to forget. Although forgetting role names is more difficult than forgetting concept names, the average number

of role names varied little across modules, as seen in Table 4.10. In general, computing UIs on minimal subsumption modules was faster, particularly for LETHE and UI-FAME.

Special Role 'RoleGroup'

In SNOMED CT, a special role name called 'RoleGroup' is used to preserve proper inferences and semantic meaning for complicated concept expressions involving, for example, several sites and morphologies [SDMW02]. The existence of 'RoleGroup' implies the presence of nested existential restrictions. We discovered that the existence of nested existential restrictions in the input ontology consumed lots of memory. This coincides with our observations made in Section 4.3, where we found that if forgetting symbols that occur as filler names for nested existential restrictions were added to the interpolation signature, forgetting can be terminated within a reasonable amount of time; otherwise, forgetting can take a long time to compute and may fail to terminate in some cases.

Inspection indicated that semantic modules ($\mathcal{M}_S$), particularly minimal subsumption modules ($\mathcal{M}_{\sqsubseteq}$), included a far smaller number of axioms containing the role 'RoleGroup'.

Size of UIs

Table 4.12 shows the sizes of LETHE's UIs in the experiment with the ERA refset. The bracketed numbers are computed only for successful cases. For semantic and minimal subsumption modules the success rates were 100%. Additionally, we found that the UIs computed were similar in size. This confirms our hypothesis that UIs computed using any type of module are equal in size when the interpolation signature is the same.

4.6 Conclusion

Through our evaluation of the workflow, we discovered several limitations when computing UIs against SNOMED CT and NCIt ontologies with real-world signatures (NHS refsets and ERA-refset of SNOMED CT and Cancer drug names of NCIt). The first limitation is the lengthy computation time required to compute UIs for SNOMED CT and NCIt ontologies without precomputing ontology modules for real-world signatures containing fewer than 35% of the ontology's symbols. A second limitation relates to

the language of the resulting uniform interpolants, which is highly dependent on the forgetting tool used. If the target language of the method is $\mathcal{ALC}$ and the input ontology is stated in $\mathcal{EL}$, the resulting UIs may not match the source language exactly; instead, they will include expressive constructors that the method supports, such as negation and universal role restrictions. A third limitation is that the created UIs may contain new concept names (definer names) that are not part of the input ontology, if such definer names are necessary to be retained. These definer names aid in the preservation of the entailments of cyclic axioms.

Our evaluation showed that forgetting tools must forget fewer symbols when computing UIs from minimal subsumption or semantic modules than when computing UIs from nested locality-based modules, which require a greater amount of symbols to be forgotten. This implies that forgetting from minimal subsumption modules or semantic modules takes significantly less time than forgetting from nested locality-based modules, as confirmed by our evaluation.

Through an empirical evaluation with MRI refset of SNOMED CT, we identified instances in which forgetting becomes computationally difficult. This was accomplished by analysing the location of forgetting symbols in the axioms of the input ontology. The results indicate that when forgetting symbols occur as filler names for nested existential restrictions, the forgetting process takes a long time to terminate and may fail in most cases.

We have discussed in Section 4.1 the issue of symbols selection for the purpose of computing extracts using modularisation and uniform interpolation. We showed using SNOMED CT examples that the resultant module or uniform interpolant is very dependent on the behaviour of the method and the relationships between the symbols in the input ontology. Even though it is well-known that SLBM generates large modules for a small number of input symbols, we have seen that nested locality-based modules of SLBM can generate empty modules when the specified input signatures are not connected via subsumption relationships in the input ontology, as is typically the case for symbols found in SNOMED CT reference sets.

We showed that modifying the input signature, either by extension or dividing it into a smaller group of closely related terms, results in the development of more useful UIs, as confirmed by one of the SNOMED CT terminologists.

We examined three forgetting tools in this chapter: NUI, UI-FAME, and LETHE. At the time of the experiments, the forgetting tool described in [WDL+20], which works with $\mathcal{ELH}$ ontologies had not yet been developed.

Chapter 5

Focus Set Subontologies

In this chapter, we introduce a notion of subontology based on the idea of a certain normal form we call abstracted definitions, because SNOMED CT users are already familiar with a variety of normal forms [Spa01, Har]. (Section 2.4.1 discusses the different normal forms used in the SNOMED CT community). Our abstracted definitions follow the same format as the commonly used canonical form. Long canonical forms explicitly state all possible constraints and defining characteristics of particular concepts to facilitate implementation, recording, storage, and retrieval within SNOMED CT. Such forms enable more precise inferred parent identification of concepts after running a classifier, they simplify parent relationship maintenance, and improve the accuracy and breadth of super and subconcepts [Ulr].

Different from modularisation and uniform interpolation, our notion of subontologies provides extracts that *define* the set of input symbols. We extract a subontology that focuses exclusively on definitions and axioms associated with the input set of symbols, resulting in more compact extracts. In a nutshell, the aim is to extract the definitions and any necessary axioms for a collection of input symbols while preserving the subsumption relations that relate to the extracted definitions' symbols.

To illustrate the core principle of our notion of subontologies, we begin with a broad example. Consider the ontology illustrated in Figure 5.1 (a), which is divided

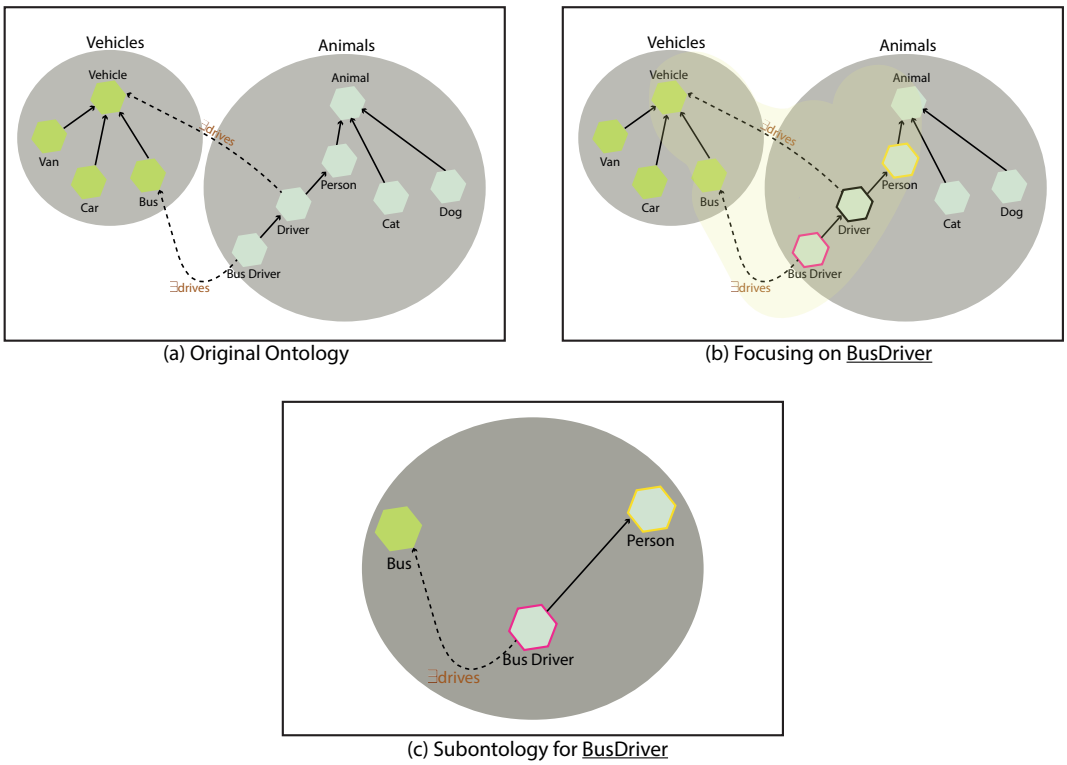

Figure 5.1: An illustration of generating a focus set subontology for the concept *Bus Driver*

into two main concept hierarchies: *Animals* and *Vehicles*. Assume the ontology consists of the following axioms:

$$\boxed{Bus\ Driver \equiv Driver \sqcap \exists drives.Bus} \tag{5.1}$$

$$\boxed{Driver \equiv Person \sqcap \exists drives.Vehicle} \tag{5.2}$$

$$\boxed{Person \sqsubseteq Animal} \tag{5.3}$$

$$Cat \sqsubseteq Animal \tag{5.4}$$

$$Dog \sqsubseteq Animal \tag{5.5}$$

$$\boxed{Bus \sqsubseteq Vehicle} \tag{5.6}$$

$$Car \sqsubseteq Vehicle \tag{5.7}$$

$$Van \sqsubseteq Vehicle \tag{5.8}$$

Further, assume a user is interested in extracting a subontology for the focus concept ***Bus Driver***, we can see upward relationships connecting ***Bus Driver*** to the concepts *Animal* and *Vehicle* as highlighted in Yellow in Figure 5.1 (b), and also represented by the axioms (5.1)–(5.3) and (5.6). If we wish to extract a subontology for ***Bus Driver***, we may need to include all of the concepts with which ***Bus Driver*** is connected. However, we seek to abstract away concepts and axioms from the subontology that have the same information as the focus concept. As can be seen in the ontology, *Driver* in axiom (5.2) is defined in terms of concepts inherited by ***Bus Driver***.

To ensure that the subontology contains all of the required conditions for a complete definition of ***Bus Driver***, the concept ***Bus Driver*** can be defined in terms of the closest primitive concept that is upwardly connected to, which is the concept *Person* as shown in Blue.

As a result, the resulting subontology depicted in Figure 5.1 (c) has the complete definition of the focus concept of interest (***Bus Driver***), but not definitions of other concepts. The resulting subontology contains the following axiom:

$$\textbf{\textit{Bus Driver}} \equiv \textit{Person} \sqcap \exists \textit{drives.Bus} \tag{5.9}$$

Figure 5.2 shows an example of an $\mathcal{ELH}$ biomedical terminology, adapted from [CAH08]. The axioms α_1 to α_7 are concept definitions or concept inclusions of the form $A \equiv C$ or $A \sqsubseteq C$, where A is the described concept. The example illustrates concepts relating to disorders characterised by inflammation. This example will be used throughout this chapter.

α_1 : **InflammatoryDisorder** $\equiv$ Disease $\sqcap$ $\exists$involves.Inflammation,
α_2 : **LiverDisease** $\equiv$ Disease $\sqcap$ $\exists$location.Liver,
α_3 : **Hepatitis2** $\equiv$ LiverDisease $\sqcap$ $\exists$involves.Inflammation,
α_4 : **ViralHepatitis1** $\sqsubseteq$ InflammatoryDisorder $\sqcap$ $\exists$location.Liver,
α_5 : **ViralHepatitis2** $\sqsubseteq$ LiverDisease $\sqcap$ $\exists$involves.Inflammation,
α_6 : **LargeLiver** $\sqsubseteq$ LiverDisease $\sqcap$ $\exists$location.EntireLiver,
α_7 : **EntireLiver** $\sqsubseteq$ Liver

Figure 5.2: An example of a biomedical terminology

In the following sections we formally define our notion of subontology and study its properties. In Section 5.1, we introduce the notion of upwardly abstracted definitions, on which the subontologies are based. The requirements that subontologies

must meet, the algorithm to compute focus set subontologies, and the method's properties are discussed in Section 5.2. In Section 5.3, we present two methods for ensuring that the created subontologies satisfy the requirements. Section 5.4 compares our notion of subontologies to the properties of related ontology extraction methods including locality-based modularisation and uniform interpolation. Section 5.5, presents an evaluation of the subontology generation method, which is divided into eight subsections including a brief description of our implementation (Section 5.5.1), the used tools (Section 5.5.2), the used data (Section 5.5.3), the conducted experiments (Section 5.5.4), and the results of our experiments are included in Sections (5.5.5–5.5.8). In Section 5.6, we discuss the main points we learned from the evaluation. Lastly, we conclude in Section 5.7.

5.1 Upwardly Abstracted Definitions

Our notion of subontologies is based on the idea of a normal form called abstracted definitions. Our abstracted definitions follow the same format as the commonly used canonical form in SNOMED CT [Spa01, Har] (see Section 2.4.1 for a description of the different normal forms in SNOMED CT).

To define abstracted definitions, we begin with the following fundamental definition, which refers to a concept's status in an ontology as being *defined* or *primitive*.

Definition 13 (Defined and Primitive Concepts)**.** *Let $\mathcal{O}$ be an ontology, A a concept name and C an $\mathcal{EL}$-concept other than A. We say A is a* defined concept name *in $\mathcal{O}$ if there is an axiom of the form $A \equiv C$ in $\mathcal{O}$. The axiom $A \equiv C$ is then referred to as a* concept definition.

Otherwise, A is called a primitive concept name.

In a concept definition $A \equiv C$, the concept C represents the necessary and sufficient conditions that define the concept name A. We refer to axioms of the form $A \sqsubseteq D$ to as *concept descriptions*, in which the concept name A is described by the concept D. The concept D represents the necessary (upward) condition that describes A.

Regardless as to whether a concept name A is defined or primitive, its abstracted definition is intended to be expressed using its closest primitive ancestor(s) in the subsumption hierarchy. Closest primitive ancestors are defined as follows.

Definition 14 (Closest Primitive Ancestor)**.** *Let $\mathcal{O}$ be an ontology, and suppose A is a defined or a primitive concept name in $\mathcal{O}$ and P is a primitive concept name in $\mathcal{O}$.*

We say that P is a closest primitive ancestor *to A in $\mathcal{O}$ if $\mathcal{O} \models A \sqsubseteq P$ and there does not exist a primitive concept name Z in $\mathcal{O}$ (other than P or A) such that $\mathcal{O} \models A \sqsubseteq Z$ and $\mathcal{O} \models Z \sqsubseteq P$. The conjunction of closest primitive ancestors to A will be denoted by $\mathcal{P}_A^{\sqcap}$.*

We define abstracted definitions as follows.

Definition 15 (Upwardly Abstracted Definition)**.** *Let $\mathcal{O}$ be an ontology, and A a concept name in $\mathcal{O}$. An* upwardly abstracted definition *of A is:*

1. *$A \equiv \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$ when A is a defined concept name in $\mathcal{O}$,*
2. *$A \sqsubseteq \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$ when A is a primitive concept name in $\mathcal{O}$,*

where $\mathcal{P}_A^{\sqcap}$ is a conjunction of the closest primitive ancestors to A, while $\mathcal{E}_A^{\sqcap}$ is a conjunction of inferred existential restrictions (of the form $\exists r.C$) such that $\mathcal{O} \models A \equiv \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$ ($\mathcal{O} \models A \sqsubseteq \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$), respectively and $sig(\mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}) \subseteq sig(\mathcal{O})$.

In Definition 15, we use the term *abstracted definitions* not only in the case 1 where A is equivalent to $\mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$, but also the case 2 where $\mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$ is only a necessary condition for A. Abstracted definitions for primitive concept descriptions are distinct from conventional definitions (where the concept is defined sufficiently by necessary and sufficient conditions $A \equiv C$). The word definition in the phrase *abstracted definition* thus has a wider interpretation. It refers to the normal form defined in Definition 15.

Example 7. *Consider the ontology*

$$\begin{aligned}\mathcal{O} = \{&A \equiv D \sqcap \exists r.C_1,\\ &D \equiv P \sqcap \exists r.C_2,\\ &P \sqsubseteq \exists r.C_3\}\end{aligned}$$

An upwardly abstracted definition of A is $A \equiv P \sqcap \exists r.C_1 \sqcap \exists r.C_2 \sqcap \exists r.C_3$ in which P is the closest primitive concept above A in the subsumption hierarchy.

As mentioned in Section 2.4.1, different equivalent logical forms of SNOMED CT concept definitions are discussed in [Spa01]. Two distinct forms of proximal primitive modelling is mentioned there: first, a *short canonical form* in which only the existential restrictions that distinguish the concept from its closest primitive ancestors are listed.

For instance, in Example 7, the short canonical form of A is $A \equiv P \sqcap \exists r.C_1 \sqcap \exists r.C_2$. The second is the *long canonical form*, which lists all inferred existential restrictions. The long canonical form can be viewed as including all defining characteristics of the concept A [Spa01]. In Example 7, the long canonical form of A is $A \equiv P \sqcap \exists r.C_1 \sqcap \exists r.C_2 \sqcap \exists r.C_3$. The concept $\exists r.C_3$ is logically superfluous in the definition of A because it is subsumed by the concept P. However, it is advantageous to include for the benefit of modellers, because it obviates the need to check the definitions of the parent concepts for possible further existential restrictions of A (in this case the parent concept P).

As mentioned in [Spa01], there are several equivalent logical representations, for a given concept definition (or description). From any of these representations the same (short or long) long canonical form can be derived. This makes canonical forms useful for comparing concept definitions for analytical or debugging reasons. Long canonical forms are particularly useful as they include all of the concept's defining characteristics, allowing concept descriptions to be easily compared. For instance, producing the long canonical forms of the concepts *ViralHepatitis1* and *ViralHepatitis2* in Figure 5.2 results in an identical description for both concepts:

$$\text{ViralHepatitis1} \sqsubseteq \text{Disease} \sqcap \exists\text{involves.Inflammation} \sqcap \exists\text{location.Liver},$$
$$\text{ViralHepatitis2} \sqsubseteq \text{Disease} \sqcap \exists\text{involves.Inflammation} \sqcap \exists\text{location.Liver}$$

Both short and long canonical forms satisfy the conditions of Definition 15, and are therefore abstracted definitions. This illustrates that abstracted definitions are not unique. The following example illustrates that abstracted definitions may also be weaker than original definitions when $\mathcal{O}$ is an *ontology* rather than a *terminology*.

Example 8. *Let $\mathcal{O} = \{\alpha_1, \alpha_2, \alpha_3\}$, where*

$$\alpha_1 : A \equiv D \sqcap \exists r.C,$$
$$\alpha_2 : D \equiv P \sqcap \exists r.C,$$
$$\alpha_3 : A \sqsubseteq P_2.$$

We notice that both ab_1: $A \equiv P \sqcap \exists r.C$ and ab_2: $A \equiv P \sqcap P_2 \sqcap \exists r.C$ are entailed by $\mathcal{O}$ and are in fact abstracted definitions of A. Let's consider the ontologies $\mathcal{O}_1 = \{ab_1, \alpha_2, \alpha_3\}$ and $\mathcal{O}_2 = \{ab_2, \alpha_2, \alpha_3\}$ in which the original definition α_1 of A is respectively replaced by ab_1 and ab_2. We observe that $\mathcal{O} \equiv \mathcal{O}_1$ but $\mathcal{O} \not\equiv \mathcal{O}_2$ because

$\mathcal{O}_2 \not\models \mathcal{O}$ as $\mathcal{O}_2 \not\models \alpha_1$ since ab_2 is weaker than α_1.

To prove the claim $\mathcal{O} \equiv \mathcal{O}_1$, we break the $\equiv$ relation into the two entailment directions $\mathcal{O} \models \mathcal{O}_1$ and $\mathcal{O}_1 \models \mathcal{O}$.

***1. Claim:** $\mathcal{O} \models \mathcal{O}_1$ and $\mathcal{O}_1 \models \mathcal{O}$:*

Proof:

1 a. $\mathcal{O} \models \mathcal{O}_1$. This follows if we can show that every $\alpha \in \mathcal{O}_1$ is entailed by $\mathcal{O}$. Clearly $\mathcal{O} \models \alpha_2$ and $\mathcal{O} \models \alpha_3$ since α_2 and α_3 both belong to $\mathcal{O}$. $\mathcal{O} \models ab_1$ since $A \equiv D \sqcap \exists r.C \equiv P \sqcap \exists r.C \sqcap \exists r.C$ in $\mathcal{O}$ by equivalent replacement of D by its definition $P \sqcap \exists r.C$. Simplifying this shows $\mathcal{O} \models ab_1$.

1 b. $\mathcal{O}_1 \models \mathcal{O}$. The easy cases are again the common axioms α_2 and α_3. We need to show that $\mathcal{O}_1 \models \alpha_1$:$A \equiv D \sqcap \exists r.C$. Note that $\mathcal{O}_1 \models ab_1$:$A \equiv P \sqcap \exists r.C$ since $P \sqcap \exists r.C \equiv P \sqcap \exists r.C \sqcap \exists r.C$ and using α_2:$D \equiv P \sqcap \exists r.C$ by equivalent replacement gives $\mathcal{O}_1 \models \alpha_1$.

To prove the claim $\mathcal{O} \not\equiv \mathcal{O}_2$, we show that $\mathcal{O} \models \mathcal{O}_2$ but $\mathcal{O}_2 \not\models \mathcal{O}$.

***2. Claim:** $\mathcal{O} \models \mathcal{O}_2$ but $\mathcal{O}_2 \not\models \mathcal{O}$*

Proof:

2 a. $\mathcal{O} \models \mathcal{O}_2$. This follows if we show that every $\alpha \in \mathcal{O}_2$ is entailed by $\mathcal{O}_2$. $\mathcal{O} \models \alpha_2$ and $\mathcal{O} \models \alpha_3$ since α_2 and α_3 both belong to $\mathcal{O}$. To prove that $\mathcal{O} \models ab_2$ we begin by proving the left to right direction:

$A \sqsubseteq D \sqcap \exists r.C$ from α_1:$A \equiv D \sqcap \exists r.C$ we have that
$A \sqsubseteq P \sqcap \exists r.C \sqcap \exists r.C$ (by equivalent replacement) using α_2: $D \equiv P \sqcap \exists r.C$
$A \sqsubseteq P \sqcap \exists r.C$ (by simplification) since $\exists r.C \sqcap \exists r.C \equiv \exists r.C$
$A \sqsubseteq P \sqcap P_2 \sqcap \exists r.C$ using α_3: $A \sqsubseteq P_2$ (since in general $X \sqsubseteq Y_2$ implies $X \sqsubseteq Y_1 \sqcap Y_2$)
Proving the right to left direction and starting with $D \sqcap \exists r.C \sqsubseteq A$ from α
$P \sqcap \exists r.C \sqcap \exists r.C \sqsubseteq A$ (by equivalent replacement) using α_2: $D \equiv P \sqcap \exists r.C$
$P \sqcap \exists r.C \sqsubseteq A$ (by simplification)
$P \sqcap P_2 \sqcap \exists r.C \sqsubseteq A$ (by conjunction introduction in RHS of A)
From the conclusion of both directions, we can see that $\mathcal{O} \models ab_2$.

2 b. However, the opposite direction does not hold, i.e. $\mathcal{O}_2 \not\models \mathcal{O}$. To show this, it suffices to give an axiom in $\mathcal{O}$ that is not entailed by $\mathcal{O}_2$. We show $\mathcal{O}_2 \not\models \alpha_1$ by showing $\mathcal{O}_2 \not\models D \sqcap \exists r.C \sqsubseteq A$.
This can be proved by contradiction. If $\mathcal{O}_2 \models D \sqcap \exists r.C \sqsubseteq A$ then $\mathcal{O}_2$ together

with the negation of $D \sqcap \exists r.C \sqsubseteq A$ *is unsatisfiable*
$ab_2 : A \equiv P \sqcap P_2 \sqcap \exists r.C$
$\alpha_2 : D \equiv P \sqcap \exists r.C$
$\alpha_3 : A \sqsubseteq P_2$
Negating $D \sqcap \exists r.C \sqsubseteq A$ *gives us:*
$(\neg A)(a)$, $D(a)$, $\exists r.C(a)$, *where* a *is a fresh constant*
From $(\neg A)(a)$ *and* ab_2 *(*$\sqsupseteq$*-direction) we derive:*
$(\neg P \sqcup \neg P_2 \sqcup \neg \exists r.C)(a)$
By splitting we can derive: $(\neg P_2)(a)$
No more inferences are possible on this branch, therefore
$\mathcal{O}_2 \cup \{(\neg A)(a), D(a), \exists r.C(a)\}$ *is satisfiable. This proves* $\mathcal{O}_2 \not\models D \sqcap \exists r.C \sqsubseteq A$ *and therefore the claim.*

The abstracted definition of A in $\mathcal{O}$ computed by our algorithm presented in Section 5.2.2 is the second case (ab_2) as part of its computation of all ancestor concepts that subsume A. As seen in the example, this definition is weaker than the original definition.

The following definition is used to determine the logical strength of an abstracted definition in the case that the generated abstracted definition is weaker than the original definition.

Definition 16 (Weaker Abstracted Definition)**.** *Let* $\mathcal{O}$ *be an ontology,* $A \equiv C$ *be a definition in* $\mathcal{O}$*, and* $A \equiv D$ *is an abstracted definition of* A *where* A *is a defined concept name and* C *and* D *are* $\mathcal{EL}$*-concepts.* $A \equiv D$ *is said to be a* weaker abstracted definition *than* $A \equiv C$ *in* $\mathcal{O}$
if $\mathcal{O} \models \{A \equiv D\}$ *but* $\mathcal{O} \backslash \{A \equiv C\} \cup \{A \equiv D\} \not\models \{A \equiv C\}$.

5.2 Computing Subontologies

5.2.1 Requirements

Our aim is to compute for a given set of symbols a domain-specific *subontology* from a source ontology that satisfies the following requirements:

1. The subontology must capture the meaning of the concepts of focus, using whenever possible abstracted definitions in long canonical form.

Algorithm 10 SubontologyExtraction($\mathcal{O}$, Σ_F)

Input: Ontology $\mathcal{O}$, Focus set Σ_F
Output: Subontology $\mathcal{S}$

1: $\mathcal{S} := \emptyset$, $\Sigma^+ := \Sigma_F$.
2: $\mathcal{O}^d :=$AddDefiners($\mathcal{O}$)
3: $\mathcal{H} :=$ Classify($\mathcal{O}^d$)
4: **for** $A \in \Sigma_F$ **do**
5: **if** A is a defined concept name in $\mathcal{O}^d$ **then**
6: $A \equiv \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap} :=$ AbstractedDefinitionExtraction(A, $\mathcal{O}^d$, $\mathcal{H}$, TRUE)
7: **else** $A \sqsubseteq \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap} :=$ AbstractedDefinitionExtraction(A, $\mathcal{O}^d$, $\mathcal{H}$, FALSE)
8: $\mathcal{S} := \mathcal{S} \cup \{A \equiv (\sqsubseteq)\ \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}\}$
9: $\Sigma^+ := \Sigma^+ \cup$ sig($A \equiv (\sqsubseteq)\ \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$)
10: $\mathcal{S} := \mathcal{S} \cup$ ComputeAdditionalAxioms(Σ^+, $\mathcal{H}$, $\mathcal{O}^d$)

2. The transitive closure of concept name subsumption of the subontology is a restriction of the original ontology's transitive closure of concept name subsumption over the signature of the subontology.

3. The subontology must capture the role inclusion axiom when the symbol of focus is a role name.

These requirements were established with a leading terminologist at SNOMED International. The first requirement states that the definitions of the concepts of interest in the subontology should have the same meaning as their definitions in the original ontology in the form of an abstracted definition. We refer to such concepts of interest as the *focus concepts*. The second requirement asserts that when we classify the resulting subontology, we obtain a correct concept hierarchy that is a restriction on the concept hierarchy of the original ontology over the signature of the subontology.

5.2.2 Algorithm

Our method to compute subontologies is presented in Algorithm 10. The algorithm takes as input an ontology $\mathcal{O}$ and a focus set Σ_F of concept and role names. The output is a subontology of abstracted definitions of the focus concepts in Σ_F that also captures the subsumption relationships among concept and roles names. The first step of the algorithm initialises the output $\mathcal{S}$, and Σ^+ is set to Σ_F.

Line 2 introduces fresh concept names denoted as definer names for each existential restriction in the input ontology. The strategy of introducing fresh definer names

Algorithm 11 AbstractedDefinitionExtraction(A, $\mathcal{O}^d$, $\mathcal{H}$, *isDefinedConcept*)

Input: The concept to define A, ontology with definers $\mathcal{O}^d$, concept hierarchy $\mathcal{H}$, defined concept checker *isDefinedConcept*
Output: The abstracted definition of A

1: $\uparrow A$:= ComputeAncestorsOf$A(A, \mathcal{H})$
2: $\mathcal{P}'_A$:= GetPrimitiveAncestors($\uparrow A$, $\mathcal{O}^d$)
3: $D'_{\mathcal{E}_A}$:= GetDefinerNames($\uparrow A$, $\mathcal{O}^d$)
4: $\mathcal{P}^{\sqcap}_A$:= RemoveRedundantConcepts($\mathcal{P}'_A$, $\mathcal{H}$)
5: $D_{\mathcal{E}_A}$:= RemoveRedundantConcepts($D'_{\mathcal{E}_A}$, $\mathcal{H}$)
6: $\mathcal{E}^{\sqcap}_A$:= ReplaceDefinersWithExistentialRestrictions($D_{\mathcal{E}_A}$)
7: **if** isDefinedConcept **then**
8: return $A \equiv \mathcal{P}^{\sqcap}_A \sqcap \mathcal{E}^{\sqcap}_A$
9: **else** return $A \sqsubseteq \mathcal{P}^{\sqcap}_A \sqcap \mathcal{E}^{\sqcap}_A$

Algorithm 12 AddDefiners($\mathcal{O}$)

Input: Ontology $\mathcal{O}$
Output: Ontology with definers $\mathcal{O}^d$

1: Mapper.(Key, $\exists r_i.B_i$) := $\emptyset$, $\mathcal{E}_{\mathcal{O}} := \emptyset$
2: $\mathcal{O}^d := \mathcal{O}$
3: $\mathcal{E}_{\mathcal{O}}$:= GetExistentialRestrictions($\mathcal{O}$)
4: **for** $\exists r_i.B_i \in \mathcal{E}_{\mathcal{O}}$ **do**
5: Mapper.Put(D_i, $\exists r_i.B_i$)
6: $\mathcal{O}^d := \mathcal{O}^d \cup \{D_i \equiv \exists r_i.B_i\}$
7: Return $\mathcal{O}^d$

to make inference rules applicable to the ontology under processing has been done by many authors in the literature [Koo15, Nik12, Zha18]. In our algorithm, these definer names are introduced for every existential restriction of the form $\exists r_i.B_i$ where r is a role name and B is a concept name, by adding an equivalent axiom that states that a definer name is equivalent to an existential restriction as Algorithm 12 shows, which returns the ontology $\mathcal{O}^d$. This strategy allows us to leverage the ELK reasoner's inference capabilities to eliminate redundant existential restrictions via the computed concept hierarchy of $\mathcal{O}^d$.

In Line 3, we classify the ontology $\mathcal{O}^d$ using ELK, where $\mathcal{O}^d$ is the ontology after introducing the definer names, to obtain the concept hierarchy $\mathcal{H}$.

Computing an abstracted definition for a focus concept $A \in \Sigma_F$ is done using the method AbstractedDefinitionExtraction presented in Algorithm 11. The method

Algorithm 13 ReplaceDefinersWithExistentialRestrictions($D_{\mathcal{E}_A}$)

Input: Definer names $D_{\mathcal{E}_A}$
Output: Existential restrictions $\mathcal{E}_A$

1: $\mathcal{E}_A^{\sqcap} := \emptyset$
2: **for** $D_i \in D_{\mathcal{E}_A}$ **do**
3: $\quad \mathcal{E}_A^{\sqcap} := \mathcal{E}_A^{\sqcap} \cup$ Mapper.get(D_i)
4: Return $\mathcal{E}_A^{\sqcap}$

starts with computing the ancestors ($\uparrow A$) of A in the concept hierarchy $\mathcal{H}$. The ancestors set $\uparrow A$ of A is the set of all upward conditions of the focus concept A.

The function GetPrimitiveAncestors in Line 2 filters the set of ancestors by checking their status in $\mathcal{O}^d$, whether they are defined or primitive concept names, to compute only the primitive concept ancestors, denoted by $\mathcal{P}'_A$.

In Line 3, the function GetDefinerNames collects the set of definer names from the ancestors set and adds them to the set $D'_{\mathcal{E}_A}$.

From the set of primitive concepts $\mathcal{P}'_A$, the function RemoveRedundantConcepts in Line 4 computes the closest primitive ancestors $\mathcal{P}_A^{\sqcap}$ by removing redundant primitive concepts. In order to compute the most specific existential restrictions, we reduce the set of definer names $D'_{\mathcal{E}_A}$ by removing redundant concepts using the function RemoveRedundantConcepts. These redundant concepts are removed according to the general rule $C \sqcap D \equiv C \Leftrightarrow C \sqsubseteq D$, which states that D is redundant when it is a sibling of C and subsumes C. The function RemoveRedundantConcepts removes redundant concepts using the concept hierarchy $\mathcal{H}$, by determining whether two concept names subsume each other in the concept hierarchy $\mathcal{H}$ in a given set ($\mathcal{P}'_A$ or $D'_{\mathcal{E}_A}$).

As seen in Lines 4 and 5 of Algorithm 11, we delete redundant concepts in two steps: one for the primitive concepts $\mathcal{P}'_A$ (Line 4) and another for the set of definer names $D'_{\mathcal{E}_A}$ (Line 5). This is done to produce an abstracted definition in long canonical form, which allows for possible redundant existential restrictions that is subsumed by a concept name in the same definition, as seen in Example 7.

In Line 6, we replace the introduced definer names with the existential restrictions using the function ReplaceDefinersWithExistentialRestrictions presented in Algorithm 13. The last step of Algorithm 11 returns the abstracted definition of A as either $A \equiv \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$ or $A \sqsubseteq \mathcal{P}_A^{\sqcap} \sqcap \mathcal{E}_A^{\sqcap}$ depending on whether A is a defined or a primitive concept name in the input ontology.

Line 9 of Algorithm 10 adds the signature (the concept and role names) of the generated abstracted definition to the set Σ^+. Line 10 returns the subontology $\mathcal{S}$ after using the function ComputeAdditionalAxioms to add extra axioms to complete the concept hierarchy of the subontology. This is performed by looking for possible subsumption relations between concept and role names in the set Σ^+. For example, the axiom α_3 in Figure 5.3 below is an additional axiom that is required for the subontology concept hierarchy to be complete.

Notably, we also compute inclusion axioms of the type $A \sqsubseteq \exists r.B$, where A is subsumed by an existential restriction, as long as the signature of the existential restriction is included in the signature set Σ^+. The computation of such axioms is done by looking in the concept hierarchy $\mathcal{H}$ for a definer name that A is subsumed by.

Then, we replace the definer name for the existential restriction that it defines using Algorithm 13.

In terms of dealing with input ontologies that allow the presence of cyclic axioms, this is reliant on the types of cycles present. If the input ontology contains cyclic existential axioms of the form $A \equiv D \sqcap \exists r.A$, then these axioms do not pose an issue for our method to derive an abstracted definition for the concept A. This is because during the computation of abstracted definitions, our method does not require the replacement of defined concepts by the closest primitive ancestors where such defined concepts exist as filler names of existential restrictions. Moreover, the ELK reasoner can handle ontologies with cyclic axioms [KKS12a], and our algorithm relies primarily on ELK to compute abstracted definitions.

If cycles of the following form exist in an ontology $\mathcal{O}$:

$$A \equiv D \sqcap \exists r.B_1,$$
$$D \equiv A \sqcap \exists r.B_2,$$

and we assume that we want extract a subontology for the focus concept A, then, our method derives the axiom α: $A \equiv \exists r.B_1 \sqcap \exists r.B_2$ which is incorrect. The definition α is in this form because of the step that filters the set of ancestors based on whether they are defined or primitive concept names in $\mathcal{O}$ to get only the primitive concepts ancestors (Line 2 of Algorithm 11), where such primitive concept names do not appear in this ontology.

5.2.3 Theoretical Properties

Our method abstracts the definitions based on the closest primitive ancestors in order to include in the resulting subontology only what is truly necessary. For example, if a user is interested in computing a subontology using the ontology in Figure 5.2 for the focus set concepts A_1:**Hepatitis2** and A_2:**LargeLiver**, then applying Algorithm 10 for A_1 and A_2 results in the subontology shown in Figure 5.3. It includes abstracted definitions of A_1 and A_2, where the concept A_3:LiverDisease occurring in the original definitions of A_1 and A_2 has been abstracted away. The use of A_3 would require the inclusion of its definition and increase the size of the desired subontology without adding additional meaning. This is because the information carried by A_3 is inherited by A_1 and A_2, and thus, the definition of A_3 is superfluous for the *focus set* $\{A_1, A_2\}$ of interest to the user.

We refer to the extra symbols occurring in the signature of the generated focus set definitions as the *supporting symbols (concept and role names)*. For example, the concept *EntireLiver* occurring in the definition of **LargeLiver** is a supporting concept.

α_1 : **Hepatitis2** $\equiv$ Disease $\sqcap$ $\exists$involves.Inflammation $\sqcap$ $\exists$location.Liver
α_2 : **LargeLiver** $\sqsubseteq$ Disease $\sqcap$ $\exists$location.EntireLiver
α_3 : EntireLiver $\sqsubseteq$ Liver

Figure 5.3: The subontology for focus concepts A_1:**Hepatitis2** and A_2:**LargeLiver**

We define a subontology for a given set Σ_F of focus symbols as follows.

Definition 17 (Focus Set Subontology)**.** *Let $\mathcal{O}$ be an $\mathcal{ELH}$ ontology and Σ_F the focus set of concept and role names. $\mathcal{S}$ is a* focus set subontology *of $\mathcal{O}$ for Σ_F if the following conditions are satisfied:*

- $\Sigma_F \subseteq sig(\mathcal{S})$*;*
- *for every $\mathcal{ELH}$-axiom α where $sig(\alpha) \subseteq sig(\mathcal{S})$ we have:*
  1. *If $\mathcal{S} \models \alpha$ then $\mathcal{O} \models \alpha$, and*
  2. *if α is of the form $A \sqsubseteq B$, then we have $\mathcal{S} \models \alpha$ when $\mathcal{O} \models \alpha$ where A and B are concept names.*

A subontology $\mathcal{S}$ can be partitioned into two main sets of axioms, the *focus set axioms* and *the supporting set axioms*.

Definition 18 (Focus Set Axioms)**.** *Let $\mathcal{S}$ be a focus set subontology for the focus set Σ_F. Axioms of the form $A \sqsubseteq C$, $A \equiv C$, or $r \sqsubseteq s$ in $\mathcal{S}$ are* focus set axioms *where A is a focus concept, and r is a focus role in Σ_F and C is an $\mathcal{EL}$-concept. The set of focus set axioms in $\mathcal{S}$ is denoted by AX_{Σ_F}.*

We define the set Σ_S of supporting symbols in the subontology $\mathcal{S}$ as follows.

Definition 19 (Supporting Set Σ_S)**.** *Let $\mathcal{S}$ be a focus set subontology for the focus set Σ_F. The* set of supporting symbols *of $\mathcal{S}$ is defined as $\Sigma_S = sig(\mathcal{S}) \setminus \Sigma_F$.*

Definition 20 (Supporting Set Axioms)**.** *Let $\mathcal{S}$ be a focus set subontology for the focus set Σ_F, Σ_S is the supporting symbol set of $\mathcal{S}$ and AX_{Σ_F} is the focus set axioms in $\mathcal{S}$. Axioms of the form $B \sqsubseteq C$ or $t \sqsubseteq s$ in $\mathcal{S}$ are* supporting set axioms *in which B is a supporting concept name in Σ_S, t is a supporting role in Σ_S, C is an $\mathcal{EL}$-concept, and sig(C) $\subseteq$ sig(AX_{Σ_F}). The set of supporting set axioms will be denoted by AX_{Σ_S}.*

Depending on the input ontology, our algorithm generates two types of subontologies: *focus set subontologies*, which may contain weaker abstracted definitions (cf. ab_2 in $\mathcal{O}_2$ in Example 8), and *equivalent focus set subontologies*, which contain abstracted definitions that are equivalent to their original definitions in $\mathcal{O}$ for all concepts in the focus set Σ_F (cf. ab_1 in $\mathcal{O}_1$ in Example 8). We define a subontology that contains equivalent focus set abstracted definitions for all focus concepts in Σ_F as follows.

Definition 21 (Equivalent Focus Set Subontology)**.** *Let $\mathcal{O}$ be an $\mathcal{ELH}$ terminology, and $\mathcal{S}$ a focus set subontology of $\mathcal{O}$ for Σ_F. $\mathcal{S}$ is an* equivalent focus set subontology *that satisfies the following condition for each abstracted definition $\alpha' \in \mathcal{S}$ for a concept name $A \in \Sigma_F$ and its original definition $\alpha \in \mathcal{O}$:*

$$\mathcal{O} \models \alpha' \text{ and } \mathcal{O} \setminus \{\alpha\} \cup \{\alpha'\} \models \alpha.$$

Focus set axioms AX_{Σ_F} in an equivalent focus set subontology satisfy both conditions in Definition 17, because for focus set axioms we have, $\mathcal{O} \models \alpha$ iff $\mathcal{S} \models \alpha$ where $\alpha \in AX_{\Sigma_F}$.

The following theorem establishes when our method can be used to obtain an equivalent focus set subontology.

Theorem 22. *For any acyclic $\mathcal{ELH}$ terminology, and a focus set $\Sigma_F \subseteq sig(\mathcal{O})$, our method returns* equivalent focus set subontologies*, i.e., the condition in Definition 21 is satisfied.*

Proof. Given that the input ontology to our method is an acyclic $\mathcal{ELH}$ terminology, where each concept name A has at most one concept definition, our algorithm can be seen to compute abstracted definitions via the notion of unfolding [BHLS17]. Using unfolding, defined concept names in the right hand side of the definition of A are replaced one by one with their definitions, which comprise a conjunction of primitive concept names and a conjunction of existential restrictions. This implies that our method returns equivalent focus set subontologies. □

The following property shows that our method produces subontologies that preserve the concept hierarchy over the focus and supporting concepts.

Property 1. *Let $\mathcal{O}$ be an $\mathcal{ELH}$ ontology, and suppose $\mathcal{S}$ has been extracted by our method from $\mathcal{O}$ for a focus set Σ_F then $\mathcal{O} \models \mathcal{H}_\mathcal{S}$, where $\mathcal{H}_\mathcal{S}$ is the concept hierarchy of the subontology $\mathcal{S}$.*

Proof. We first show that (i) $\mathcal{O} \models \mathcal{S}$, and then conclude that (ii) $\mathcal{O} \models \mathcal{H}_\mathcal{S}$. (i) From Definition 17, $\mathcal{O} \models \alpha$ if $\mathcal{S} \models \alpha$ for all $\alpha \in \mathcal{S}$. This means that $\mathcal{O} \models \mathcal{S}$. (ii) Since $\mathcal{H}_S$ is the subsumption hierarchy of concept names in $\mathcal{S}$, then $\mathcal{S} \models \mathcal{H}_\mathcal{S}$. By (i) and transitivity of entailment it follows that $\mathcal{O} \models \mathcal{H}_\mathcal{S}$. □

Through Property 1, we meet the second requirement for focus set subontologies, which is to generate a subontology where its concept hierarchy is restricted to the original ontology's transitive closure of concept name subsumption over the signature of the subontology. Notably, this property holds whether $\mathcal{O}$ is an ontology or a terminology.

To allow us quantify the benefits of abstracting focus concept definitions in terms of possibly eliminating superfluous concepts from the resulting extract, we define the interval of concepts between a focus concept name A (in Σ_F) and its closest primitive ancestors $\mathcal{P}_A^\sqcap$ in $\mathcal{O}$.

An *interval* $[A, B]$ in $\text{sig}_C(\mathcal{H})$ is the set of concept names Z such that $\mathcal{O} \models A \sqsubseteq Z \sqsubseteq B$ and contains both A and B. The open interval (A, B) is the set of concepts Z in $\text{sig}_C(\mathcal{H})$ satisfying $\mathcal{O} \models A \sqsubset Z \sqsubset B$. The set X of all concept names that subsume a concept name A in $\text{sig}_C(\mathcal{H})$ is the ancestor set of A, denoted by $\uparrow A$, while the set Y of all concept names that are subsumed by A in $\mathcal{H}$ is the descendant set of A, denoted by $\downarrow A$.

Definition 23 (The $(A, \mathcal{P}_A^\sqcap)$-interval)**.** *Assume $\mathcal{O}$ is an $\mathcal{ELH}$-terminology, $\mathcal{H}$ is the concept hierarchy of $\mathcal{O}$, A is a concept name in the focus set Σ_F in sig$(\mathcal{O})$, and $\mathcal{P}_A^\sqcap$ is the set of closest primitive ancestor(s) of A in $\mathcal{O}$. The open* interval *$(A, \mathcal{P}_A^\sqcap)$ is the set*

of concept names D that occur between A and $\mathcal{P}_A^{\sqcap}$ in $\mathcal{H}$ satisfying $\mathcal{O} \models A \sqsubset D \sqsubset P$ for some primitive concept name $P \in \mathcal{P}_A^{\sqcap}$.

Intuitively, the interval $(A, \mathcal{P}_A^{\sqcap})$ is defined as the set of concepts D shared among A's ancestors ($\uparrow A$) and $\mathcal{P}_A^{\sqcap}$'s descendants ($\downarrow \mathcal{P}_A^{\sqcap}$) in the concept hierarchy $\mathcal{H}$.

We note that all concepts D in $(A, \mathcal{P}_A^{\sqcap})$ are defined concept names since they occur on the right hand sides of definitions of focus concept names. For example, the concept *LiverDisease* in Figure 5.2 is a defined concept name that occurs in the open interval between the focus concept *Hepatitis2* and its closest primitive ancestor *Disease*.

Concepts in the interval set might occur in the subontology if they are supporting concepts for other focus set axioms. The interval set does not include symbols on the right-hand side of the definitions of the interval set's concepts.

We define the open interval between a focus set Σ_F and the closest primitive ancestors $\mathcal{P}_{\Sigma_F}^{\sqcap}$ to the focus concepts in Σ_F as follows.

Definition 24 (The $(\Sigma_F, \mathcal{P}_{\Sigma_F}^{\sqcap})$-interval)**.** *Assume $\mathcal{O}$ is an $\mathcal{EL}\mathcal{H}$-terminology, $\mathcal{H}$ is the concept hierarchy of $\mathcal{O}$ and Σ_F is a focus set. Recall, $\mathcal{P}_A^{\sqcap}$ denotes the conjunction of closest primitive ancestors to the concept name A in $\mathcal{O}$. By $(\Sigma_F, \mathcal{P}_{\Sigma_F}^{\sqcap})$ we denote the set of concept names D such that $D \in (A, \mathcal{P}_A^{\sqcap})$, for some $A \in \Sigma_F$.*

The following property states that an abstracted definition of a concept name A in Σ_F inherits all the information that occurs in the definitions of concepts in the open interval $(A, \mathcal{P}_A^{\sqcap})$.

Property 2. *Let $\mathcal{O}$ be an $\mathcal{EL}\mathcal{H}$ acyclic terminology and $\mathcal{S}$ is an equivalent focus set subontology extracted from $\mathcal{O}$ for focus concept names in Σ_F, then, a focus concept name $A \in \Sigma_F$ inherits all of the necessary (and sufficient) conditions of definitions of concepts in $(A, \mathcal{P}_A^{\sqcap})$.*

Proof. From Definition 23, the focus concept A is connected upwardly to concepts in the interval set $(A, \mathcal{P}_A^{\sqcap})$ via a chain of concept inclusions until it reaches the closest primitive ancestor concepts $\mathcal{P}_A^{\sqcap}$ in the concept hierarchy $\mathcal{H}$. Via transitivity of the subsumption relation, A inherits all of the necessary (and sufficient) conditions of concepts in the open interval between A and its closest primitive ancestors $\mathcal{P}_A^{\sqcap}$ $(A, \mathcal{P}_A^{\sqcap})$. □

This property holds when the input ontology is an acyclic terminology as there is at most one definition for each concept name in $\mathcal{O}$. Thus, any defined concept name D

in $(A, \mathcal{P}_A^{\sqcap})$ does not carry information other than what A inherits through definition expansion. The definitions of concepts found in the interval $(A, \mathcal{P}_A^{\sqcap})$ are therefore redundant. Therefore, incorporating the abstracted definitions of focus concepts is sufficient to capture the complete semantics of focus concepts.

As already said earlier, some concepts in the interval may belong to the subontology's signature as supporting concepts for the definitions (or descriptions) of focus concepts.

5.3 Verifying Subontologies

In this section, we outline two techniques to verify that the computed subontologies comply with the requirements stated in Section 5.2.1. Although we demonstrated theoretically that the second requirement of the focus set subontologies should be met according to our definition and algorithm (see Property 1), we wanted to have this empirically verified in order to validate the implementation. As stated in Property 1, this property holds even if the input ontology is not a terminology.

Verification of the first requirement. The first requirement of our method states that the definitions of focus set concepts in the subontology must be equivalent to those in the original ontology. Theorem 22 tells us this is true when the input ontology to our method is a terminology. However, if the input ontology is not a terminology, then the computed focus set abstracted definitions can be weaker than the original definitions. Weaker abstracted definitions occur when the input focus concept A has more than one axiom in the input ontology, where the computation of the abstracted definition for A in $\mathcal{O}$ condenses all of A's axioms into a single definition (see our earlier Example 8).

Recall, the definition of UI-witness of an ontology w.r.t another ontology in Definition 12 in Section 3.2.1. When the input ontology to our method is not a terminology, then we can verify whether an abstracted definition $\alpha' \in \mathcal{S}$ for a focus concept A is equivalent to its original definition $\alpha \in \mathcal{O}$ if the UI-based semantic difference between $\mathcal{O}$ and $\mathcal{S}$ for $\{A\}$ is empty.

Moreover, this verification allows us measure the degree of incompleteness of the focus set definitions in a subontology by counting how many focus set axioms in the original ontology are not entailed by the computed subontology.

We define the set of focus set witnesses between a subontology $\mathcal{S}$ and an ontology $\mathcal{O}$ as follows.

Definition 25 (Focus Concept Witness in the Original Ontology). *Let $\mathcal{O}$ be an ontology and $\mathcal{S}$ a subontology extracted from $\mathcal{O}$ for a focus set Σ_F. Suppose A is a focus concept name in Σ_F and α is a UI-witness of a difference in $\mathcal{O}$ w.r.t $\mathcal{S}$ (i.e., $\alpha \in$ UI-Diff$(\mathcal{S}, \mathcal{O})$). If α contains A, then α is* focus concept witness in $\mathcal{O}$.

All such focus concept witnesses will be denoted by UI-Diff$_{\Sigma_F}(\mathcal{S}, \mathcal{O})$.

Note that all such witnesses are axioms of the form $C \sqsubseteq A$ representing the downward direction of the definition of the focus concept A. There are no focus concept witnesses of the form $A \sqsubseteq C$ in the original ontology, as the subontology generation method (Algorithm 10) captures all the upward conditions associated with an input focus concept A in the original ontology $\mathcal{O}$.

We use the UI-based semantic difference in our evaluation of the focus set definitions in the focus set subontologies generated from the Gene ontology (see Section 5.5.5).

Verification of the second requirement. The second requirement of our method states that the concept hierarchy of a computed subontology is a restriction of the concept hierarchy of the original ontology over the signature of the subontology. Algorithm 14 is designed to verify this requirement.

The algorithm starts by initialising three sets: $\mathcal{H}_\mathcal{S}$ denoting the concept hierarchy of the subontology $\mathcal{S}$, $TC_\mathcal{O}$ denoting the transitive closure of the original ontology $\mathcal{O}$ restricted to the signature of the subontology, and $TC_\mathcal{S}$ denoting the transitive closure of the subontology $\mathcal{S}$. Lines 2 and 3 of the algorithm classify the subontology $\mathcal{S}$ and the original ontology $\mathcal{O}$ by calling the ELK reasoner to obtain $\mathcal{H}_\mathcal{S}$ and $\mathcal{H}_\mathcal{O}$, respectively. Line 4 computes the set $\text{sig}_C(\mathcal{H}_\mathcal{S})$ of concept names that occur in $\mathcal{H}_\mathcal{S}$.

Lines 6–10 detail the steps of computing the transitive closure of the original ontology that is constrained to the signature of the subontology, while Lines 11–14 computes the transitive closure of the subontology.

The loop in Lines 5–10 iterates through every concept name A in $\text{sig}_C(\mathcal{H}_\mathcal{S})$. In Line 6, The function ComputeAncestorsOfA computes the ancestors of the concept A from the hierarchy of the original ontology $\mathcal{H}_\mathcal{O}$ to get the set $\uparrow A_\mathcal{O}$.

From Line 7 to Line 10 an iteration over the concepts in $\uparrow A_\mathcal{O}$ begins. This is done in order to compute axioms α of the form $A \sqsubseteq \text{Ancestor}_\mathcal{O}$, where A is the current concept in the iteration of concepts in $\text{sig}_C(\mathcal{H}_\mathcal{S})$ and $\text{Ancestor}_\mathcal{O}$ is the current concept in the iteration of concepts in $\uparrow A_\mathcal{O}$. In Line 8, we check if the signature of the axiom α is in $\text{sig}_C(\mathcal{H}_\mathcal{S})$. If this is the case, then the axiom α is added to the set $TC_\mathcal{O}$ (Line 10).

Algorithm 14 IsConceptHierarchyVerified($\mathcal{S}$, $\mathcal{O}$)

Input: Original ontology $\mathcal{O}$, Subontology $\mathcal{S}$
Output: IsVerified

1: $\mathcal{H}_\mathcal{S} := \emptyset, TC_\mathcal{O} := \emptyset, TC_\mathcal{S} := \emptyset.$
2: $\mathcal{H}_\mathcal{S} :=$ Classify($\mathcal{S}$)
3: $\mathcal{H}_\mathcal{O} :=$ Classify($\mathcal{O}$)
4: $\text{sig}_C(\mathcal{H}_\mathcal{S}) :=$ GetTheConceptSignature($\mathcal{H}_\mathcal{S}$)
5: **for** $A \in \text{sig}_C(\mathcal{H}_\mathcal{S})$ **do**
6: $\quad\uparrow A_\mathcal{O} :=$ ComputeAncestorsOf$A(A, \mathcal{H}_\mathcal{O})$
7: $\quad$**for** Ancestor$_\mathcal{O} \in\uparrow A_\mathcal{O}$ **do**
8: $\quad\quad$Compute $\alpha : A \sqsubseteq$ Ancestor$_\mathcal{O}$
9: $\quad\quad$**if** $\text{sig}_C(\alpha) \in \text{sig}_C(\mathcal{H}_\mathcal{S})$ **then**
10: $\quad\quad\quad TC_\mathcal{O} := TC_\mathcal{O} \cup \{\alpha\}$
11: $\quad\uparrow A_\mathcal{S}$ = ComputeAncestorsOf$A(A,\mathcal{H}_\mathcal{S})$
12: $\quad$**for** Ancestor$_\mathcal{S} \in\uparrow A_\mathcal{S}$ **do**
13: $\quad\quad$Compute $\beta : A \sqsubseteq$ Ancestor$_\mathcal{S}$
14: $\quad\quad TC_\mathcal{S} := TC_\mathcal{S} \cup \{\beta\}$
15: **if** $TC_\mathcal{O} = TC_\mathcal{S}$ **then**
16: $\quad$return IsVerified := TRUE

After the iteration through the concepts in $\uparrow A_\mathcal{O}$ ends, the set $TC_\mathcal{O}$ is computed.

In Line 11, we compute the ancestors set of A using the subontology hierarchy $\mathcal{H}_\mathcal{S}$ to obtain $\uparrow A_\mathcal{S}$. Then, we go through every ancestor concept in $\uparrow A_\mathcal{S}$ to compute the axiom β: $A \sqsubseteq$ Ancestor$_\mathcal{S}$ to add it to the set of the transitive closure of the subontology $TC_\mathcal{S}$. Following the completion of the iteration over the concepts in $\uparrow A_\mathcal{S}$, the set $TC_\mathcal{S}$ is computed.

Lastly, we check if both sets $TC_\mathcal{O}$ and $TC_\mathcal{S}$ are equal. If this is the case, then the algorithm returns TRUE.

For subontologies computed in our evaluation (Section 5.5), Algorithm 14 was used to test all generated subontologies. All passed, i.e., the algorithm returned TRUE for the variable IsVerified.

This means that the generated subontologies satisfy the method's second requirement, which states that the concept hierarchy of the subontology is a restriction of the concept hierarchy of the original ontology over the signature of the subontology.

5.4 Related Ontology Extraction Methods

This section summarises the key difference points between our method and some of the related methods discussed in Chapter 3, namely the SLBM and the uniform interpolation methods.

Table 5.1 presents a summary of the comparison.

Table 5.1: A summary of the key points about the three different methods

Aspect	**Focus set subontologies**	**Syntactic locality-based modules**	**Uniform interpolants**
Solved problem	Extracts an ontology to define input focus symbols in Σ_F	Returns all axioms relevant to the meaning of a given set of concepts and roles	Extracts an ontology expressed over the set of input symbols
Axiom structure	Abstracted definitions and concept inclusions	Subset of original ontology $\mathcal{O}$	Rewritten axioms and might contain definer names
Extract size	Limited by number of focus set AX_{Σ_F} and supporting set AX_{Σ_S} axioms	Modules can be very large, maximally as large as $\mathcal{O}$	Can be triple exponential in the size of $\mathcal{O}$
Computation time	Polynomial in the size of the input ontology and focus set	Polynomial in the size of the input ontology [VKP+13]	2ExpTime-complete for $\mathcal{ALC}$ [LW11] and ExpTime-complete for $\mathcal{EL}$ ontologies [LSW12]

Table 5.1 summarises the difference points between our method to generate focus set subontologies, SLBM and uniform interpolation. We make comparisons between the three module notions on the basis of four distinct aspects. The first aspect clarifies what problem the method is attempting to solve by examining the type of extract generated by an ontology extraction method. The second aspect establishes whether the ontology extraction notion preserves the input ontology's axiom structure. The third aspect is about comparing the size of the extract, while the fourth aspect compares the computation time required to construct an extract.

In relation to the **solved problem** using the three methods, our subontology generation method computes subontologies that define the set of input terms. It is similar to the notion of SLBM, in which the input symbols serve as seed symbols for extracting

definitions, except that SLBM additionally generates more axioms. These additional axioms completely capture the meaning of a given set of terms; that is, any axioms necessary for the meaning of these terms are included in the final module. This is an overly strict condition, which may result in extremely large modules. On the other hand, uniform interpolation generates UIs defined by the set of input symbols. In other words, UIs are highly dependent on the symbols of interest and their relationships within the input ontology. If such relationships between input symbols are missing, the resulting UIs may be too small to incorporate the definitions of input symbols or, in some cases, may be empty.

With respect to preserving the input ontology's **axiom structure**, our subontology generation method provides abstracted definitions and subsumption axioms that complete the subontology's hierarchical relationships. These abstracted definitions are in a normal form that is recommended by SNOMED International because of the many benefits it provides when modelling concept definitions or when the ontology is used in query-related applications. Therefore, subontologies preserve the original structure of focus set axioms, while incorporating supporting set axioms via subsumption relationships that are restricted to the signature of focus set axioms. In the case of SLBM, because the generated modules are subsets of the original ontology, the axioms in the modules preserve their original structure. On the other hand, uniform interpolation rewrites the ontology to generate axioms specified by the set of input symbols.

The **size** of extracts generated by our subontology generation method is limited by the axioms of the focus set and the supporting set, resulting in an extract that completely captures the semantics of focus symbols while being concise. On the other hand, when employed with very large ontologies, SLBM can create extremely large modules, and can reach up to the size of the input ontology $\mathcal{O}$. Uniform interpolation generates UIs that are constrained by the set of input symbols; as a result, UIs are usually quite small. Even though it has been demonstrated theoretically that the size of given uniform interpolants can be exponentially three times larger than the size of the input ontology [NR14, LW11], the results of our evaluations of uniform interpolation using real-world signatures in Chapter 4, did not exhibit the worst case behaviour of this theoretical finding.

In terms of the **computation time**, our subontology generation method runs in polynomial time in the size of the input ontology and focus set. This is also true for the efficient SLBM. On the other hand, deciding the existence of uniform interpolants is 2ExpTime-complete for $\mathcal{ALC}$ TBoxes [LW11]. Deciding the existence of UIs of $\mathcal{EL}$

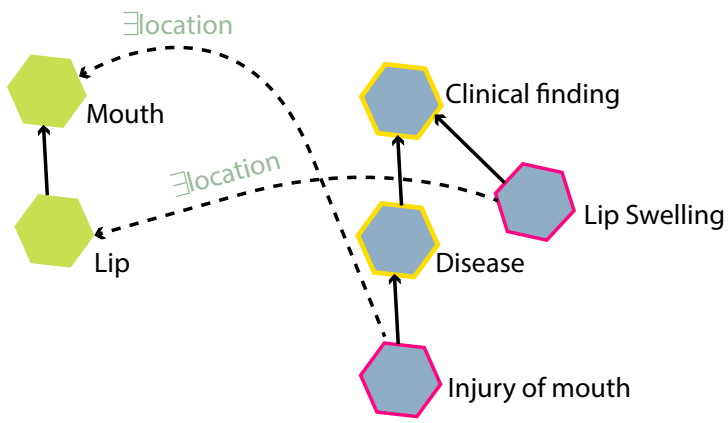

Figure 5.4: The result of extracting a subontology using our method

ontologies, is ExpTime complete [LSW12].

Example

We present an example about ontology extraction from SNOMED CT to illustrate the difference in the outcomes by the different extraction methods: uniform interpolation, SLBM and our focus set subontology extraction method.

We assume that a user is interested in extracting an ontology about two focus concept names from SNOMED CT: ***Injury of mouth***, and ***Lip swelling***.

If the subontology extraction method is used, then it results in the subontology illustrated in Figure 5.4. The axioms in the resulting subontology are:

$$\textbf{\textit{Injury of mouth}} \equiv \mathit{Disease} \sqcap \exists \mathit{location}.\mathit{Mouth} \tag{5.10}$$

$$\textbf{\textit{Lip swelling}} \equiv \mathit{Clinical\ finding} \sqcap \exists \mathit{location}.\mathit{Lip} \tag{5.11}$$

$$\mathit{Lip} \sqsubseteq \mathit{Mouth} \tag{5.12}$$

Both of the focus concepts are defined in terms of the closest primitive ancestors which are *Clinical finding* and *Disease*, resulting in complete definitions of the focus concepts (axioms 5.10 and 5.11). This abstraction helps in reducing the size of the resulting subontology by 32 concept definitions in the interval set $(A, \mathcal{P}_A^{\sqcap})$ (see Definition 24 about the interval set $(A, \mathcal{P}_A^{\sqcap})$). Additionally, the subontology derives axioms for the supporting concepts *Lip* and *Mouth*, indicating a relationship of subsumption between the two concepts (axiom 5.12), where such concepts existed in the signature of focus set axioms; the abstracted definitions of ***Injury of mouth*** and ***Lip swelling***.

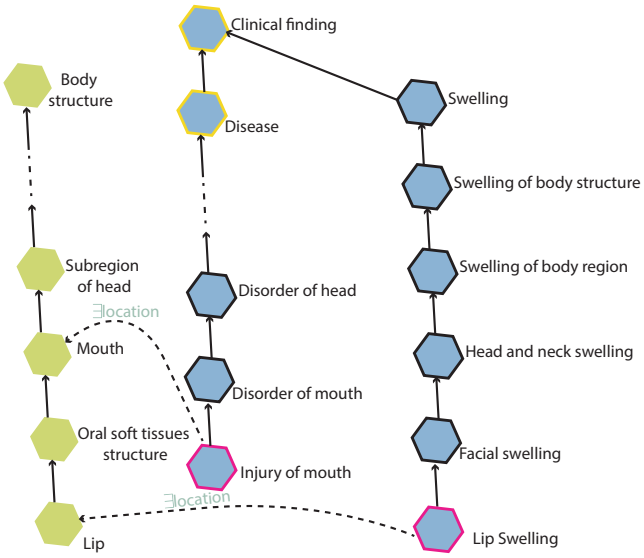

Figure 5.5: The result of extracting a bottom module using the SLBM method, the $\bot$-type

Extracting a $\bot$-module for the focus set $\{$***Injury of mouth***, ***Lip swelling***$\}$ results in the module depicted in Figure 5.5. In Figure 5.5, we show only part of the resulting module because it is too large to illustrate graphically. The resulting $\bot$-module consists of all of the relations and concepts that ***Injury of mouth*** and ***Lip swelling*** are *upwardly* connected to, which totals 83 logical axioms.

To extract a uniform interpolant from SNOMED CT for the concepts ***Injury of mouth*** and ***Lip swelling***, the uniform interpolation method performs forgetting of concepts not included in the input signature. This results in an empty extract as a result of forgetting the symbols in the right-hand sides of the definitions of ***Injury of mouth*** and ***Lip swelling***.

5.5 Evaluation

We evaluate our method for computing focus set subontologies in this section. To begin with, we describe the main components of the implementation of our subontology generation method. Second, we outline the evaluation tools that were utilised in the experiments. Thirdly, we describe the corpus used in the experiments. Fourth, we discuss the different experiments that we carried out. Fifth, we discuss the results of our experiments in detail and analyse them using proposed quality measures including, the size, precision rate, interval values and the semantic difference between the generated

extracts.

5.5.1 Implementation

We implemented our method in Java using the OWL API.[1] The main components of our subontology generation prototype is illustrated in Figure 5.6, where the arrows indicate the direction of information flow between the system's components.

Due to the ELK reasoner's high efficiency in reasoning, we make extensive use of its reasoning capabilities in our implementation. Two primary services are utilised from ELK: ontology classification and computation of a concept's ancestors.

Another component of the system is the pre-processor, which adds definers for every existential restriction in the input ontology as described in Section 5.2.2, and eliminates them when needed.

The subontology extractor component contains three main methods, including the abstracted definitions generator, the concept inclusions generator and the role inclusions generator. This component makes use of the ELK reasoning services component to generate the abstracted definitions and the concept and role inclusions axioms.

The post-processor component contains two methods, the first one (RemoveTransitiveClosureAxioms) removes unnecessary axioms that are generated during the derivation of abstracted definitions, concept inclusion axioms, and role inclusion axioms. The second method (AddAnnotationAxioms) adds labels to the resulting subontology to make it readable, and add labels to distinguish between focus and supporting symbols.

The subontology verifier component contains two methods. The first one makes use of the UI-based semantic difference tool in order to evaluate the completeness of the focus set definitions in case the input ontology is an ontology rather than a terminology. The UI-based semantic difference tool makes use of the $\mathcal{ELH}$ forgetter [LLA+21a] and the HermiT reasoner [GHM+14].

The second method is the concept hierarchy verifier, which implements Algorithm 14. It verifies the resulting subontology concept hierarchy as being a restriction of the original ontology's concept hierarchy.

All of the system's components make use of the OWL API [HB09]. The OWL API provides services for loading, manipulating, and saving of ontologies. Moreover, the

[1] `http://owlapi.sourceforge.net/`

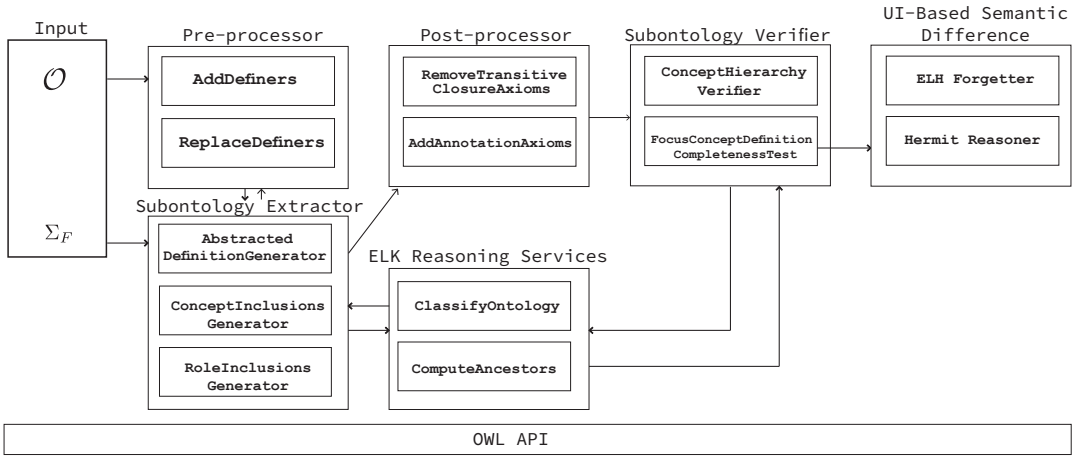

Figure 5.6: The subontology generation prototype architecture

ELK reasonser [KKS14] and the UI-based semantic difference tool [LLA+21a] rely on the OWL API in our implementation.

5.5.2 Used Tools

Part of our evaluation is to compare our results to the extraction methods we outlined in Section 5.4, which are the $\bot$-type of SLBM and the uniform interpolation methods. We make use of the following three systems:

1. The subontology generation prototype described in the previous section.[2]

2. The uniform interpolation tool for $\mathcal{ELH}$ ontologies of the group of Yizheng Zhao [LLA+21a].

3. The OWL API's built-in SLBM method, $\bot$-type.

Given how the three methods work (see Section 5.4 for a comparison between the three methods), we used the following input sets to generate the different extracts:

1. The focus set was used as the input signature for computing the bottom modules and subontologies, because both SLBM and our subontology generation method extend the signature as needed.

[2]The system can be downloaded from: `https://tinyurl.com/subontologies-sys`

2. As input to the UI method, we used the signatures of the computed subontologies. This is because UIs for focus sets would not adequately capture their definitions.

5.5.3 Data Used

SNOMED CT

SNOMED CT (version July 2017) was used in this study. This version and earlier versions of SNOMED CT are terminologies that do not contain GCI axioms of the form $C \sqsubseteq A$ or $C \sqsubseteq D$, where A is a concept name and C and D are $\mathcal{EL}$-concepts. Beginning with the January 2018 release, GCI axioms, transitive and reflexive role axioms were introduced and incorporated SNOMED CT, which our method does not support.

SNOMED CT (version July 2017) had a total of 335 245 logical axioms, 335 225 concept names and 97 role names. The ontology is in the scope of $\mathcal{ELH}$ with the exception of one role chain axiom ($r \circ s \sqsubseteq r$). To be employed in the experiments, these types of axioms and $\mathcal{ELH}$-axioms in other ontologies were omitted.

As focus sets, we used 11 focus sets of medical conditions for computing SNOMED CT extracts used in the experiments of [ACSDD+19] (see Section 2.4.2 about refsets). Such medical conditions that affect humans and animals include *heart failure*, *asthma*, *epilepsy*, *glaucoma*, *chronic kidney disease*, *osteoarthritis*, *anaemia*, *arthritis*, *diabetes*, *hypertension*, and *obesity*. The supplementary information of these experiments includes a list of 19–39 concept names for each medical condition.[3]

We computed subontologies for each medical condition focus set. Due to the small size of the input symbols, we were unable to investigate certain quality aspects of subontologies that we considered in our evaluation, such as the effect of abstracted definitions. As a result, we computed descendant concepts of these input concepts in order to increase the size of the extracted data. The extended focus sets consisted of 491–74 291 concept names for each medical condition. We refer to such extended focus sets as the *large* focus sets. The number of concepts in each medical condition's focus set before and after extension is depicted in Table 5.2.

During our tests, we discovered that the uniform interpolation tool was incapable of generating views for the medical conditions *large* focus sets. We traced the cause of this problem to the vast number of symbols that must be forgotten from the input

[3] `https://tinyurl.com/medical-conditions-signature`

Table 5.2: Number of concept names in the SNOMED CT medical condition focus sets before and after downward extension

Focus set	Concepts (before extension)	Concepts (after extension)
Heart Failure	37	4522
Asthma	37	7934
Epilepsy	19	491
Glaucoma	39	8743
CKD	19	2733
Osteoarthritis	38	8990
Anaemia	34	4401
Arthritis	33	7805
Diabetes	29	74291
Hypertension	31	14828
Obesity	28	9885

ontology. Another reason is the complex nature of SNOMED CT where forgetting becomes very difficult. (See Chapter 4 (Section 4.4) for detailed study concerning forgetting from SNOMED CT.)

To address this issue, we created UIs employing the *small* focus sets. We tested computing uniform interpolants for the signature of the subontologies extracted using the set of medical conditions before extending them as mentioned above. Running the uniform interpolation method on these medical conditions generated UIs for only two of them, namely *epilepsy* and *obesity*.

We also ran the uniform interpolation method for the signature of the subontologies extracted for focus sets listed in [oPb]. Such focus sets are provided by the Chartered Society of Physiotherapy (CSP) in the UK that are used in their electronic health records. Out of these focus sets we were able to generate UIs for *Pain aggravating factors*, *Pain easing factors*, *Posture findings*, and *Transfer ability findings*, which consisted of 7–70 concept names.

Gene Ontology

We used the Gene ontology (GO) (version February 2021). GO does not meet the requirements of a terminology, as it may contain more than one axiom for a concept name A. GO does not include GCI axioms of the forms $C \sqsubseteq A$ and $C \sqsubseteq D$ where C and D are $\mathcal{EL}$-concepts.

The Gene ontology had a total of 102 203 logical axioms, 44 085 concept names and 8 role names. The concept number excludes 6 430 deprecated concepts that exist

in the version 01-02-2021 and 8 role names. GO contains one inverse role, 29 disjoint concepts, four transitive roles, and two role chain axioms. These axioms were omitted from GO to carryout the experiments.

We used 12 focus sets of the GO slim sets provided in [The]. Each set represented a flat list of concept and role names that are specific to certain species or organisms.

All experimental data used in our experiments is available at `https://tinyurl.com/evaluation-data`.

5.5.4 Conducted Experiments

The experiments were conducted on machines equipped with Intel Xeon CPU E5-2640 v3 running at 2.60GHz with 32GB of RAM. Different experiments were performed using the aforementioned tools and data. We outline and discuss the settings of each experiment as follows.

1. **Evaluating focus set definitions.** (See Section 5.5.5.)
   We performed two tests to evaluate the generated focus set abstracted definitions. The first test applies the first verification method described in Section 5.3. The method assesses the completeness of the focus set definitions using the notion of UI-based semantic difference.

   The second was done by performing the test of Definition 16, which assesses the logical strength of the generated abstracted definitions as to whether they are weaker or equivalent to the original definitions. To perform this test, we implemented Algorithm 15. The algorithm is intended to demonstrate empirically that the generated abstracted definitions can be weaker than the original definitions when the input ontology is not a terminology and contains more than one definition for a concept name A.

   Algorithm 15 takes as input an ontology $\mathcal{O}$ and a focus set Σ_F. It begins by initialising an ontology $\mathcal{O}'$ to $\mathcal{O}$, and two sets, one for the original definitions of the input focus concepts denoted by A_{Original} and another for the abstracted definitions of the input focus concepts denoted by $A_{\mathsf{Abstracted}}$. Lines 2 and 3 compute the ontology with definer names $\mathcal{O}^d$ and its concept hierarchy $\mathcal{H}$, respectively, which are used in Line 6 to compute an abstracted definition of a focus concept A.

   The process of substituting an abstracted definition α' for its original definition α in the ontology $\mathcal{O}'$ is done in Lines 5–10.

Algorithm 15 IsWeakerAbstractedDefinition($\mathcal{S}$, $\mathcal{O}$)

Input: Original ontology $\mathcal{O}$, Focus set Σ_F

Output: Count number of weaker abstracted definitions $\mathsf{Counter}_{\mathsf{Weaker}}$.

1: $\mathcal{O}' := \mathcal{O}$, $A_{\mathsf{Original}} := \emptyset$, $A_{\mathsf{Abstracted}} := \emptyset$, $\mathsf{Counter}_{\mathsf{Weaker}} := 0$.
2: $\mathcal{O}^d :=$ AddDefiners($\mathcal{O}$)
3: $\mathcal{H} :=$ Classify($\mathcal{O}^d$)
4: **for** Focus concept $A \in \Sigma_F$ **do**
5: $\quad\alpha$ = Get original definition of A in $\mathcal{O}$
6: $\quad A_{\mathsf{Original}} := A_{\mathsf{Original}} \cup \{\alpha\}$
7: $\quad\alpha'$ = AbstractedDefinitionExtraction(A, $\mathcal{O}^d$, $\mathcal{H}$, TRUE)
8: $\quad A_{\mathsf{Abstracted}} := A_{\mathsf{Abstracted}} \cup \{\alpha'\}$
9: $\quad\mathcal{O}' := \mathcal{O}' \backslash \{\alpha\}$
10: $\quad\mathcal{O}' := \mathcal{O}' \cup \{\alpha'\}$
11: **for** $\alpha \in A_{\mathsf{Original}}$ **do**
12: $\quad$**if** $\mathcal{O}' \not\models \alpha$ **then**
13: $\quad\quad\mathsf{Counter}_{\mathsf{Weaker}} := \mathsf{Counter}_{\mathsf{Weaker}} + 1$
14: Return $\mathsf{Counter}_{\mathsf{Weaker}}$

Lines 11–13 check for possible weaker abstracted definitions by checking whether the ontology $\mathcal{O}'$ entails the original definitions in the set A_{Original}. If an original definition is not entailed by $\mathcal{O}'$, then its corresponding abstracted definition in $\mathcal{O}'$ is weaker than the original definition, and the counter $\mathsf{Counter}_{\mathsf{Weaker}}$ is increased by one.

Our implementation checks whether $\mathcal{O}' \models \alpha$ where α is an original definition of a focus concept and $\mathcal{O}'$ is the ontology after exchanging all of the focus concepts' original definitions by their abstracted definitions. Our test is different from what Definition 16 states. In Definition 16, the entailment test $\mathcal{O}' \models \alpha$ is applied to single definition exchange rather than multiple definitions exchanges as our test does. We implemented the test in this manner (Lines 11–13) for efficiency considerations, in order to avoid initialising the reasoner for the ontology $\mathcal{O}'$ once each focus concept definition exchange is completed (Lines 9 and 10). Even when implemented in this manner, Algorithm 15 should confirm that a terminology would not contain weaker abstracted definitions.

2. **Comparing the size in different extracts.** (See Section 5.5.6.)
We computed the size of the generated extracts. The size represents the number of logical axioms, concepts and roles. We used this metric to compare the extract types in terms of their size. The smaller the extract type for the same set of input

symbols, the more concise it is.

When we measured the size of the resulting UIs, we found that they are very large. Our inspection revealed that this is due to the existence of large number of redundant axioms. Moreover, such axioms were in normalised form. Therefore, we post-processed the resulting UIs using two distinct methods to ensure that they are as close to our results as possible. The first method is to denormalise the UIs according to two denormalisation rules:

(a) $A \equiv C$ iff $A \sqsubseteq C$ and $C \sqsubseteq A$.

(b) $A \sqsubseteq D_i \sqcap .. \sqcap D_n$ iff $A \sqsubseteq D_i$ and .. and $A \sqsubseteq D_n$.

The second method is to eliminate the redundant axioms by applying the rule: if $\mathcal{O} \backslash \{\alpha\} \models \alpha$ then α is redundant.

Table 5.3 shows the number of logical axioms in the Gene ontology generated UIs prior to and following post-processing. As can be observed from the table, a UI can have a significant number of superfluous axioms, which can be deleted using the post-processing methods described above. For instance, the *goslim_chembl* UI began with 4855 and ended with fewer than 500 axioms after post-processing.

Table 5.3: Number of logical axioms in the Gene ontology focus sets before and after post-processing

Focus set	Logical Axioms (before)	Logical Axioms (after)
goslim_agr	141	86
goslim_aspergillus	794	158
goslim_candida	612	160
goslim_chembl	4855	467
goslim_drosophila	2891	266
goslim_generic	1014	245
goslim_metagenomics	798	216
goslim_mouse	127	79
goslim_pir	3719	639
goslim_plant	543	155
goslim_pombe	392	140
goslim_yeast	1393	284

3. **Evaluating the effect of abstracted definitions.** (See Section 5.5.7.)
Given that the $\bot$-type of SLBM computes modules that describe the upward

relationships of the input symbols, we consider the computed subontologies to be within the computed bottom modules, as subontologies also represent the upward relationships of the input focus symbols. Such upward relationships in subontologies are described by the abstracted definitions of focus concepts.

On this basis, we examined the impact of computing abstracted definitions on reducing the size of subontologies. Two metrics were used to quantify the effect, including the interval set, and the precision rate of bottom modules, which are defined next.

(a) The interval set $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})$ (see Definition 24), which is divided into two sets:

   i. $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \backslash \text{sig}(\mathcal{S})$: This set includes concepts in the interval set that are outside the signature of the subontology. We computed such concepts to identify the number of symbols that were omitted by the abstraction in the algorithm for subontology generation.

   ii. $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \cap \text{sig}(\mathcal{S})$: This set includes concepts in the interval set that occur in the signature of the computed subontologies as supporting concepts for focus set definitions.

   It is worth noting that this metric is applicable when the input ontology is a terminology. We used the measure $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \backslash \text{sig}(\mathcal{S})$ to assess the resulting bottom modules in terms of carrying symbols that can be ignored as a result of abstraction. The lower the number of concepts in the interval set in a bottom module, the fewer concepts exist in the computed extracts that could potentially carry the same concepts that the focus concepts inherit.

(b) Precision rate: We measured the ratio of relevant symbols that occurred in the signature of a subontology over the number of symbols in the bottom module, which is defined by the following formula also used in our evaluation in Section 4.4.

$$\begin{aligned}&\mathsf{Precision}(\bot\text{-module}, \mathcal{S}) := \\ &|(\text{sig}(\mathcal{S}) \cap \text{sig}(\bot\text{-module})) \cup \{\top\}| / |\text{sig}(\bot\text{-module})|.\end{aligned}$$

   We regard a $\bot$-module to be precise if it contains no symbols outside the subontology's signature. This is because subontologies consist of the focus set definitions and supporting set inclusions, which represent the necessary

(and sufficient) information about the input focus set that a user is interested in. The greater the value of Precision measure, the higher the quality of a $\bot$- module.

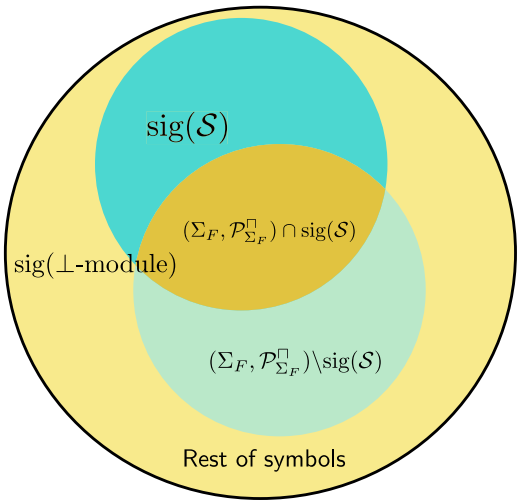

Figure 5.7: Types of symbols in the signature of a $\bot$-module

Figure 5.7 depicts the types of symbols we defined in a $\bot$-module. These include the signature of a corresponding subontology $\text{sig}(\mathcal{S})$, the interval sets $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \backslash \text{sig}(\mathcal{S})$, $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \cap \text{sig}(\mathcal{S})$ and the Rest of symbols set. The Rest of symbols includes symbols that are not part of the interval symbols $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \backslash \text{sig}(\mathcal{S})$ nor the signature of the subontologies. It is defined as:

$$\text{Rest of Symbols} := \text{sig}(\bot\text{-module}) \backslash (\text{sig}(\mathcal{S}) \cup ((\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \backslash \text{sig}(\mathcal{S})))$$

4. **Examining the semantic difference between the extracts.** (See Section 5.5.8.) Using the semantic difference measure, we can see if there are any axioms that can be captured by bottom modules and UIs but not by subontologies within the range of common signature between the subontologies and bottom modules (or UIs). The smaller the set of such differences, the higher the quality of the generated subontologies.

   We employed the notion of UI-based semantic difference, which involves computing a UI for the ontology against which we wish to compare the differences using the shared signature between the two ontologies (for a definition of uniform interpolation see Definition 10, and for a definition of UI-based semantic

difference see Definition 12). Then, using the Hermit reasoner [GHM+14], witnesses are detected, which are the axioms in the second ontology that are not entailed by the UI of the first ontology (or vice versa).

Through these experiments, we aim to test empirically the following hypotheses about our method, including:

1. The abstracted definitions in the generated subontologies are equivalent to their original definitions when the input ontology is a terminology. This is shown by evaluating the equivalence of focus set definitions and testing the logical strength of the generated abstracted definitions (Section 5.5.5).

2. Subontologies are smaller in size than bottom modules, and smaller or equal in size than UIs (Section 5.5.6).

3. Abstracted definitions aid in the reduction of axioms deemed redundant in relation to the input focus set. To illustrate this, we computed the interval set $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})\backslash\text{sig}(\mathcal{S})$ within the bottom modules (Section 5.5.7).

4. According to our definition of Precision (see our description about the third experiment above) the precision values in the bottom modules demonstrate the beneficial effect of computing an extract based on the notion of abstracted definitions, which significantly reduces the size of the extract (Section 5.5.7).

5. The semantic difference between a subontology and a bottom module or a UI might not be empty given how the SLBM $\bot$-type and uniform interpolation methods work (Section 5.5.8).

Using our method, the average time required to compute subontologies for medical conditions focus sets in SNOMED CT is 16.81 minutes. As shown in Table 5.4, as the number of input focus symbols rises, the computation time with our method increases as well. For example, the shortest computation time is 25 seconds for the *epilepsy* subontology, given the relatively small number of input focus concepts (491 concept names). Note that the time required to load the input ontologies is included in our computation time measurement, as this is a natural process when utilising an ontology-based tool.

Table 5.4: Computation times of each focus set subontology in SNOMED CT

Focus set	$\lvert\Sigma_F\rvert$	Computation time in minutes
Heart Failure	4522	1.39
Asthma	7934	3.73
Epilepsy	491	0.42
Glaucoma	8743	2.86
CKD	2733	0.91
Osteoarthritis	8990	3.36
Anaemia	4401	1.53
Arthritis	7805	2.75
Diabetes	74291	158.16
Hypertension	14828	7.32
Obesity	9885	2.51

5.5.5 Evaluating Focus Set Definitions

Completeness of focus set definitions

Given that we use in our evaluation the Gene ontology that does not enjoy the properties of terminologies, i.e., can have more than one definition for a concept name A, we verify the completeness of the focus set definitions in the subontologies using the first verification method (see Section 5.3). As mentioned, the method makes use of the UI-based semantic difference tool.

We computed $\mathsf{UI\text{-}Diff}_{\Sigma_F}(\mathcal{S}, \mathcal{O})$ (Definition 25) for each focus set subontology of the Gene ontology. The results in Table 5.5 demonstrate that, except for the subontology of the focus set *goslim_plant*, the focus set witnesses in $\mathsf{UI\text{-}Diff}_{\Sigma_F}(\mathcal{S}, \mathcal{O})$ were not empty for all comparisons between the original ontology and the gene slim focus sets' subontologies. Such focus set witnesses represent sufficient conditions of focus concepts that were not captured by our subontology generation method. The reason for this is the fact the original ontology contained more than one definition for a concept name. To illustrate our point, consider the following example from our inspection of the results:

Table 5.5: Results of logical strength test and UI-Diff$(\mathcal{S}, \mathcal{O})$ of the abstracted definitions of in the gene slim focus sets in GENE ONTOLOGY.

Focus set Σ_F	UI-Diff$_{\Sigma_F}(\mathcal{S}, \mathcal{O})$		# Equivalent Σ_F defs	# Weaker Σ_F defs	Total Σ_F
	$A \sqsubseteq C$	$C \sqsubseteq A$			
goslim_agr	0	1	52	1	53
goslim_aspergillus	0	8	80	2	82
goslim_candida	0	6	85	1	86
goslim_chembl	0	15	302	2	304
goslim_drosophila	0	6	148	2	150
goslim_generic	0	7	143	2	145
goslim_metagenomics	0	5	114	0	114
goslim_mouse	0	2	42	2	44
goslim_pir	0	15	452	1	453
goslim_plant	0	0	96	0	96
goslim_pombe	0	13	56	2	58
goslim_yeast	0	33	160	3	163

Example 9. *Let $\mathcal{O}$ =*

$$\{A_1 \equiv P \sqcap \exists r.E, \tag{5.13}$$
$$B_1 \equiv B_3 \sqcap \exists r.M, \tag{5.14}$$
$$B_1 \sqsubseteq B_2, \tag{5.15}$$
$$B_2 \sqsubseteq A_1, \tag{5.16}$$
$$A_2 \equiv B_3 \sqcap \exists r.M\} \tag{5.17}$$

and assume we extracted a subontology $\mathcal{S}$ from the source ontology $\mathcal{O}$ for the focus set $\Sigma_F = \{A_1, A_2\}$. Then the subontology is

$$\mathcal{S} = \{A_1 \equiv P \sqcap \exists r.E, \tag{5.18}$$
$$A_2 \equiv B_3 \sqcap \exists r.M\} \tag{5.19}$$

Then computing the UI-Diff$(\mathcal{S}, \mathcal{O})$, which is done by computing a uniform interpolant UI for the common symbols $\Sigma = \{A_1, P, r, E, B_3, M\}$ between both ontologies, implies forgetting the symbols $\{B_1, B_2\}$ from $\mathcal{O}$ to compute the UI

$$\mathcal{UI} = \{A_1 \equiv P \sqcap \exists r.E, \tag{5.20}$$
$$A_2 \equiv B_3 \sqcap \exists r.M, \tag{5.21}$$
$$B_3 \sqcap \exists r.M \sqsubseteq A_1\} \tag{5.22}$$

Checking for axioms that are in the UI of $\mathcal{O}$ but not entailed by the subontology $\mathcal{S}$ results in UI-Diff$(\mathcal{S}, \mathcal{O}) = \{B_3 \sqcap \exists r.M_1 \sqsubseteq A_1\}$.

If the original ontology contains only one definition for the concept name B_1, then the UI-Diff$(\mathcal{S}, \mathcal{O})$ *would be empty.*

Table 5.5 shows that there were no witnesses of the form $A \sqsubseteq C$, confirming that our method ensures that all the necessary/upward conditions for the concepts in Σ_F are included in the subontology (Definition 25).

Logical strength

We utilised Algorithm 15, which comutes the number of abstracted definitions that are weaker than the original definitions, as illustrated by the fourth column (#Weaker Σ_F defs). We determined the number of equivalent focus set definitions by subtracting the number of weaker abstracted definitions from the total number of focus set definitions (as illustrated by the third column).

We found that when using SNOMED CT to compute subontologies, the abstracted definitions have the same logical strength as the original definitions. This is because SNOMED CT is a terminology that includes no more than one concept definition for each concept name, hence empirically verifying our Theorem 22.

On the other hand, when the Gene ontology is used, the abstracted definitions derived may be weaker than the original definitions. The results for the logical strength of the abstracted definitions in the subontologies of the Gene ontology are shown in Table 5.5.

Table 5.6: Number of logical axioms, concepts and roles in the different types of extracts for focus sets in SNOMED CT and GENE ONTOLOGY.

Entity type	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles
SNOMED CT	Subontologies			Bottom modules			UIs		
Min.	683	779	19	2 758	2 754	30	N/A	N/A	N/A
Max.	86 811	86 608	46	102 628	102 617	55	N/A	N/A	N/A
Avg.	16 334.45	16 259.36	30.91	26 117	26 107.91	44.82	N/A	N/A	N/A
Med.	9 841	9 835	31	19 132	19 124	44	N/A	N/A	N/A
GENE ONTOLOGY	Subontologies			Bottom modules			UIs		
Min.	79	82	2	467	301	8	82	82	2
Max.	639	641	6	2 243	1 554	8	746	641	6
Avg.	241.25	245.83	3.67	1 162.83	758.67	8	288.25	245.83	3.67
Med.	188	191	4	974.5	618.5	8	217	191	4

Table 5.7: Number of logical axioms, concepts and roles in the different types of extracts for focus sets in SNOMED CT.

Entity type	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles
SNOMED CT	Subontologies			Bottom modules		
Heart Failure	5 815	5 782	38	15 132	15 119	52
Asthma	10 467	9 835	25	21 666	21 656	47
Epilepsy	690	779	22	2 758	2 754	30
Glaucoma	12 950	12 968	36	22 238	22 229	43
CKD	3 779	3 720	25	11 644	11 635	42
Osteoarthritis	9 898	10 024	37	17 754	17 744	44
Anaemia	6 712	6 398	38	22 917	22 907	53
Arthritis	9 276	9 176	23	17 706	17 699	38
Diabetes	86 819	86 608	19	102 628	102 617	47
Hypertension	19 589	19 659	46	33 712	33 703	55
Obesity	13 816	13 904	31	19 132	19 124	42

Table 5.8: Number of logical axioms, concepts and roles in the different types of extracts for small focus sets in SNOMED CT.

Entity type	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles
SNOMED CT	Subontologies			Bottom modules			UIs		
Epilepsy	29	40	5	144	145	5	29	41	5
Obesity	35	55	7	160	161	11	35	56	7
Pain aggravating factors	12	13	0	18	19	0	12	13	0
Pain easing factors	7	8	0	13	14	0	7	8	0
Posture findings	19	26	5	82	83	5	19	26	5
Transfer ability findings	76	88	3	117	118	3	77	89	3

5.5.6 Comparing the Size in Different Extracts

Subontology Results Against Bottom SLBM

Table 5.6 shows the minimum, maximum, average and median size of concepts, roles, and logical axioms in the subontologies and bottom modules. The results show that the number of concept and role names in the bottom modules were significantly larger than those in the subontologies in both corpora.

Looking at SNOMED CT results, the average number of logical axioms, concepts, and roles grew by 59.88%, 60.57%, and 45% in the bottom modules, respectively, in comparison to the subontologies. We show the sizes of each subontology and bottom module that were computed for each focus set in SNOMED CT *large* medical conditions sets (Table 5.7).

Subontology Results Against UIs

The size of the computed UIs for SNOMED CT's small focus sets is shown in Table 5.8. Except for the UI for *Transfer ability findings*, which surpasses the computed

Table 5.9: Number of logical axioms, concepts and roles in the different types of extracts for focus sets in GENE ONTOLOGY.

Entity type	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles	Logical Axioms	Concepts	Roles
GENE ONTOLOGY	Subontologies			Bottom modules			UIs		
goslim_agr	86	89	2	548	353	8	92	89	2
goslim_aspergillus	158	161	4	928	588	8	193	161	4
goslim_candida	160	163	3	921	587	8	180	163	3
goslim_chembl	467	467	6	1 741	1 174	8	582	467	6
goslim_drosophila	266	292	5	1 560	993	8	367	293	5
goslim_generic	245	248	4	1 166	1 554	8	290	248	4
goslim_metagenomics	216	219	4	901	611	8	241	219	4
goslim_mouse	79	82	2	467	301	8	82	82	2
goslim_pir	639	641	4	2 243	1 554	8	746	641	4
goslim_plant	155	158	3	755	493	8	172	158	3
goslim_pombe	140	142	3	1 021	626	8	155	142	3
goslim_yeast	284	287	4	1 703	1 702	8	359	287	4

subontology by one axiom, other UIs are equal in size to the computed subontologies. This axiom expresses a necessary condition of a supporting concept. We attribute the resemblance between UIs and subontologies to the fact that SNOMED CT is a terminology, and our subontology generation method captured all of the axioms for the focus set, but may miss some axioms for supporting symbols, as they are not focus concepts.

Table 5.9 shows the size of UIs of the Gene ontology. The results show that the number of logical axioms in UIs is greater than that in subontologies. There are two main explanations for this observation. The first is that the UI method has incorporated inferred axioms of the form $C \sqsubseteq A$, where A denotes a focus or a supporting concept name. For example, forgetting B_1 and B_2 from $\{B_1 \equiv C,\ B_1 \sqsubseteq B_2,\ B_2 \sqsubseteq A\}$, where all symbols in C are in the signature of the subontology, derives $C \sqsubseteq A$. The second is that when a focus concept A is described by multiple axioms, our method generated an abstracted definition that condenses these multiple axioms into a single axiom. For instance, the UI might contain two axioms for A, $A \equiv C_1$ and $A \sqsubseteq C_2$, while our method infers the axiom $A \equiv C_1 \sqcap C_2$, which results in fewer axioms in the subontology, but is a weaker definition for A.

Tables 5.8 and 5.9 show the numbers of concepts and roles in UIs are either the same as the corresponding subontology, or are one concept more than the number of subontology concepts. This additional concept was found to be the $\top$-concept.

5.5.7 Effect of Abstracted Definitions

We investigated the effect of abstracted definitions using the computed bottom modules because both extracts (subontologies and bottom modules) capture the input symbols'

Table 5.10: The values of Precision, $|((\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})\backslash \text{sig}(\mathcal{S}))|$, $|((\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \cap \text{sig}(\mathcal{S}))|$ and Rest of symbols of different medical conditions and gene slim focus sets in SNOMED CT (SCT) and GENE ONTOLOGY (GO), respectively.

Focus set Σ_F	Precision	$\lvert((\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})\backslash \text{sig}(\mathcal{S}))\rvert$	$\lvert((\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \cap \text{sig}(\mathcal{S}))\rvert$	Rest of symbols
Heart Failure (SCT)	0.38	745	723	8 606
Asthma (SCT)	0.45	1 176	1 666	10 667
Epilepsy (SCT)	0.29	25	29	1 958
Glaucoma (SCT)	0.58	362	1 093	8 906
CKD (SCT)	0.32	612	341	7320
Osteoarthritis (SCT)	0.57	683	315	7 044
Anaemia (SCT)	0.28	1 563	443	14 961
Arthritis (SCT)	0.51	864	1 023	7 674
Diabetes (SCT)	0.84	89	15 941	15 948
Hypertension (SCT)	0.58	566	1 459	13 487
Obesity (SCT)	0.74	162	995	5 069
goslim_agr (GO)	0.24	2	1	268
goslim_aspergillus (GO)	0.27	5	1	426
goslim_candida (GO)	0.27	4	1	425
goslim_chembl (GO)	0.39	40	3	669
goslim_drosophila (GO)	0.29	13	0	691
goslim_generic (GO)	0.32	6	1	502
goslim_metagenomics (GO)	0.35	14	2	382
goslim_mouse (GO)	0.26	1	2	224
goslim_pir (GO)	0.41	18	2	899
goslim_plant (GO)	0.31	9	0	331
goslim_pombe (GO)	0.22	3	0	486
goslim_yeast (GO)	0.26	16	0	773

upward relationships. We illustrate how computing subontologies from abstracted definitions can greatly reduce the size of subontologies, by utilising two metrics: precision and interval. Additionally, we also computed the precision values in uniform interpolants.

Precision

Table 5.10 shows the computed values for Precision for each bottom module, as well as the interval sets $|(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})\backslash \text{sig}(\mathcal{S})|$, $|(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F}) \cap \text{sig}(\mathcal{S})|$ and the Rest of Symbols. Precision ratings for the bottom modules range between 28% and 84% for SNOMED CT focus sets and are almost half that for the Gene ontology focus sets (22%–41%). Low precision rates are as a result of how $\bot$-type of SLBM works, which captures all axioms relating to the input signature ascending until it reaches the $\top$ concept in $\mathcal{O}$. (See Section 3.1.1 (SLBM).)

Computing precision rates for the UIs returned 100% for all UIs since we assumed that the $\top$ concept is included in the computed extracts even if it is not explicitly stated.

Interval $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})$

The value $|(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})|$ is the number of concepts between the focus concepts and their closest primitive ancestors (Definition 23). Table 5.10 shows the number of concepts in the interval set in the bottom modules. We show the number of concepts in the interval sets that are in the signature of the computed subontologies (column 3) and those that are outside the signature of the subontologies (column 4). The sizes of the set $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})\backslash\text{sig}(\mathcal{S})$ indicate that there are a considerable number of concepts in the bottom modules that can be avoided by abstracting the definitions to their closest primitive ancestors.

For information we also computed the $|(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})|$ values for the Gene ontology's bottom modules. Because the Gene ontology is not a terminology, the concepts in $(\Sigma_F, \mathcal{P}^{\sqcap}_{\Sigma_F})\backslash\text{sig}(\mathcal{S})$ cannot be considered as being redundant, as they can be described in terms of concepts that the focus concepts may not inherit by abstraction.

5.5.8 Examining the Semantic Difference

The signature of the subontologies is a subset of the signature of the computed bottom modules and UIs. This means that the shared signature Σ used when computing the semantic difference between subontologies and the other extract types is the signature of the subontology. This is illustrated via the following relation.

$$\text{Because } \ \text{sig}(\mathcal{S}) \subseteq \text{sig}(\text{extract})$$
$$\text{Therefore } \ \text{sig}(\mathcal{S}) \cap \text{sig}(\text{extract}) = \text{sig}(\mathcal{S})$$

Two sets of UI-based semantic differences were generated between a subontology $\mathcal{S}$ and either a $\bot$-module or a UI using the SNOMED CT and the Gene ontology corpus. These sets are the following:

1. UI-Diff(extract, $\mathcal{S}$): This is the set of axioms in a subontology that are not entailed by an extract over the shared signature. This set was found to be empty for both types of extracts in both corpora. This means that all the axioms in the computed subontologies are entailed by the $\bot$-modules and the UIs.

2. UI-Diff($\mathcal{S}$, extract): This is the set of axioms in an extract that are not entailed by a subontology over the shared signature. When computing the UI-Diff between a subontology and a $\bot-$module or a UI, we found that in all cases the set contained witnesses in both corpora.

Table 5.11: UI-based witness analysis in the 12 go slim focus sets of Gene ontology for the set UI-Diff($\mathcal{S}$, $\bot$-module)

Difference Type	Σ_F witnesses		Σ_S witnesses		GCIs	Total
Axiom form	$A \sqsubseteq C$	$C \sqsubseteq A$	$A \sqsubseteq C$	$C \sqsubseteq A$	$C \sqsubseteq D$	
goslim_agr	0	1	5	6	6	13
goslim_aspergillus	0	9	47	35	89	180
goslim_candida	0	6	38	17	45	106
goslim_chembl	0	15	73	106	775	969
goslim_drosophila	0	9	75	113	391	588
goslim_generic	0	7	55	39	91	192
goslim_metagenomics	0	5	31	8	75	119
goslim_mouse	0	2	5	0	1	8
goslim_pir	0	15	92	40	381	528
goslim_plant	0	0	47	9	44	100
goslim_pombe	0	14	28	7	15	64
goslim_yeast	0	34	52	44	106	236

Table 5.12: UI-based witness analysis in the 12 go slim focus sets of Gene ontology for the set UI-Diff($\mathcal{S}$, UI)

Difference Type	Σ_F witnesses		Σ_S witnesses		GCIs	Total
Axiom form	$A \sqsubseteq C$	$C \sqsubseteq A$	$A \sqsubseteq C$	$C \sqsubseteq A$	$C \sqsubseteq D$	
goslim_agr	0	1	5	1	4	11
goslim_aspergillus	0	5	46	11	21	83
goslim_candida	0	2	37	5	14	58
goslim_chembl	0	10	73	24	85	192
goslim_drosophila	0	8	75	22	71	176
goslim_generic	0	7	54	16	24	101
goslim_metagenomics	0	1	31	5	19	56
goslim_mouse	0	2	5	0	1	8
goslim_pir	0	5	90	11	95	201
goslim_plant	0	0	47	5	13	65
goslim_pombe	0	4	28	5	7	44
goslim_yeast	0	10	51	22	46	129

Tables 5.11 and 5.12 illustrate the number of computed semantic differences between subontologies and bottom modules UI-Diff($\mathcal{S}$, $\bot$-module) (or UIs UI-Diff($\mathcal{S}, \mathcal{UI}$)) for the Gene ontology extracts. We observe that the semantic differences computed between subontologies and bottom modules are higher than those computed between subontologies and UIs. The primary reason for this observation is that the bottom modules contained a greater number of axioms, which captured additional information about the signature of the computed subontologies. This increased information

Table 5.13: UI-based witness analysis in the six small focus sets of SNOMED CT for the set UI-Diff($\mathcal{S}$, $\bot$-module)

Difference Type	Σ_F witnesses		Σ_S witnesses		GCIs	Total
Axiom form	$A \sqsubseteq C$	$C \sqsubseteq A$	$A \sqsubseteq C$	$C \sqsubseteq A$	$C \sqsubseteq D$	
Epilepsy	0	0	0	0	0	0
Obesity	0	0	3	0	0	3
Pain aggravating factors	0	0	0	0	0	0
Pain easing factors	0	0	0	0	0	0
Posture findings	0	0	0	0	0	0
Transfer ability findings	0	0	1	0	0	1

Table 5.14: UI-based witness analysis in the six small focus sets of SNOMED CT for the set UI-Diff($\mathcal{S}$, UI)

Difference Type	Σ_F witnesses		Σ_S witnesses		GCIs	Total
Axiom form	$A \sqsubseteq C$	$C \sqsubseteq A$	$A \sqsubseteq C$	$C \sqsubseteq A$	$C \sqsubseteq D$	
Epilepsy	0	0	0	0	0	0
Obesity	0	0	2	0	0	2
Pain aggravating factors	0	0	0	0	0	0
Pain easing factors	0	0	0	0	0	0
Posture findings	0	0	0	0	0	0
Transfer ability findings	0	0	1	0	0	1

captured is represented in the semantic difference between subontologies and bottom modules.

In both comparisons UI-Diff($\mathcal{S}, \bot$-module), UI-Diff($\mathcal{S}, \mathcal{UI}$), all semantic differences with respect to Σ_F are of the kind $C \sqsubseteq A$ (Tables 5.11 and 5.12). This is because the subontology generation method captured all of the upward conditions associated with focus symbols, as indicated by the results. Notably, because the Gene ontology is not a terminology, we obtain a number of focus set axioms that are caught by the SLBM $\bot$-type and uniform interpolation methods but not by the subontology generation method. However, when the input ontology is a terminology, the subontology generation method captures all of the focus set axioms AX_{Σ_F}, and thus the semantic differences with respect to Σ_F between the subontologies and the other extract types would be empty. This is demonstrated in Tables 5.13 and 5.14, where the use of SNOMED CT, which is a terminology, did not result in focus set witnesses.

Additionally, the number of supporting set witnesses outweighed the number of focus set witnesses as illustrated by the third column in Tables 5.11, 5.12, 5.13 and 5.14. This is natural, given that our method derives the necessary conditions for concepts in Σ_S only if they appear in the signature of the focus set definitions.

From the UI-based logical difference results, we observe that for the evaluation on both ontologies showed, the following expected relationships hold:

$$\bot\text{-module} \models \mathcal{S} \text{ but } \mathcal{S} \not\models \bot\text{-module} \tag{5.23}$$

$$\mathcal{UI} \models \mathcal{S} \text{ but } \mathcal{S} \not\models \mathcal{UI} \tag{5.24}$$

5.6 Discussion

The following are broad observations regarding our method that are confirmed by our empirical evaluation.

Minimal Extract. Among the two ontology extraction methods described, including the $\bot$-type of SLBM and uniform interpolation methods in our comparative evaluations, our subontologies represent the minimal information required by the set of input focus symbols, as indicated by the relations (5.23) and (5.24) in our semantic differences analysis.

Effect of Abstraction. As can be seen from Table 5.10, the values of Precision and Interval in bottom modules indicate that employing the notion of abstracted definitions to compute subontologies significantly reduced the size of the resulting subontologies. This is because abstracted definitions help removing symbols that are deemed *redundant*, when the input ontology from which we aim to produce subontologies is a terminology. The conditions of such redundant concepts are inherited by the generated abstracted definitions.

Semantic Difference. It was found that a large number of axioms in the bottom modules were not entailed by the subontologies (see Tables 5.11 and 5.13), whereas a smaller number of axioms in the UIs were not entailed by the subontologies (see Tables 5.12 and 5.14). This is because bottom modules have a high proportion of supporting symbols that do not exist in the corresponding subontology, as indicated by the values of the Precision metric (see Table 5.10).

Terminology or Not. Although the used input signature for computing uniform interpolants from SNOMED CT in our evaluation is quite small (see Table 5.8 for sizes of SNOMED CT *small* focus sets extracts), inspection of the results indicated that uniform interpolants are very close to subontologies when the input ontology is a *terminology* (as is the case with SNOMED CT).

5.7 Conclusion

In this chapter, we introduced a notion of subontology based on the idea of abstracted definitions. Our abstracted definitions follow the same format as the commonly used canonical form in SNOMED CT. This ensures abstracted definitions explicitly state all possible constraints and defining characteristics of particular concepts to facilitate implementation, recording, storage, and retrieval within SNOMED CT. Additionally, they enable more precise inferred parent identification of concepts after running a classifier, they simplify parent relationship maintenance, and improve the accuracy and breadth of super and subconcepts [Ulr].

Our notion of subontologies can be advantageous for ontologies other than SNOMED CT, as it is generally applicable to ontologies defined in the $\mathcal{ELH}$ language that do not include GCIs, such as SNOMED CT, the Gene Ontology, and the Sequence Ontology.

We demonstrated that when the input ontology is a terminology, the method generates equivalent focus set subontologies; however, when the input ontology is not a terminology, the method may yield subontologies with weaker definitions.

We illustrated the quality of the generated subontologies by comparing them to $\bot$-modules and UIs in terms of size, precision, and semantic difference of the different extracts. Through the size measure, we found that the computed subontologies are more concise than both $\bot$-modules and UIs. Specifically, when compared to bottom modules, our abstracted definition-based subontologies contain far fewer supporting set symbols and axioms while keeping the familiar axiom structure.

The precision metric demonstrated that high precision was achieved when computing UIs. Moreover, the interval notion provides insight into why abstracted definitions in subontologies are smaller than bottom modules while maintaining all of the defining characteristics of the focus concepts.

Finally, the semantic difference analysis demonstrated that bottom modularisation can be used to collect axioms about the signature of computed subontologies that the uniform interpolation approach does not capture. This is to be expected, as bottom

modularisation captures all of the super concepts contained in the input seed signature, whereas UIs represent constrained views of ontologies that are exactly defined by the input symbols.

Chapter 6

Focus Set Semantic Differences

Biomedical ontologies such as SNOMED CT and NCIt are divided into modules that describe various sub-domains such as cancer, general practice and vaccines domains. As discussed in Chapter 3, tracking semantic differences between massive ontologies can result in extremely large differences that are dispersed across various subdomains. This complicates and impairs the analysis of such differences. A solution to such issue is to zoom in on a specific portion of the broad ontology in order to track the differences associated with that portion.

In this chapter, we present a method to generating focus set semantic differences based on input focus set of symbols. The approach combines two existing methods, the method for generating focus set subontologies (see Chapter 5) [ASDG21b], and the UI-Diff tool introduced in [LLA+21a] (see Chapter 3, Section 3.2.1). Our method produces subontologies for a specified focus symbols of two different ontologies' versions based on the user's selection of focus symbols, and then utilise the generated subontologies to track semantic differences between them. The resulting semantic differences are subsequently categorised using our witnesses analysis scheme. This scheme aids the analysis of the resulting differences.

In Section 6.1, we motivate our method for generating focus set semantic differences using a motivating example, and illustrate why using the UI-Diff method alone is impractical when dealing with large ontologies. In Section 6.2, we describe our approach to computing focused differences. In Section 6.3, we discuss key aspects related to our method that affect the method's properties. We describe a technique for analysing the resulting witness sets in Section 6.4. In Section 6.5, we evaluate our method for computing focus set semantic differences using several standard refsets of SNOMED CT. We discuss some scenarios that illustrate how modellers can benefit

from witness sets generated by our method in Section 6.6. Lastly, in Section 6.7, we bring this chapter to a close.

6.1 Motivating Example

Assume that the user is interested in tracking the differences between $\mathcal{O}_1$ and $\mathcal{O}_2$ with respect to a focus set of symbols, and in learning about changes to the focus concepts' definitions and signatures. Then one can generate subontologies from the original ontologies for the chosen focus set to track the semantic difference relevant to the input focus set. In the following example, we illustrate how tracking semantic differences using subontologies gives more relevant differences to the chosen set of focus symbols.

Example 10. *Consider the following two ontologies, describing definitions about two renal diseases from the ERA refset [NCS+18]:*

$$\begin{aligned}\mathcal{O}_1 = \{&\textbf{\textit{Renal artery stenosis}} \equiv \mathit{Disease} \sqcap \exists \mathit{location.Renal\ artery},\\ &\textbf{\textit{Acute renal failure syndrome}} \equiv \mathit{Disease} \sqcap \exists \mathit{location.Kidney},\\ &\mathit{Renal\ artery} \sqsubseteq \mathit{Vascular\ structure\ of\ kidney},\\ &\mathit{Vascular\ structure\ of\ kidney} \sqsubseteq \mathit{Kidney}\}\end{aligned}$$

and

$$\begin{aligned}\mathcal{O}_2 = \{&\textbf{\textit{Renal artery stenosis}} \equiv \mathit{Disease} \sqcap \exists \mathit{location.Renal\ artery},\\ &\textbf{\textit{Acute renal failure syndrome}} \equiv \mathit{Disease} \sqcap \exists \mathit{location.Kidney},\\ &\mathit{Renal\ artery} \sqsubseteq \mathit{Arterial\ supply},\\ &\mathit{Arterial\ supply} \sqsubseteq \mathit{Vascular\ structure\ of\ kidney},\\ &\mathit{Vascular\ structure\ of\ kidney} \sqsubseteq \mathit{Abdominal\ organ}\}\end{aligned}$$

Assume that the user is interested in tracking the differences between $\mathcal{O}_1$ and $\mathcal{O}_2$ with respect to the focus set $\Sigma_F = \{$Renal artery stenosis, Acute renal failure syndrome$\}$, to track semantic differences about two types of renal diseases in Σ_F. Then, the user would generate the subontologies $\mathcal{S}_1$ and $\mathcal{S}_2$ for Σ_F from $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively which

are:

$$\mathcal{S}_1 = \{\textbf{\textit{Renal artery stenosis}} \equiv \textit{Disease} \sqcap \exists \textit{location.Renal artery},$$
$$\textbf{\textit{Acute renal failure syndrome}} \equiv \textit{Disease} \sqcap \exists \textit{location.Kidney},$$
$$\textit{Renal artery} \sqsubseteq \textit{Kidney}\}$$

and

$$\mathcal{S}_2 = \{\textbf{\textit{Renal artery stenosis}} \equiv \textit{Disease} \sqcap \exists \textit{location.Renal artery},$$
$$\textbf{\textit{Acute renal failure syndrome}} \equiv \textit{Disease} \sqcap \exists \textit{location.Kidney}\}.$$

The focus concepts are highlighted in bold in $\mathcal{S}_1$ and $\mathcal{S}_2$. Then computing UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$ *for the common symbols $\Sigma = \{$**Renal artery stenosis**, Disease, location, Renal artery, **Acute renal failure syndrome**, Kidney$\}$ to get the axioms entailed by $\mathcal{S}_1$ but not $\mathcal{S}_2$ gives the witness set:*

$$\mathcal{W} = \{\textit{Renal artery} \sqsubseteq \textit{Kidney}\} \tag{6.1}$$

On the other hand, computing UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ *for the common symbols $\Sigma = \{$**Renal artery stenosis**, Disease, location, Renal artery, **Acute renal failure syndrome**, Kidney, Vascular structure of kidney $\}$ gives the witness set:*

$$\mathcal{W} = \{\textit{Vascular structure of kidney} \sqsubseteq \textit{Kidney}\} \tag{6.2}$$

We notice that the witness (6.1) generated when computing UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$ is more relevant to the focus set Σ_F than the witness (6.2) that was generated when computing UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$. This is because the set UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$ illustrates a change that corresponds to the concept *Renal artery*, which is a supporting concept used in the definition of the focus concept *Renal artery stenosis*. On the other hand, the set UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ illustrates a change corresponding to the parent concept of *Renal artery*, which is *Vascular structure of kidney*.

One could argue that we can restrict the common signature Σ when computing UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ by excluding the concept *Vascular structure of kidney* from Σ to give exactly the same result as when computing UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$. However, there are several issues with this approach, some of them are:

1. It will be hard for the user to manually exclude or add concepts to the common signature set Σ when computing the UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ to compute only those focused differences related to a particular section of the ontology.

2. The computation of UI-Diff directly between ontologies as large as SNOMED CT is intricate and might not terminate in some cases in reasonable time (see our results on uniform interpolation method in Section 4.4).

3. Since $\mathcal{O}_1$ and $\mathcal{O}_2$ are typically significantly larger than their corresponding sub-ontologies $\mathcal{S}_1$ and $\mathcal{S}_2$ the witness set in UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ can also be large and hard to analyse.

6.2 Generating Focus Set Semantic Differences

The aim of our method is to identify semantic differences between two ontology versions for a given set of focus symbols Σ_F that is specific to a particular section of the ontologies.

Before discussing our method's steps, we give a reminder on computing semantic differences using the UI-Diff method.

As discussed in Section 3.2.1, the UI-Diff method computes a finite representation of the semantic differences (witnesses) $\mathcal{W}$, which represent ontology changes. The difference between $\mathcal{O}_1$ and $\mathcal{O}_2$ is empty iff $\mathcal{O}_1 \models \mathcal{UI}_2$, where $\mathcal{UI}_2$ is a Σ-uniform interpolant of $\mathcal{O}_2$ computed for $\Sigma \subseteq \text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_2)$. If $\mathcal{O}_1 \not\models \mathcal{UI}_2$ it means that every $\alpha \in \mathcal{UI}_2$ not entailed by $\mathcal{O}_1$ is a witness. Generating the differences using UI-Diff can help understand whether there has been changes in the axioms of $\mathcal{O}_2$ that share symbols from $\mathcal{O}_1$ but are not entailed by $\mathcal{O}_1$.

The steps of computing semantic differences using the UI-Diff method between $\mathcal{O}_1$ and $\mathcal{O}_2$ are as follows.

1. Using uniform interpolation, compute $\mathcal{UI}_2$ of $\mathcal{O}_2$ for $\Sigma = \text{sig}(\mathcal{O}_1) \cap \text{sig}(\mathcal{O}_1)$.

2. Using an external reasoner, compute the set $\mathcal{W}_2$, which consists of the axioms $\alpha \in \mathcal{UI}_2$ but for which $\mathcal{O}_1 \not\models \alpha$.

The HermiT reasoner is utilised to do the entailment check in our implementation, as it contains a built-in function for entailment checking [GHM+14].

Figure 6.1 shows our method of generating focus set semantic differences for two versions of an ontology. The input to our method is two ontology versions $\mathcal{O}_1$ and $\mathcal{O}_2$

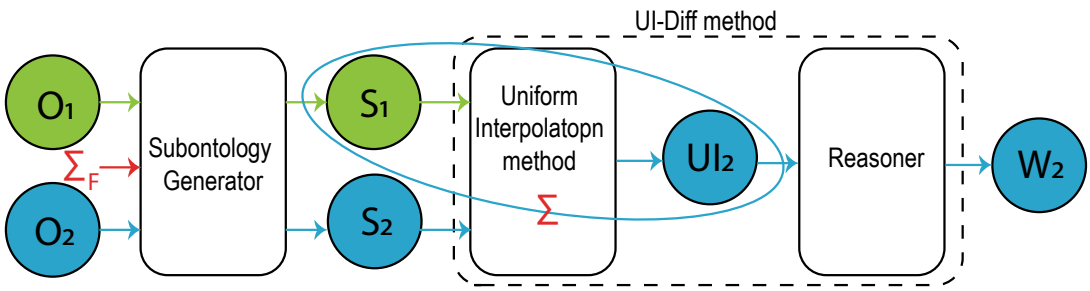

Figure 6.1: Computing UI-based differences between two ontologies for input focus set Σ_F based on subontology generation

and a focus set Σ_F. The method computes the output $\mathcal{W}$, which is the set of focus set semantic difference. The generation of focus set semantic difference is done in two steps:

- Step (1): Using the focus set subontology generation method, we generate two subontologies $\mathcal{S}_1$ and $\mathcal{S}_2$ for the input focus set Σ_F from $\mathcal{O}_1$ and $\mathcal{O}_2$. (See Chapter 5 about the generation of focus set subontologies).

- Step (2): Using the UI-Diff method, witnesses can be computed between the subontologies $\mathcal{S}_1$ and $\mathcal{S}_2$. In Figure 6.1, the UI-Diff method generates axioms α that are in $\mathcal{UI}_2$ computed from $\mathcal{S}_2$ but for which $\mathcal{S}_1 \not\models \alpha$. This step generates the set of witnesses $\mathcal{W}_2$.

The witness set $\mathcal{W}_2$ is a representation of UI-Diff$(\mathcal{S}_1, \mathcal{S}_2)$. The method can be utilised to generate the set of witnesses $\mathcal{W}_1$ by switching the places of $\mathcal{S}_1$ and $\mathcal{S}_2$.

Assuming that $\mathcal{O}_1$ is the older version, while $\mathcal{O}_2$ is the newer version, we note that UI-Diff$(\mathcal{S}_1, \mathcal{S}_2)$ computes the information gained from $\mathcal{S}_1$ to $\mathcal{S}_2$, or the information lost in $\mathcal{S}_2$ from $\mathcal{S}_1$, where $\mathcal{S}_1$ and $\mathcal{S}_2$ are subontologies extracted from $\mathcal{O}_1$ and $\mathcal{O}_2$, respectively, for a given focus set Σ_F.

After generating the witness sets, we analyse them using the segmentation technique described in Section 6.4.

6.3 Key Aspects of the Method

Certain factors have an effect on the method's properties. We illustrate these factors with accompanying examples that demonstrate how they affect the method's results.

- **The first factor:** Abstracted definitions of focus concepts lead to additional witnesses pertinent to the input focus symbols. (See Example 12).
- **The second factor:** Focus concepts in subontologies are defined in terms of the closest primitive ancestors, which may not be the case in the original ontology. As a result, the syntactical structure of the definitions in the original ontology and the subontology would differ. Such difference in the syntactical structure may yield different results when computing UI-Diff between subontologies rather than between the original ontologies. (See Example 13).
- **The third factor:** If the input ontology is not a terminology, then the generated abstracted definition may be weaker, which is then reflected in the resulting set of witnesses. (See Example 14).

Each of these factors is now discussed in detail.

Factor 1: Abstracted Definitions of Focus Concepts. Because our approach for computing subontologies relies on computing abstracted definitions for focus concepts, these abstracted definitions retain certain redundant conditions. As a property of abstracted definitions, we include all of the defining characteristics of the focus concept in the form of existential restrictions. Some of these existential restrictions are redundant if they are subsumed by atomic parent concepts in the same definition. (See Section 5.1 about upwardly abstracted definitions). The following example is a reminder of the presence of a redundant condition in the abstracted definition.

Example 11. *Consider the ontology*

$$\begin{aligned}\mathcal{O} = \{&A \equiv D \sqcap \exists r.C_1,\\ &D \equiv P \sqcap \exists r.C_2,\\ &P \sqsubseteq \exists r.C_3\}\end{aligned}$$

Since P is the closest primitive concept above A in the concept hierarchy, an abstracted definition of A is

$$A \equiv P \sqcap \exists r.C_1 \sqcap \exists r.C_2 \sqcap \exists r.C_3.$$

As a result of how redundant conditions are being eliminated from abstracted definitions, computing UI-Diff between subontologies can result in computing a witness set

that contain redundant axioms. These redundant axioms in the witness sets reflect the redundant conditions in the generated abstracted definitions. The following example illustrates this.

Example 12. *Let*

$$\begin{aligned}\mathcal{O}_1 = \{&A \equiv P_1 \sqcap \exists r.B_1,\\ &P_1 \sqsubseteq P_2 \sqcap \exists r.B_2,\\ &P_3 \sqsubseteq P_1\}\end{aligned}$$

and

$$\begin{aligned}\mathcal{O}_2 = \{&A \equiv P_2 \sqcap \exists r.B_1,\\ &P_1 \sqsubseteq P_2 \sqcap \exists r.B_2,\\ &P_3 \sqsubseteq P_1\}.\end{aligned}$$

and $\Sigma = sig(\mathcal{O}_1) = sig(\mathcal{O}_2)$ *are the common symbols of* $\mathcal{O}_1$ *and* $\mathcal{O}_2$*. We can see that* $\mathcal{O}_1$ *and* $\mathcal{O}_2$ *are similar to each other, where the only difference is in the definition of* A*. The concept name* A *in* $\mathcal{O}_2$ *is defined in terms of* P_2 *rather than* P_1 *as in* $\mathcal{O}_1$*. Computing* UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ *gives the witness* $A \sqsubseteq P_1$*. However, computing* UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$ *between the generated subontologies* $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *of* $\mathcal{O}_1$ *and* $\mathcal{O}_2$ *for the focus set* $\Sigma_F = \{A,\ P_3\}$ *respectively, where =*

$$\begin{aligned}\mathcal{S}_1 = \{&A \equiv P_1 \sqcap \exists r.B_1 \sqcap \exists r.B_2,\\ &P_3 \sqsubseteq P_1 \sqcap \exists r.B_2,\\ &P_1 \sqsubseteq \exists r.B_2\}\end{aligned}$$

and

$$\begin{aligned}\mathcal{S}_2 = \{&A \equiv P_2 \sqcap \exists r.B_1,\\ &P_3 \sqsubseteq P_1 \sqcap \exists r.B_2,\\ &P_1 \sqsubseteq P_2 \sqcap \exists r.B_2\}\end{aligned}$$

gives the witness set: $\{A \sqsubseteq P_1, A \sqsubseteq \exists r.B_2\}$.

We can see that UI-Diff$(\mathcal{S}_2, \mathcal{S}_1) = \{A \sqsubseteq P_1,\ A \sqsubseteq \exists r.B_2\}$ includes the additional axiom $A \sqsubseteq \exists r.B_2$. This additional witness was not revealed when comparing between

the original ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$. This is because the definition of A in $\mathcal{O}_1$ does not include all the existential restrictions (defining characteristics) that the concept A inherits (including $\exists r.B_2$). Thus, the definition of A in $\mathcal{UI}_1$ generated when computing $\mathsf{UI\text{-}Diff}(\mathcal{O}_2, \mathcal{O}_1)$ does not include the additional condition ($\exists r.B_2$) that A inherits from P_1. This is because the uniform interpolation method computes only strongest Σ-entailments of the input ontology. For clarification, we display the set of axioms within the $\mathcal{UI}_1$ when computing $\mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1)$.

$$\mathcal{UI}_1 \text{of } \mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1) = \{A \sqsubseteq P_1, \tag{6.3}$$
$$A \sqsubseteq \exists r.B_1, \tag{6.4}$$
$$\boxed{A \sqsubseteq \exists r.B_2}, \tag{6.5}$$
$$P_1 \sqcap \exists r.B_1 \sqcap \exists r.B_2 \sqsubseteq A, \tag{6.6}$$
$$P_3 \sqsubseteq P_1, \tag{6.7}$$
$$P_3 \sqsubseteq \exists r.B_2, \tag{6.8}$$
$$P_1 \sqsubseteq \exists r.B_1\} \tag{6.9}$$

We can see that $\mathcal{UI}_1$ of $\mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1)$ includes normalised axioms of $\mathcal{S}_1$ where their symbols are in Σ. After checking which axioms in $\mathcal{UI}_1$ that are not entailed by $\mathcal{S}_2$, the resulting set of witnesses is $\{A \sqsubseteq P_1, A \sqsubseteq \exists r.B_2\}$.

We note that the additional witness $A \sqsubseteq \exists r.B_2$ tells the user which condition that the focus concept A inherits from the parent concept P (in $\mathcal{S}_1$) where such condition is not entailed by the latter version ($\mathcal{S}_2$).

Factor 2: Syntactic Difference of Definitions. The focus concept in the subontology is defined in terms of its closest primitive ancestors, which may not be the case in the original ontology. This results in a syntactic difference between the focus concept's definitions in the subontology and the original ontology, which may produce different results when computing $\mathsf{UI\text{-}Diff}$. To demonstrate this, consider the following example.

Example 13. *Let*

$$\begin{aligned}\mathcal{O}_1 = \{&A \sqsubseteq D_1 \sqcap \exists r.B,\\ &D_1 \equiv D_5 \sqcap \exists r.B,\\ &D_5 \sqsubseteq \exists r.B\}\end{aligned}$$

and

$$\begin{aligned}\mathcal{O}_2 = \{&A \equiv D_2 \sqcap \exists r.B,\\ &D_1 \equiv D_5 \sqcap \exists r.B,\\ &D_5 \sqsubseteq \exists r.B\}\end{aligned}$$

Generating the subontologies $\mathcal{S}_1$ and $\mathcal{S}_2$ form $\mathcal{O}_1$ and $\mathcal{O}_2$ for $\Sigma_F = \{A, D_1\}$ gives the following:

$$\begin{aligned}\mathcal{S}_1 = \{&A \sqsubseteq D_5 \sqcap \exists r.B,\\ &D_1 \equiv D_5 \sqcap \exists r.B,\\ &D_5 \sqsubseteq \exists r.B\}\end{aligned}$$

and

$$\mathcal{S}_2 = \mathcal{O}_2$$

We note that $\mathcal{S}_1$ is logically equivalent to $\mathcal{O}_1$ but syntactically differ. $\mathcal{S}_1$ defines A in terms of its closest primitive ancestor, which is D_5.
Computing UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ *to generate axioms that are entailed by $\mathcal{O}_1$ but not $\mathcal{O}_2$ results in the witness $A \sqsubseteq D_1$. However, computing* UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$ *gives the witness $A \sqsubseteq D_5$.*

We can see why there is a difference in the results when computing UI-Diff between the original ontologies than between the subontologies by looking at the generated UI. When computing UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ the generated UI for the common symbols $\Sigma = \{A,\ D_1,\ r,\ D_5,\ B\}$ is:

$$\begin{aligned}\mathcal{UI}_1 \text{of UI-Diff}(\mathcal{O}_2, \mathcal{O}_1) = \{&\boxed{A \sqsubseteq D_1},\\ &A \sqsubseteq \exists r.B,\\ &D_1 \sqsubseteq D_5,\\ &D_1 \sqsubseteq \exists r.B,\\ &D_5 \sqcap \exists r.B \sqsubseteq D_1\}.\end{aligned}$$

However, when computing $\mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1)$ the generated UI is:

$$\begin{aligned}\mathcal{UI}_1 \text{of } \mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1) = \{&\boxed{A \sqsubseteq D_5},\\ &A \sqsubseteq \exists r.B,\\ &D_1 \sqsubseteq D_5,\\ &D_1 \sqsubseteq \exists r.B,\\ &D_5 \sqcap \exists r.B \sqsubseteq D_1\}.\end{aligned}$$

$\mathcal{UI}_1$ of $\mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1)$ includes the axiom $A \sqsubseteq D_5$ (the axiom within the box) instead of $A \sqsubseteq D_1$ because A in $\mathcal{S}_1$ is defined in terms of the closest primitive ancestor D_5. We note that $\mathcal{S}_1$ preserves the relation $A \sqsubseteq D_1$ in the concept hierarchy $\mathcal{H}_{\mathcal{S}_1} = \{A \sqsubseteq D_1, D_1 \sqsubseteq D_5\}$. The relation $A \sqsubseteq D_1$ is stronger than the relation $A \sqsubseteq D_5$ because D_1 directly subsumes A, whereas D_5 subsumes A indirectly. The concept hierarchy $\mathcal{H}_{\mathcal{S}_1}$ of $\mathcal{S}_1$ is identical to the concept hierarchy of $\mathcal{O}_1$. The reason the UI computed when producing $\mathsf{UI\text{-}Diff}(\mathcal{S}_2, \mathcal{S}_1)$ does not result in such an inferred stronger relation ($A \sqsubseteq D_1$) is that forgetting was not used to generate such an inferred axiom, i.e., no symbols were forgotten in order to for the inferred axiom $A \sqsubseteq D_1$ to be generated.

Factor 3: Terminology or Not. When the input ontology to the subontology generation method is not a terminology, i.e., contains more than one definition for a concept name A, then the generated abstracted definition for A can be weaker than the original definition. This might be reflected in the resulting set of witnesses as a witness of the form $C \sqsubseteq A$, where A is a focus concept. Consider the following example.

Example 14. *Let*

$$\begin{aligned}\mathcal{O}_1 = \{&A \equiv P_1 \sqcap \exists r.C_1,\\ &A \sqsubseteq P_2,\\ &P_3 \sqsubseteq P_1 \sqcap P_2\}\end{aligned}$$

and

$$\begin{aligned}\mathcal{O}_2 = \{&A \equiv P_1 \sqcap \exists r.C_1,\\ &P_3 \sqsubseteq P_1 \sqcap P_2\}.\end{aligned}$$

We note that $\mathcal{O}_1$ includes two axioms for the concept name A.

$\Sigma = sig(\mathcal{O}_1) = sig(\mathcal{O}_2)$ is a set of the common symbols in $\mathcal{O}_1$ and $\mathcal{O}_2$. Generating the subontologies $\mathcal{S}_1$ and $\mathcal{S}_2$ from $\mathcal{O}_1$ and $\mathcal{O}_2$ for $\Sigma_F = \{A, P_3\}$ gives the following subontologies.

$$\mathcal{S}_1 = \{A \equiv P_1 \sqcap P_2 \sqcap \exists r.C_1,$$
$$P_3 \sqsubseteq P_1 \sqcap P_2\}$$

and $\mathcal{S}_2 = \mathcal{O}_2$.

Computing UI-Diff$(\mathcal{S}_2, \mathcal{S}_1)$ to generate axioms that are entailed by $\mathcal{S}_1$ but not $\mathcal{S}_2$ results in the witness $A \sqsubseteq P_2$. The other direction UI-Diff$(\mathcal{S}_1, \mathcal{S}_2)$ to generate axioms that are entailed by $\mathcal{S}_2$ but not $\mathcal{S}_1$ results in the witness $P_1 \sqcap P_2 \sqcap \exists r.C_1 \sqsubseteq A$.

The witness $P_1 \sqcap P_2 \sqcap \exists r.C_1 \sqsubseteq A$ was generated as a result of computing the abstracted definition of A, which is weaker than the original definition. On the other hand, computing UI-Diff$(\mathcal{O}_1, \mathcal{O}_2)$ results in empty set, since the definition of A in $\mathcal{O}_2$ is not weaker than A's definition in $\mathcal{O}_1$. Such witnesses of focus concepts are not computed if the input ontology to the subontology generation method is a terminology.

Based on the aforementioned factors, we conclude the following aspects about our method:

1. Through the first factor, we discovered that additional witnesses can be generated as a result of abstracting focus concept definitions. These additional witnesses assist modellers in precisely understanding conditions that the focus concepts inherit, where such conditions are not entailed by the other ontology version.

2. The second factor demonstrates that witnesses computed between subontologies might differ to those that were computed between the original ontologies. This is because abstracted definitions associated with focus concepts in the subontology differ syntactically from their original definitions in the original ontology.

3. The third factor states that the input ontologies should be terminologies in order to compute semantic differences between subontologies. This is to avoid generating witnesses as a result of computing weaker definitions of focus concepts than the original definition (see Section 5.1 about upwardly abstracted definitions).

6.4 Analysis of the Witness Sets

In this section, we distinguish between the resulting witnesses based on whether the witness is stated or inferred. We also distinguish them according to whether the witness is associated with a focus or supporting concept. Based on such distinctions we present an analysis scheme to analyse the resulting witnesses by partitioning them according to the identified sets.

6.4.1 Stated or Inferred Witnesses

Distinguishing stated from inferred witnesses enables a clear understanding of the resulting witnesses, as inferred witnesses highlight hidden axioms or unexpected consequences identified while tracking semantic differences. The set of *stated* and *inferred* witnesses, are defined as follows.

Definition 26 (Stated (Inferred) Witness)**.** *Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two normalised ontologies. Let $\mathcal{W}$ be the witness set of all axioms α given by computing* UI-Diff*($\mathcal{O}_2$, $\mathcal{O}_1$). We say that $\mathcal{IW}$ is the set of* inferred witnesses *where $\mathcal{O}_1 \models \alpha$ but $\alpha \notin \mathcal{O}_1$. If $\alpha \in \mathcal{O}_1$ then all such witnesses α are* stated witnesses *denoted by $\mathcal{SW}$.*

To generate the set of inferred witnesses $\mathcal{IW}$ and the set of stated witnesses $\mathcal{SW}$, our method transforms the axioms in the original ontology into a form similar to the axioms' form in the set of witnesses $\mathcal{W}$. This process produces the normalised ontology $\mathcal{O}^{normalised}$. Then, we determine whether or not the axioms $\alpha \in \mathcal{W}$ are stated in $\mathcal{O}^{normalised}$. If $\alpha \in \mathcal{O}^{normalised}$, it is added to $\mathcal{SW}$; if not, it is inferred and added to $\mathcal{IW}$. Note that the axioms in the witness sets are normalised because axioms within the $\mathcal{UI}$ are in normalised form.

Consider the following example, which shows that normalising the original ontology is important to determine stated and inferred witnesses.

Example 15. *Let*

$$\mathcal{O}_1 = \{A_1 \equiv P \sqcap \exists r.C_1, \}$$

and

$$\mathcal{O}_2 = \{A_1 \equiv P_2 \sqcap \exists r.C_2, \\ A_2 \sqsubseteq \exists r.C_1\}$$

and $\Sigma = \{A_1, r, C_1\}$ *be the common symbols between* $\mathcal{O}_1$ *and* $\mathcal{O}_2$*. Computing* UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ *by forgetting* A_2*,* P_2 *and* C_2 *gives the UI:*

$$\mathcal{UI}_1 = \{A_1 \sqsubseteq \exists r.C_1\}$$

Then, every axiom in $\mathcal{UI}_1$ *is checked if it is entailed by* $\mathcal{O}_2$ *or not. This results in the witness set:*

$$\mathcal{W}_1 = \{A_1 \sqsubseteq \exists r.C_1\}$$

Checking if every axiom in $\mathcal{W}_1$ *is stated in* $\mathcal{O}_1$ *will comprise an empty set for* $\mathcal{SW}$ *and* $\mathcal{W}_1$ *is considered to contain inferred axioms. However, when normalising* $\mathcal{O}_1$*:*

$$\begin{aligned}\mathcal{O}_1^{normalised} = \{&A_1 \sqsubseteq P,\\ &A_1 \sqsubseteq \exists r.C_1,\\ &P \sqcap \exists r.C_1 \sqsubseteq A_1\}\end{aligned}$$

to generate the inferred and stated witness sets, we find that the axiom in $\mathcal{W}_1$ *is a stated witness in* $\mathcal{O}_1^{normalised}$*. Thus, it is added to the stated witness set:*

$$\mathcal{SW}_1 = \{A_1 \sqsubseteq \exists r.C_1\}$$

This is the derived outcome as the ontology after normalisation states the generated witnesses.

Notably, all witnesses in $\mathcal{IW}$ are inferred as a result of the forgetting process. However, witnesses in $\mathcal{SW}$ can be produced even if no forgetting occurs during witnesses generation, more precisely during the first step of the UI-Diff method. The first step of UI-Diff computes a $\mathcal{UI}$ by forgetting symbols not in the common signature set Σ. The following example shows a case of an inferred witness resulting from the forgetting steps.

Example 16. *Let*

$$\begin{aligned}\mathcal{O}_1 = \{&A_1 \equiv P \sqcap \exists r.C,\\ &A_2 \sqsubseteq P\}\end{aligned}$$

and

$$\mathcal{O}_2 = \{A_1 \sqsubseteq \exists r.C, \\ A_2 \sqsubseteq P_2\}$$

and $\Sigma = \{A_1, r, C, A_2\}$ *be the common symbols between* $\mathcal{O}_1$ *and* $\mathcal{O}_2$*. Computing* UI-Diff$(\mathcal{O}_2, \mathcal{O}_1)$ *by forgetting* P *and* P_2 *gives the UI:*

$$\mathcal{UI}_1 = \{A_2 \sqcap \exists r.C \sqsubseteq A_1, \\ A_1 \sqsubseteq \exists r.C\}$$

This results in the witness set:

$$\mathcal{W}_1 = \{A_2 \sqcap \exists r.C \sqsubseteq A_1\}$$

and the axiom in it has been inferred by forgetting P *in* $\mathcal{O}_1$*. It is generated according to the following inference rule for concept name elimination [LLA+21a]:*

$$A_2 \sqsubseteq P, P \sqcap \exists r.C \sqsubseteq A_1 \Longrightarrow A_2 \sqcap \exists r.C \sqsubseteq A_1$$

where P *was forgotten because it is not in* Σ*. The inferred witness set is:*

$$\mathcal{IW}_1 = \{A_2 \sqcap \exists r.C \sqsubseteq A_1\},$$

because $A_2 \sqcap \exists r.C \sqsubseteq A_1$ *is not stated in* $\mathcal{O}_1^{normalised}$*.*

6.4.2 Focus or Supporting Concept Witnesses

An important point to consider when using subontologies to compute witnesses rather than the original ontologies, is how much knowledge does the subontology preserve from the original ontology. To help us understand this aspect, we categorise the resulting witnesses depending on whether they are associated with focus concepts in Σ_F or supporting concepts in Σ_S.

By using subontologies to keep track of semantic differences, all possible witnesses that are associated with focus concepts are generated. This is because subontologies preserve the semantic relationships associated with the input focus set, particularly when the input ontology is a terminology. Focus set subontologies, on the other hand,

may exclude some axioms associated with supporting concepts (see Chapter 5 about focus set subontologies). This means that some witnesses related to supporting symbols are not computed using focus set subontologies.

We define the set of focus and supporting concept witnesses as follows.

Definition 27 (Focus (Supporting) Concept Witness)**.** *Let $\mathcal{S}_1$ and $\mathcal{S}_2$ be two subontologies extracted from two $\mathcal{EL}\mathcal{H}$ terminologies $\mathcal{O}_1$ and $\mathcal{O}_2$ for a focus set Σ_F. Suppose $A \in \Sigma_F$ (Σ_S) where Σ_S is the supporting set in $\mathcal{S}$. A witness of the form $A \sqsubseteq C$ or $C \sqsubseteq A$ in* UI-Diff*($\mathcal{S}_1$, $\mathcal{S}_2$) is said to be a* focus (supporting) concept witness *where C is an $\mathcal{EL}$-concept. This witness is said to be associated with the focus (supporting) concept name A. Focus concept witnesses will be denoted by $\mathcal{W}_{\Sigma_F}$, while supporting concept witnesses will be denoted by $\mathcal{W}_{\Sigma_S}$.*

6.4.3 Analysis Scheme

The subontology generation method makes it easier for the user to distinguish between focus and supporting concept axioms (see Definition 18 about focus set axioms and Definition 20 about supporting set axioms). This distinction serves as inspiration for our witnesses segmentation mechanism, which starts by first determining if the change belongs to a focus or a supporting concept. A change to a focus concept axiom may have greater priority in the analysis than a change to a supporting concept axiom. Thus, differentiating between a difference belonging to a focus or supporting concept can considerably aid the analysis process.

Additionally, as discussed with a SNOMED CT terminologist, there are instances where the emphasis is on revealing variations affecting the subsumption relationships between supporting symbols, which the current toolings do not provide [AGS]. In Section 6.6 (Scenario 2), we illustrate two of these cases by demonstrating the use of the supporting set witnesses as a result of segmenting the witness set. This indicates that such segmentation permits straightforward analysis of those witnesses.

Our witness analysis scheme is depicted in Figure 6.2. The analysis process begins by determining if a witness is a focus or a supporting concept witness (Definition 27). This results in two sets: $\mathcal{W}_{\Sigma_F}$, which denotes the witnesses for the focus set, and $\mathcal{W}_{\Sigma_S}$, which denotes the witnesses for the supporting set. Each of the focus and supporting concept witnesses is further partitioned into two groups based on whether the witness is a stated or an inferred one (see Definition 26). This results in four sets, which are $\mathcal{SW}_{\Sigma_F}$, $\mathcal{SW}_{\Sigma_S}$, $\mathcal{IW}_{\Sigma_F}$ and $\mathcal{IW}_{\Sigma_S}$. Each of these four sets is subdivided according to

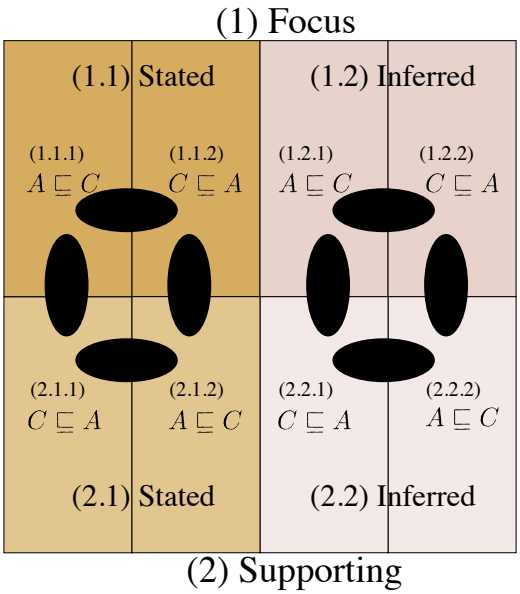

Figure 6.2: Witnesses Segmentation Scheme

the two axioms forms: $A \sqsubseteq C$ and $C \sqsubseteq A$, where A is a concept in either Σ_F or Σ_S and C is an $\mathcal{EL}$-concept. The total resulting sets is eight sets.

Additionally, we illustrate the possible intersections of the generated sets. This intersection happens as a result of segmenting the witnesses sets into focus and supporting concept witnesses, where the left and right hand-sides of axioms might be either a focus or supporting concepts. The black oval shape in Figure 6.2 indicates where there may be non-empty intersections, resulting in the sets: (1.1.1 $\cap$ 2.1.1), (1.1.2 $\cap$ 2.1.2), (1.2.1 $\cap$ 2.2.1) and (1.2.2 $\cap$ 2.2.2).

6.5 Evaluation of Focus Set Semantic Difference

This section evaluates our method to generating focus set semantic differences in depth. First, we describe the corpus used in our experiments (Section 6.5.1). Second, we describe the different experiments conducted (Section 6.5.2). Thirdly, we discuss our results from each experiment (Sections 6.5.3–6.5.6).

6.5.1 Corpus

We evaluate our approach with SNOMED CT. Concepts within SNOMED CT are organised into 19 concept hierarchies such as *Procedure*, *Clinical Finding*, *Body structure* and *Organism* (see Section 2.4 about SNOMED CT).[1]

We focused on evaluating our approach against early versions of SNOMED CT as recent versions include an extended fragment of $\mathcal{ELH}$, which our method cannot handle. This extended fragment includes GCIs of the form $C \sqsubseteq A$, transitive roles, reflexive roles, and role chain axioms.

As mentioned in Section 2.4, SNOMED International maintains a collection of standard reference sets that were created for a variety of purposes, including data entry and electronic health records uses.[2] These refsets include the General Practice/Family Practice (GPFP), the International Classification of Nursing Practice Diagnoses (ICNP-Diagnoses), the International Classification of Nursing Practice Interventions (ICNP-Interventions) and the General Dentistry refset.

We will now list the reference sets that were used in this evaluation. (See Section 2.4.2 about reference sets).

General Practice/Family Practitioner (GPFP) refset This refset contains SNOMED CT concepts for two data types that are frequently used in electronic health records for general/family practise, which are Reasons For Encounter (RFEs) and (ii) health concerns [Intn]. Concepts in this refset belong to four main concept hierarchies in SNOMED CT, namely *Clinical finding*, *Event*, *Procedure* and *Situation with explicit context*. We analysed the number of concepts in each version of this refset to determine the proportion of concepts that fell into each of these four primary concept hierarchies. Our analysis demonstrates that one of the concepts in the GPFP refset is under the *Body structure* concept hierarchy in refset versions January 2016 to July 2017 (as shown in Figure A.3 in Appendix A).

The International Classification of Nursing Practice Diagnoses (ICNP Diagnoses) refset The ICNP Diagnoses refset[3] contains concepts that correspond to those in

[1] Full list of SNOMED CT concept hierarchies can be browsed through the ontology's official browser: `https://browser.ihtsdotools.org/`

[2] `https://mlds.ihtsdotools.org/#/ihtsdoReleases`

[3] `https://www.icn.ch/what-we-do/projects/ehealth-icnptm/about-icnp`

the nursing diagnostics (problems) refset, which was intended to improve terminology standardisation and interoperability in health information systems for nursing diagnoses [Into]. Concepts contained in this refset fall into two main SNOMED CT concept hierarchies: *Clinical finding* and *Situation with explicit context*. Figure A.4 in Appendix A illustrates the number of concepts in January 2016 to July 2017 versions that belong to these main concept hierarchies.

The International Classification of Nursing Practice Interventions (ICNP Interventions) refset The ICNP-Interventions refset functions similarly to the ICNP-Diagnosis refset [Intp]. As documented in [Intp], the concepts contained in this refset are all members of the SNOMED CT *Procedure* hierarchy. Between January 2016 and July 2017 (Figure A.5 in Appendix A), 4% of the total number of concepts in this refset were classified as belonging to the *Situation with explicit context* concept hierarchy.

General Dentistry (GD) refset The General Dentistry diagnostic term set is intended to provide dentistry with terms that span the great majority of dental care [Intq].

We used our method for producing focus set semantic differences (described in Section 6.2) to compute witness sets for each refset of SNOMED CT. We computed witness sets for the GPFP, ICNP, and ICNP-Interventions refsets using the SNOMED CT versions 1601 until 1707, where 1601 refers to the January 2016 version and the rest of the versions are similarly situated. For each refset, the resulting witness sets belonged to six pairs of UI-Diff comparisons (as seen in Figure 6.3), representing information gained and lost.

To compute witness sets representing gained and lost information for the General Dentistry refset, we used the SNOMED CT versions from 1701 (January 2017) to 2007 (July 2020). The output of witness sets belonged to 16 pairs of UI-Diff comparisons (as seen in Table 6.1). The number of UI-Diff comparisons computed for the GD refset is more than the other three refsets because the analysis in Section 6.5.4 required more comparisons of later versions to gain a general observation about the modifications that occurred to the GD part of SNOMED CT.

6.5.2 The Conducted Experiments

We performed the following experiments after generating the witness sets.

- **Comparing the size of the witness sets (see Section 6.5.3)**

We computed the size of witness sets of the three refsets including GPFP, ICNP and ICNP-Interventions. The results in [LLA+21a] (presented in Figure 4 [LLA+21a]) included the size of the witness sets when computing UI-Diff between 15 consecutive versions of the core editions of SNOMED CT. We show the size of the witness sets of the versions January 2016 to July 2017 from the results in [LLA+21a] (shown in Yellow bar in Figure 6.3), and compared them to the witness sets' sizes of the three refsets; GPFP, ICNP and ICNP-Interventions.

- **Evaluating the usefulness of the UI-Diff witnesses (see Section 6.5.4)**

  We analysed the witness sets belonging to the GD refset to evaluate the usefulness of the witnesses generated using the UI-Diff method. This was accomplished in order to have a better understanding of certain general observations about modelling changes to the GD subontology of SNOMED CT.

- **Analysis of the witness sets (see Section 6.5.5)**

  We segmented the witness sets according to the analysis scheme presented in Section 6.4. This analysis was conducted using three refsets: GPFP, ICNP, and ICNP-Interventions. We computed the size of the witness sets after segmentation in order to get the number of focus and supporting concept witnesses that are either stated or inferred.

- **Analysis of the top-level concept hierarchy of focus concept witnesses (see Section 6.5.6)**

  Given that our analysis scheme distinguishes between witnesses associated with focus concepts and those associated with supporting concepts, we performed an additional analysis on the resulting focus concept witnesses $\mathcal{W}_{\Sigma_F}$ to determine which SNOMED CT top-level concept hierarchy subsumes the focus concepts A in the set of axioms $A \sqsubseteq C$ and $C \sqsubseteq A$. Witness sets of the GPFP, ICNP, and ICNP-Interventions refsets of the corpus were used in this analysis.

In the following sections, we discuss the results of our experiments.

6.5.3 Comparing the Size of the Witness Sets

Figure 6.3 shows the number of witnesses in different comparisons of consecutive versions of SNOMED CT. The Yellow bar indicates the witness sets from [LLA+21a]. As mentioned, the evaluation in [LLA+21a] shows the number of witnesses when tracking

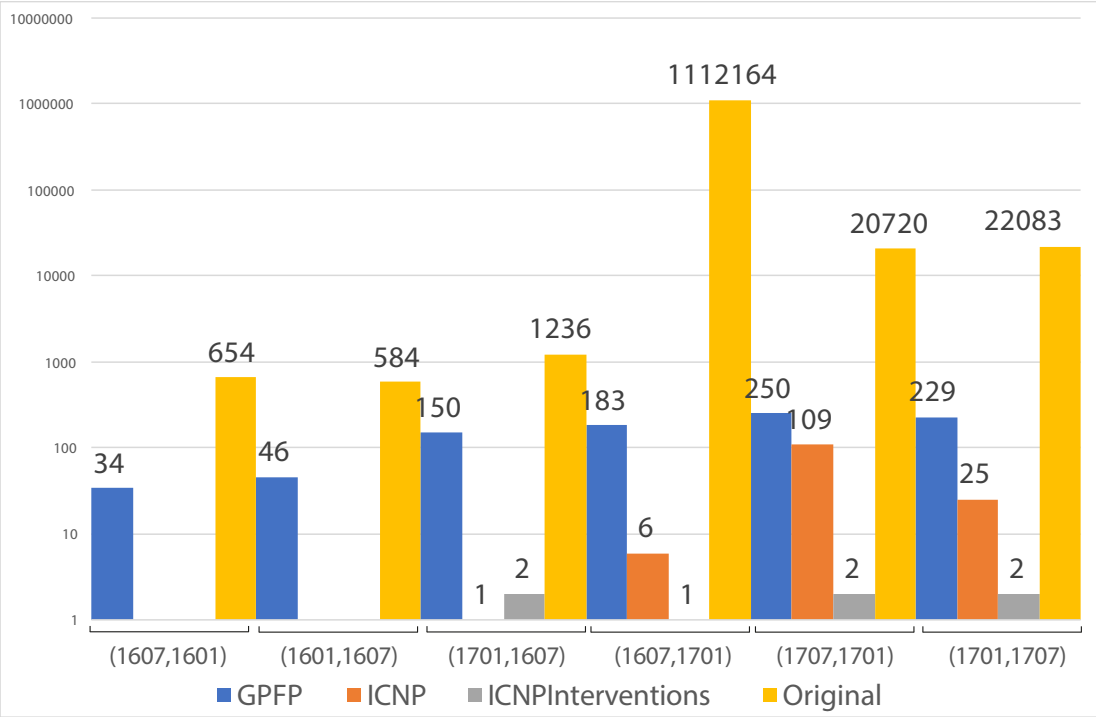

Figure 6.3: Number of witnesses in different comparisons of consecutive versions of computed subontologies for the refsets: GPFP, ICNP and ICNP Interventions of SNOMED CT

semantic differences between the core editions of SNOMED CT. On the other hand, our results demonstrate the number of semantic differences between the subontologies computed for the three refsets, GPFP, ICNP, and ICNP-Interventions. The number of witnesses computed between SNOMED CT's core editions is enormous in comparison to the witnesses computed for the three refsets. This is expected, as tracking semantic differences between subontologies computed for the refsets results in a focused set of witnesses with a smaller size. For example, the $\mathsf{UI\text{-}Diff}(1601, 1607)$ comparison of the *GPFP refset* includes only 46 witnesses, compared to 584 witnesses in the original comparison between the whole SNOMED CT versions. This is also true for the remaining refsets; for example, the witnesses belonging to *ICNP refset* in the comparison $\mathsf{UI\text{-}Diff}(1607, 1701)$ are only a few out of over a million witnesses in the original comparison.

Comparing the number of witnesses across all refsets shows that the majority of semantic differences belong to the GPFP subontology of SNOMED CT. This shows that terminologists focused their modifications on the GPFP portions of the ontology.

Table 6.1: GD witnesses of forms: $GD\_concept \sqsubseteq C$ (Form I), $C \sqsubseteq GD\_concept$ (Form II), $GD\_concept \equiv C$ (Form III)

Tasks	Form I	Form II	Form III	Tasks	Form I	Form II	Form III
(1701,1707)	4	0	0	(1707,1701)	0	4	4
(1707,1801)	3	0	0	(1801,1707)	0	0	0
(1801,1807)	8	4	0	(1807,1801)	13	1	2
(1807,1901)	15	3	0	(1901,1807)	6	4	4
(1901,1907)	17	5	0	(1907,1901)	9	4	6
(1907,2001)	10	5	0	(2001,1907)	11	3	3
(2001,2003)	0	0	0	(2003,2001)	0	0	0
(2003,2007)	6	13	0	(2007,2003)	4	14	4

6.5.4 Evaluating the Usefulness of the UI-Diff Witnesses

In this section, we explore the semantic differences between several versions of the SNOMED CT General Dentistry reference set and attempt to deduce what they mean. The purpose of this study is to explore how UI-based semantic differences can aid SNOMED International's modellers in comprehending the evolution of the general dentistry domain.

We discovered that a number of defined concepts were revised in a similar manner: the prior versions entailed the downward directions of the GCI of the defined concepts belonging to the GD refset, but not the upward directions. For example, in the comparison between the subontologies 1701 and 1707 we found that the previous version entailed the downward direction of the witness $C \sqsubseteq$ *Exostosis of jaw (disorder)* of the defined concept *Exostosis of jaw (disorder)* but not the upward direction, where C was a complex concept. This was because the ontology engineers considered a more precise location for *Exostosis of jaw (disorder)*, which was *Bone structure of jaw (body structure)* in the later version. The original was *Jaw region structure (body structure)*, which is a super concept of *Bone structure of jaw (body structure)*.

Clearly, the ontology engineers at SNOMED International have updated the definitions of clinical terms in subsequent versions by making concepts more explicit and the definitions more precise.

We computed statistics of the GD set witnesses determining whether the change occurred to the necessary or sufficient conditions of the GD concepts. Table 6.1 summarises witness counts in the following formats: $GD\_concept \sqsubseteq C$ (Form I), $C \sqsubseteq GD\_concept$ (Form II) and $GD\_concept \equiv C$ (Form III) where C is a complex expression and $GD\_concept$ is a named concept in the focus set (GD set). The results indicate that the majority of changes occurred to the necessary conditions of the GD

Table 6.2: Total focus and supporting set witnesses count table

Refset	Comparison	$\mathcal{IW}_{\Sigma_F}$	$\mathcal{IW}_{\Sigma_S}$	$\mathcal{IW}_{\Sigma_F} \cap \mathcal{IW}_{\Sigma_S}$	$\mathcal{SW}_{\Sigma_F}$	$\mathcal{SW}_{\Sigma_S}$	$\mathcal{SW}_{\Sigma_F} \cap \mathcal{SW}_{\Sigma_S}$
GPFP	(1607,1601)	11	0	0	21	4	2
	(1601,1607)	4	1	0	35	6	4
	(1701,1607)	20	2	0	124	17	13
	(1607,1701)	17	3	0	157	9	3
	(1707,1701)	90	10	3	148	10	5
	(1701,1707)	51	4	2	157	30	11
ICNP-Diagnosis	(1607,1601)	0	0	0	0	0	0
	(1601,1607)	0	0	0	0	0	0
	(1701,1607)	1	0	0	0	0	0
	(1607,1701)	2	0	0	4	0	0
	(1707,1701)	0	6	0	102	1	0
	(1701,1707)	21	1	0	3	0	0
ICNP-Interventions	(1607,1601)	0	0	0	0	0	0
	(1601,1607)	0	0	0	0	0	0
	(1701,1607)	2	0	0	0	0	0
	(1607,1701)	0	0	0	1	1	1
	(1707,1701)	0	0	0	1	1	0
	(1701,1707)	0	1	0	1	0	0

refset's focus concepts (Form I).

6.5.5 Witnesses Analysis

Table 6.2 includes statistics on the overall number of focus and supporting set witnesses when tracking semantic difference between different versions of SNOMED CT belonging to the three refsets: GPFP, ICNP and ICNP-Interventions. These witnesses are partitioned according to the scheme outlined in Section 6.4. In each of the three refsets, the witnesses associated with the focus concepts outnumber those associated with the supporting concepts in most of the comparisons. This indicates that there have been a significant number of changes to the definitions of the focus concepts over the course of the version releases.

When comparing the January and July 2017 versions ($\mathsf{UI\text{-}Diff}(1707, 1701)$ of *ICNP-Diagnosis refset*, a relatively higher number (109) of witnesses appeared, indicating lost information. Seven of these witnesses are associated with supporting concepts; only one of them is a stated witness, while the others are inferred. On the other hand, there is a lesser number (26) of witnesses representing gained information was entailed by July 2017 but was no longer entailed by January 2017. Most of these witnesses (23) were inferred and not explicitly stated in the *ICNP-Diagnosis* Subontology-1707. These inferred witnesses point to hidden changes that deserves further investigation.

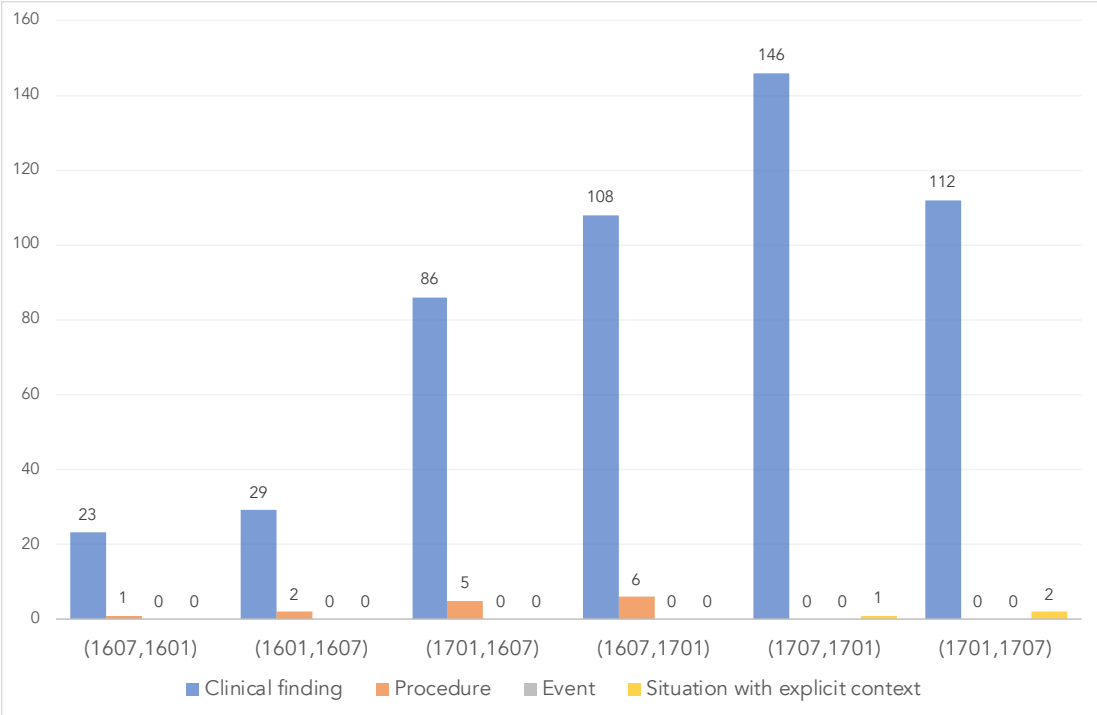

Figure 6.4: Number of focus concept witnesses belonging to four main concept hierarchies of SNOMED CT in the GPFP comparisons

The changes associated with the *ICNP-Interventions* are very subtle, and the comparison $\mathsf{UI\text{-}Diff}(1707, 1701)$ appeared to contain only one inferred focus concept witness. Such witness may be worth examining, as it may indicate unexpected consequence as a result of modifying the ontology.

There are no focus, nor supporting concept witnesses in the comparisons between the January 2016 and July 2016 versions of the ICNP Diagnosis and ICNP Interventions refsets. This likewise holds true in the opposite direction (from July 2016 version to January 2016 version).

6.5.6 Focus Concept Witnesses' Top-level Concept Hierarchy

Figure 6.4 shows the number of focus concept witnesses that belong to the top-level concept hierarchies in the comparisons of the GPFP refset. We found that the majority of focus concept witnesses in all comparisons of the GPFP refset fall under the *Clinical finding* concept hierarchy. The results indicate that not all focus concepts belonging to the four top-level concept hierarchies (*Clinical finding*, *Event*, *Procedure* and *Situation*

with explicit context) to which concepts in the GPFP refset belong have associated focus concept witnesses. For instance, there are no focus concept witnesses exist in the comparison UI-Diff$(1601, 1607)$ whose focus concepts reside in the *Situation with explicit context* concept hierarchy. The terminologist can conclude from the results that changes associated with the GPFP refset between these two versions (1601 and 1607) did not affect the *Situation with explicit context* concept hierarchy.

The figures for the top-level concept hierarchies of the focus concept witnesses that belong to the ICNP-Diagnosis and ICNP-Interventions refsets are shown in Appendix A (Figures A.1 and A.2). Although concepts in the ICNP-Diagnosis refset belong to two main concepts hierarchies, which are *Clinical finding* and *Situation with explicit context* (See Figure A.1), witnesses associated with focus concepts only belong to *Clinical finding* concept hierarchy. A very small number of focus set witnesses belong to ICNP-Interventions (Figure A.2).

6.6 Practical Scenarios

This section discusses various scenarios that demonstrate how the modeller can benefit from the various witness sets. To yield these scenarios, we examined several examples from the results of our evaluation (in Section 6.5).

Scenario 1

Assume the modeller is interested in knowing about changes associated with the GPFP focus set, more specifically, changes to the definition of the focus concept ***Vocal cord palsy (disorder)*** in the GPFP comparison (1707,1701). In particular, the modeller wishes to verify whether the definition of ***Vocal cord palsy (disease)*** has undergone changes since the prior version 1701 that might render its previous definition invalid in the subsequent version 1707. To begin the analysis, the modeller examines focus concept witnesses $\mathcal{W}_{\Sigma_F}$ since the concept of interest (***Vocal cord palsy (disorder)***) is a focus concept. Such set contains three witnesses associated with ***Vocal cord palsy (disorder)***. Assume now the modeller wishes to determine which of the ***Vocal cord palsy (disorder)*** witnesses are stated and which are inferred. The modeller can inspect two sets: the inferred focus concept witnesses $\mathcal{IW}_{\Sigma_F}$ and the stated focus concept witnesses $\mathcal{SW}_{\Sigma_F}$. By inspecting the set $\mathcal{IW}_{\Sigma_F}$, the modeller discovers two inferred

witnesses associated with the concept ***Vocal cord palsy (disorder)***:

$$\textbf{\textit{Vocal cord palsy (disorder)}} \sqsubseteq \textit{Disorder of respiratory system (disorder)} \tag{6.10}$$

$$\textbf{\textit{Vocal cord palsy (disorder)}} \sqsubseteq \textit{Paralytic syndrome (disorder)} \tag{6.11}$$

Inferred witnesses represent entailments that are not stated in the ontology. Deriving the reason for the inferred witnesses can be done by using an OWL justification tool [HPS15] or by manual inspection. Using the OWL justification tool, we illustrate the causes of inferred witnesses (see Section 2.3 for a definition about justification axioms). Two inputs are considered when using the justification tool: the inferred witness set, and the subontology-1701. The set of axioms that explain the inferred witness (6.10) is:

$$\textbf{\textit{Vocal cord palsy}} \sqsubseteq \textit{Paralysis of vocal cords or larynx} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of nervous system}) \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Vocal cord structure}) \tag{6.12}$$
$$\textit{Paralysis of vocal cords or larynx} \sqsubseteq \textit{Disorder of respiratory system} \tag{6.13}$$

We can see that ***Vocal cord palsy (disorder)*** is subsumed by *Disorder of respiratory system (disorder)* through the concept *Paralysis of vocal cords or larynx (disorder)*. This subsumption relation was revealed by forgetting the concept *Paralysis of vocal cords or larynx (disorder)* resulting in the above witness (6.10).

Generating the explanation axioms for the second inferred witness (6.11), gives the following set of axioms:

$$\textbf{\textit{Vocal cord palsy (disorder)}} \sqsubseteq \textit{Paralysis of vocal cords or larynx (disorder)} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of nervous system (body structure))}$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Vocal cord structure (body structure))} \tag{6.14}$$
$$\textit{Paralysis of vocal cords or larynx (disorder)} \sqsubseteq \textit{Paralytic syndrome (disorder)} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Structure of nervous system (body structure))} \sqcap$$
$$\exists \textit{Role group.}(\exists \textit{Finding site.Vocal cord structure (body structure))} \tag{6.15}$$

The explanation axioms shows that the forgotten concept *Paralysis of vocal cords or larynx* is subsumed by the concept *Paralytic syndrome* resulting in the second inferred witness (6.11).

Upon generating the explanation axioms for the inferred witnesses, several actions can be done by the modeller, including logging the witness set along with their explanation axioms for future improvements of the ontology.

Inspecting the stated focus concept witnesses set $\mathcal{SW}_{\Sigma_F}$ revealed one stated witness associated with the concept ***Vocal cord palsy (disorder)*** which is:

$$\textbf{\textit{Vocal cord palsy (disorder)}} \sqsubseteq \exists \textit{Role group.}(\exists \text{Finding site.Structure of nervous system (body structure)}) \quad (6.16)$$

It is worth noting that the stated witness of ***Vocal cord palsy (disorder)*** (6.16) is a result of abstracting the focus concept definition. (See Section 6.3 to understand the first aspect of the focus set semantic difference generator method). Abstracting the definition of ***Vocal cord palsy*** helps understanding which necessary condition that ***Vocal cord palsy*** inherits from *Paralysis of vocal cords or larynx* that is not entailed by the subontology version 1707. Computing the semantic difference between the original ontologies rather than the subontologies, on the other hand, does not reveal the witness (6.16), which may result in an incomplete understanding of all the changes to the definition of the focus concept ***Vocal cord palsy***.

Scenario 2

Reviewing modifications to the concept hierarchy is one of the version control duties undertaken by SNOMED CT modellers. Such changes become significant when dealing with patient case queries as they may affect the "is a" relationships, not the actual patient cases, which can generate inconsistencies in reporting trends [LCL11, LdKLC14]. Following a discussion with a SNOMED CT terminologist [AGS], there are instances when modellers are particularly interested in knowing about changes to the hierarchy of supporting concepts. Particularly, in the case where supporting concepts occur in definitions of focus concepts that remain unchanged between versions. To demonstrate such instances, we present two examples from the evaluation results.

Example 1 We assume that the modeller wishes to investigate for a change in the supporting concepts hierarchy among the witnesses specified in ICNP-Interventions $\mathsf{UI\text{-}Diff}(1707, 1701)$.

The first step is to determine whether there are any supporting concept witnesses in $\mathcal{W}_{\Sigma_S}$. The supporting concept witnesses in ICNP-Interventions $\mathsf{UI\text{-}Diff}(1707, 1701)$

do not include any inferred witnesses (as seen in Table 6.2). Thus, the modeller inspects the collection of $\mathcal{SW}_{\Sigma_S}$ (stated supporting concept witnesses). Here, the modeller discovers a single witness:

$$\underline{\textit{Finding related to ability to walk}} \sqsubseteq \textit{Finding of activity of daily living} \tag{6.17}$$

<u>*Finding related to ability to walk*</u> is a supporting concept that exist in the definition of the focus concept ***Assessment of ability to walk***. This focus concept is defined identically in both subontologies versions 1701 and 1707, by:

$$\begin{aligned}&\textbf{\textit{Assessment of ability to walk}} \sqsubseteq \textit{Procedure} \sqcap \\ &\exists \textit{RoleGroup.}(\exists \textit{Method.Evaluation - action}) \sqcap \\ &\exists \textit{RoleGroup.}(\exists \textit{Has focus.}\underline{\textit{Finding related to ability to walk}})\end{aligned} \tag{6.18}$$

The modeller learns from the change (6.17) that the position of the supporting concept (<u>*Finding related to ability to walk*</u>) in the concept hierarchy has shifted. This is demonstrated by the axiom (6.17), which is entailed by the 1701 version but not by the 1707 version. To understand the new position of the supporting concept (*Finding related to ability to walk*), the modeller checks the later version (subontology-1707) and finds that it has been changed to be subsumed by *Clinical finding* in the subontology-1707.

$$\underline{\textit{Finding related to ability to walk}} \sqsubseteq \textit{Clinical finding} \tag{6.19}$$

It is worth noting that the concept <u>*Finding related to ability to walk*</u> is located beneath the concept *Finding related to ability to move* in the concept hierarchy of the original ontology-1707. The reason that the concept <u>*Finding related to ability to walk*</u> was not discovered to be subsumed by the concept *Finding related to ability to move* in the subontology-1707 is that *Finding related to ability to move* is neither a focus nor a supporting concept in the subontology-1707.

Computing UI-Diff between the $\bot$-modules does not reveal the witness (6.17) associated with the supporting concept <u>*Finding related to ability to walk*</u>. This was checked by performing a spot check which involves computing the bottom modules for Σ_F as we did not have access to the results of the UI-Diff comparisons between the original ontologies. Then, we computed the UI-Diff between the generated $\bot$-modules to determine whether computing the UI-Diff between the original ontologies (which

are represented by the $\perp$-modules) would reveal the witness associated with the supporting concept *Finding related to ability to walk*. Our test showed no such witness, confirming that computing the UI-Diff using focus set subnotologies rather than the original ontologies generates changes associated with supporting concepts that cannot be revealed using other methods.

Example 2 Assume the modeller is interested in learning about changes to the hierarchy of supporting concepts within the GPFP$(1601, 1607)$ comparison. The modeller begins by scanning the supporting set witnesses $\mathcal{W}_{\Sigma_S}$. If the modeller is looking for changes stated explicitly in the subontology-1607, the modeller examines the set $\mathcal{SW}_{\Sigma_S}$. This set has six witnesses, where one of them represents a change to the supporting concept hierarchy and their focus concept definitions remain the same in both versions (witness 6.20).

$$\textit{Malunion of fracture} \sqsubseteq \textit{Fracture} \tag{6.20}$$

Computing the UI-Diff between the original ontologies of the SNOMED CT versions 1601 and 1607 did not reveal any change associated with *Malunion of fracture*. This is because the concept *Malunion of fracture* is subsumed by the same conditions in both versions, which are *Abnormal healing of fracture* and *Fracture*. However, the UI-Diff computed between the original ontologies detected a change associated with the parent concept *Abnormal healing of fracture*. The concept *Abnormal healing of fracture* in the version 1607 is subsumed by *Healing fracture* which is not the case in the version 1601, i.e., *Abnormal healing of fracture* is not subsumed by *Healing fracture* in the previous version (1601). Since *Healing fracture* is subsumed by *Fracture*, then by transitivity of the subsumption relation through *Abnormal healing of fracture*, *Malunion of fracture* becomes subsumed by *Fracture*.

To ascertain the position of *Malunion of fracture* in the previous version, the modeller examines the subontology-1601. The modeller finds that *Malunion of fracture* was subsumed by *Lesion* in the previous version (1601).

Both Examples 1 and 2 demonstrate the critical role of subontologies in identifying specific changes associated with supporting concepts that are related to the definitions of focus concepts. This is because the information within subontologies is focused only on the definitions of concepts of interest (the focus concepts) and the hierarchical relationships of their supporting concepts.

6.7 Conclusion

We introduced a method for tracking semantic differences between ontologies. The aim of the method is to limit the tracking of semantic changes to a portion of the ontology. This is accomplished by first extracting subontologies for a given collection of focus symbols, which are then utilised to compute semantic differences. This leads in a focused collection of semantic changes that are pertinent to the selected sub-domains of the original ontology.

Our approach, which begins by extracting focus set subontologies from the source ontology, involves generating abstracted definitions of focus concepts. These abstracted definitions assist in stating all of the defining characteristics of focus concepts in the form of existential restrictions that might not be expressed in the original definition of the focus concept. These defining characteristics aid in the identification of additional witnesses linked with focus concepts that would not be generated without abstraction. These additional witnesses inform modellers to probable changes in the meaning of concepts as a result of adding or removing defining characteristics from the definition of focus concepts during release revisions.

Our approach was evaluated by comparing semantic differences between subontologies of standard reference sets that are actively used in electronic health record systems. Our evaluation highlights the value of our method for tracking focus set semantic differences by facilitating the examination of semantic differences arising from very large ontologies such as SNOMED CT especially when only a subset (e.g., GPFP, ICNP-Diagnosis or ICNP-Interventions) of the original ontology's domain is important for the application and users should be unconcerned about changes to the remainder of the ontology. We discovered that semantic differences tend to be subtle between subontologies for specific focus sets (reference sets). For instance, we found that very small number of semantic differences occur between the July 2016 and July 2017 versions of the ICNP-Diagnosis and ICNP-Interventions refsets.

The segmentation of witnesses demonstrates our method's usefulness at simplifying analysis tasks when the modeller is primarily interested in identifying differences relating to focus concepts or supporting ones. Moreover, the distinction between stated and inferred witnesses aids the modeller in capturing unobserved inferred witnesses for further investigation. This demonstrates the method's utility in identifying obscure witnesses associated with a particular refset.

The concrete scenarios illustrate how our method can be used to track and analyse changes between subontologies. We examined two scenarios in which the modeller

could be interested in tracking changes to either focus concepts (Scenario 1) or supporting concepts (Scenario 2). Both scenarios illustrated the ability of our method for identifying changes that are otherwise undetectable by existing methods; in particular, neither the SNOMED CT versioning technique nor the use of the UI-Diff method directly between original ontologies reveal the type of witnesses investigated in the scenarios.

Chapter 7

Conclusions and Future Work

With the emergence of new ontologies in the biomedical and life sciences domains and the continued growth and complexity of current ones, there is an increasing need to utilise, develop, and expand ontology extraction formalisms for a range of applications. Ontology extraction methods have been developed with the objective of containing only the terms that are of interest to the developer or end-user. Although ontology modules have poor precision rates and current forgetting tools have difficulty when applied directly to huge ontologies for small signature sizes, in Chapter 4, we developed a workflow that successfully couples the two approaches. The results show a practical approach with high precision is provided by our workflow for computing uniform interpolants for real-world, small-sized signatures of prominent, large ontologies including SNOMED CT and NCIt.

Our study in Chapter 4 involved assessing the computation of uniform interpolants for real-world signatures such as SNOMED CT reference sets through the developed workflow. The workflow for computing uniform interpolants incorporates preprocessing methods, including newly developed signature extension and partitioning techniques, and existing ontology modularity tools. The signature extension algorithm was used to augment the interpolation signature by including required symbols from the input ontology in order to compute complete definitions using uniform interpolation. While the generated uniform interpolants were valuable in this respect, when utilising the LETHE and UI-FAME tools, they contained rewritten axioms with overly expressive constructors in comparison to the original ontology. While this study did not meet the required usability criteria that SNOMED CT users would seek, it served as a useful prelude to the development of the subontology extraction and semantic difference methods.

In Chapter 5, we investigated an alternative method to generating subontologies, namely a definition-driven approach for producing subontologies that are focused on the given set of symbols. Our approach for generating subontologies is very useful for SNOMED CT users. The subontologies adhere to a set of preset criteria that make them suitable for SNOMED CT users. Among these criteria is the provision of complete definitions for the given set of symbols from the original ontology. The second requirement is for the created subontologies to be concise. The third assures that the subontology's axioms have the same structure as those in the original ontology.

Our subontologies contain axioms expressed in the long canonical form, which benefits the SNOMED community in a variety of ways. By modelling concept definitions in this form, a reasoner can infer concept names that are either superconcepts or subconcepts of the given concept more precisely. Additionally, the long canonical form decomposes a concept's conditions maximally into its primitive concepts and defining characteristics, which is advantageous for recording, retrieval or analysis of stored clinical data [Spa01].

Our method for generating focus set subontologies is novel in the field of ontology extraction. Ontology modularity methods, such as semantic and minimal subsumption modularisation, aim to generate extracts specified by the set of input symbols. This also holds true for uniform interpolation. Rather than that, our method is a definition-driven approach, where the subontology defines the set of input symbols by extracting their corresponding definitions from the original ontology. The generation of subontologies in this way guaranteed that the original structure of axioms is preserved in the resulting subontologies, which is critical for SNOMED CT users.

It is worth noting that our method for generating subontologies aims to encompass all axioms that are essential solely for the definitions of focus concepts. This way, we trade the *completeness* of providing all of the information about the supporting symbols used in the definitions of the focus set's symbols for *conciseness*. This came with the assumption that the user is not interested in all of the information about the supporting symbols. Allowing for the inclusion of all information pertaining to supporting symbols in the generated subontologies may result in a very large ontology similar to those obtained through locality-based modularisation [GHKS07]. On the other hand, if we achieve completeness by computing only axioms about supporting symbols, such that all of the supporting symbols' entailments are preserved in the subontology with the exclusion of any symbol outside the range of the supporting symbols, we may achieve this only through rewriting, in which case we will end up with axioms that are

rewritten similar to those obtained through uniform interpolation [LW11].

Our subontology generation method is based on computing abstracted definitions for the input set of focus symbols. We studied the impact of abstracted definitions on reducing the size of the generated subontology. To achieve that, we introduced the notion of interval, which allowed us to clearly explain why subontologies built on abstracted definitions are smaller than bottom locality-based modules while retaining all of the focus concept's defining characteristics. The results of computing the interval set on the bottom modules showed that a significant number of concept definitions that are deemed unnecessary because they carry the same information as the abstracted definitions inherit can be ignored as a result of abstraction.

In Chapter 6, we investigated the feasibility of tracking semantic differences between retrieved subontologies for focus sets by developing a method for generating semantic differences that are domain-specific to the original ontology. The objective is to constrain the generation of witness sets depending on the user-specified symbols. Our approach begins with the extraction of the abstracted definition focus set subontologies from the source ontology. The generated subontologies was done by computing abstracted definitions which include all of the focus concept's defining characteristics. As a result, abstracting focus concept definitions reveal additional differences about the focus concept, where conditions that were not explicitly stated in the definition of the focus concept prior to abstraction are regarded as distinctions that the old (new) version of the ontology may not include. The provision of such additional witnesses assists modellers to identify possible changes in the meaning of concepts as a result of addition or deletion of defining characteristics from the definition of focus concepts during release revisions.

Our evaluation (Section 6.5) illustrates the use of our method to generating focus set semantic differences by demonstrating how it enables the identification and analysis of differences of different versions of very large ontologies such as SNOMED CT. For example, our method allowed us to reveal semantic distinctions between the SNOMED CT parts represented by the ICNP Diagnosis and ICNP Interventions refsets that did not exist in previous versions of the ontology (January to July of 2016 and vice versa). Additionally, there are a few semantic differences between the SNOMED CT subontologies of the ICNP Diagnosis and ICNP Interventions refsets as of July 2016 and July 2017. These findings illustrate that extracting subontologies for input sets of focus concepts in order to track differences between them can aid in decreasing the effort required to analyse the very large number of witnesses that tend to be

generated when comparing the original ontologies. This is particularly advantageous when only a portion of the original ontology's domain (e.g., GPFP, ICNP-Diagnosis, or ICNP-Interventions) is required for the application and users are unconcerned about modifications to the remainder of the ontology.

Potential Applications

One of the main contributions of this thesis is a new notion of subontologies and an automated method to compute them. Our subontologies are intended to aid SNOMED CT users in extracting subontologies for subdomains of the original ontology while keeping the model and semantics of the original ontology. Envisaged applications include combining multiple subontologies relevant to a particular application into a single ontology. For example, all medicinal products can be retrieved from the various SNOMED CT extensions such as the UK, US, Canada, and Australia and integrated into a single ontology that encompasses the entire range of medicinal products. Another topic of study is the development and distribution of terminology. For instance, if particular content from a national extension such as the US is relevant for the UK edition, it can be extracted as a subontology and imported as part of the UK edition, which provides greater flexibility and simplifies content management. Furthermore, content development of the international edition might be done by extracting subontologies from extensions and importing it into the international (core) edition.

Because changes can have an impact on how SNOMED CT is used within an organisation's electronic patient record (EPR) systems, it is critical for an organisation to analyse and comprehend changes made in each SNOMED CT release, particularly those that affect concepts included in the organisation's existing refsets [LCL11]. Our focus set semantic difference generation method enables the inspection of such alterations. Among the numerous changes that the SNOMED CT in particular is vulnerable to are those that affect the concept's status (as defined or primitive), defining characteristics (existential restrictions), normal forms, and a concept's location within the ontology's concept hierarchy. Our method permits the inspection of such changes because it is applied to subontologies that meet the SNOMED CT modelling guidelines. Additionally, because our method is built on uniform interpolation for detecting semantic differences between ontology versions, it can reveal implicit entailments that a particular subdomain's version may not entail.

7.1 Limitations and Future Work

In terms of the subontology generation method, the following limitations should be considered in the future.

- The method is currently limited to $\mathcal{ELH}$ ontologies that lack GCI axioms of the form $C \sqsubseteq D$ or $C \sqsubseteq A$ where C and D are both complex concepts and A is a concept name. Constructing the normal forms when A additionally has a GCI axiom of the form $C \sqsubseteq A$ is a different issue that requires in-depth investigation to ensure that the complete definition of A is captured by abstraction.

- To cover the full expressivity of different versions of SNOMED CT that are expressed in $\mathcal{ELH}$ additionally with role chains, transitive roles, and reflexive roles it would be beneficial to enhance the method.

- While the method was capable of producing subontologies from ontologies as large as SNOMED CT, the extraction process can be optimised further. Specifically, the procedure of removing transitive closure axioms is currently performed in a way that can be further improved.

- As demonstrated in Chapter 5, when the input ontology is not a terminology i.e., might contain two axioms for a concept name A such as $A \equiv C$ and $A \sqsubseteq D$, the generated abstracted definition is $A \equiv C \sqcap D$, which is weaker than the original definition $A \equiv C$. One possible way to resolve this might be by restricting the process of searching for the closest primitive ancestors of a focus concept A to its respective multiple axioms in the ontology.

In terms of the computation of semantic difference between subontologies, we outline the following limitations that needs to be looked at in the future.

- Due to the fact that our approach for generating focus set subontologies does not include all of the axioms associated with supporting symbols, tracking semantic differences with respect to such symbols is incomplete. In other words, not all differences between two subontology versions associated with supporting symbols are computed. This constraint can be overcome by iteration. This means that when it is necessary to keep track of all possible semantic differences between supporting symbols, the procedure can be iterated by selecting such symbols to be considered as focus concepts. At the moment, the approach generates a complete list of semantic differences associated with the input focus

set between two subontology versions only when the input ontologies are terminologies. Ordinarily, this is something to improve in the future in order to handle $\mathcal{ELH}$ ontologies instead of only $\mathcal{ELH}$ terminologies.

- Partitioning the resulting set of witnesses is advantageous and facilitates analysis activities. However, like with the ECCO tool, the presentation of the created differences can be improved by matching them with the axioms that have caused them.

Bibliography

[ACSDD+19] Mercedes Arguello-Casteleiro, Robert Stevens, Julio Des-Diz, Chris Wroe, Maria Jesus Fernandez-Prieto, Nava Maroto, Diego Maseda-Fernandez, George Demetriou, Simon Peters, Peter-John M Noble, Phil H Jones, Jo Dukes-McEwan, Alan D Radford, John Keane, and Goran Nenadic. Exploring Semantic Deep Learning for Building Reliable and Reusable one Health Knowledge from PubMed Systematic Reviews and Veterinary Clinical Notes. *Journal of biomedical semantics*, 10(1):1–28, 2019.

[AGS] Ghadah Alghamdi, Yongsheng Gao, and Renate A Schmidt. Focus Set Semantic Differences Meeting on 14-12-2021. Meeting conducted to discuss the approach in Chapter 6 and results outlined in Section 6.5 with SNOMED CT terminologist.

[AKGBGS18] Simin Ahmadi-Karvigh, Ali Ghahramani, Burcin Becerik-Gerber, and Lucio Soibelman. Real-time Activity Recognition for Energy Efficiency in Buildings. *Applied energy*, 211:146–160, 2018.

[AKM+17] Mona Alshahrani, Mohammad Asif Khan, Omar Maddouri, Akira R Kinjo, Núria Queralt-Rosinach, and Robert Hoehndorf. Neuro-symbolic Representation Learning on Biological Knowledge Graphs. *Bioinformatics*, 33(17):2723–2730, 2017.

[ASDG21a] Ghadah Alghamdi, Renate A. Schmidt, Warren Del-Pinto, and Yongsheng Gao. Upwardly Abstracted Definition-Based Subontologies. In *Proceedings of the 34th International Workshop on Description Logics (DL 2021) part of Bratislava Knowledge September (BAKS 2021), Bratislava, Slovakia, September 19th to 22nd, 2021*, volume 2954 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

[ASDG21b] Ghadah Alghamdi, Renate A. Schmidt, Warren Del-Pinto, and Yongsheng Gao. Upwardly Abstracted Definition-Based Subontologies. In *K-CAP'21: Knowledge Capture Conference*, pages 209–216. ACM, 2021.

[Baa00] Franz Baader. Tableau Algorithms for Description Logics. In *Automated Reasoning with Analytic Tableaux and Related Methods, International Conference, TABLEAUX 2000, St Andrews, Scotland, UK, July 3-7, 2000, Proceedings*, volume 1847 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2000.

[BBB+16] Anita Bandrowski, Ryan Brinkman, Mathias Brochhausen, Matthew H. Brush, Bill Bug, Marcus C. Chibucos, Kevin Clancy, Mélanie Courtot, Dirk Derom, Michel Dumontier, Liju Fan, Jennifer Fostel, Gilberto Fragoso, Frank Gibson, Alejandra Gonzalez-Beltran, Melissa A. Haendel, Yongqun He, Mervi Heiskanen, Tina Hernandez-Boussard, Mark Jensen, Yu Lin, Allyson L. Lister, Phillip Lord, James Malone, Elisabetta Manduchi, Monnie McGee, Norman Morrison, James A. Overton, Helen Parkinson, Bjoern Peters, Philippe Rocca-Serra, Alan Ruttenberg, Susanna-Assunta Sansone, Richard H. Scheuermann, Daniel Schober, Barry Smith, Larisa N. Soldatova, Christian J. Stoeckert Jr., Chris F. Taylor, Carlo Torniai, Jessica A. Turner, Randi Vita, Patricia L. Whetzel, and Jie Zheng. The Ontology for Biomedical Investigations. *PloS one*, 11(4):e0154556, 2016.

[BBL05] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ Envelope. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30 - August 5, 2005*, pages 364–369. Professional Book Center, 2005.

[BCM+07] Franz Baader, Diego Calvanese, Deborah Mcguinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2 edition, 2007.

[BEF+21] Meisam Booshehri, Lukas Emele, Simon Flügel, Hannah Förster,

Johannes Frey, Ulrich Frey, Martin Glauer, Janna Hastings, Christian Hofmann, Carsten Hoyer-Klick, Ludwig Hülk, Anna Kleinau, Kevin Knosala, Leander Kotzur, Patrick Kuckertz, Till Mossakowski, Christoph Muschner, Fabian Neuhaus, Michaja Pehl, Martin Robinius, Vera Sehn, and Mirjam Stappel. Introducing the Open Energy Ontology: Enhancing Data Interpretation and Interfacing in Energy Systems Analysis. *Energy and AI*, 5:100074, 2021.

[BG16] Tim Benson and Grahame Grieve. *Coding and Classification Schemes*, pages 135–154. Springer International Publishing, Cham, 2016.

[BH91] Franz Baader and Philipp Hanschke. A Scheme for Integrating Concrete Domains into Concept Languages. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence. Sydney, Australia, August 24-30, 1991*, pages 452–457. Morgan Kaufmann, 1991.

[BHLS17] Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[BLS06] Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. CEL - A Polynomial-Time Reasoner for Life Science Ontologies. In *Automated Reasoning, Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, volume 4130 of *Lecture Notes in Computer Science*, pages 287–291. Springer, 2006.

[BN07] Franz Baader and Werner Nutt. *Basic Description Logics*, page 47–104. Cambridge University Press, 2 edition, 2007.

[Bra04] Sebastian Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and - what else? In Ramón López de Mántaras and Lorenza Saitta, editors, *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, pages 298–302. IOS Press, 2004.

[CAH08] Ronald Cornet and Ameen Abu-Hanna. Auditing Description-Logic-based Medical Terminological Systems by Detecting Equivalent Concept Definitions. *International Journal of Medical Informatics*, 77 5:336–45, 2008.

[CAS+19a] Jieying Chen, Ghadah Alghamdi, Renate A. Schmidt, Dirk Walther, and Yongsheng Gao. Modularity Meets Forgetting: A Case Study with the SNOMED CT Ontology. In *Proceedings of the 32nd International Workshop on Description Logics*, volume 2373. CEUR-WS.org, 2019.

[CAS+19b] Jieying Chen, Ghadah Alghamdi, Renate A. Schmidt, Dirk Walther, and Yongsheng Gao. Ontology Extraction for Large Ontologies via Modularity and Forgetting. In *Proceedings of the 10th International Conference on Knowledge Capture, K-CAP'19*, pages 45–52. ACM, 2019.

[CASG] Jieying Chen, Ghadah Alghamdi, Renate A Schmidt, and Yongsheng Gao. `https://tinyurl.com/2p8a7n7y`. (Focus on Abstraction) meeting on 15-03-2019 with SNOMED CT Terminologist.

[CCF05] Irving Copi, Carl Cohen, and Daniel Flage. *Essentials of Logic*. Routledge, 2005.

[CDD+04] Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, and Kevin Wilkinson. Jena: Implementing the Semantic Web Recommendations. In *Proceedings of the 13th international conference on World Wide Web - Alternate Track Papers & Posters, WWW 2004, New York, NY, USA, May 17-20, 2004*, pages 74–83. ACM, 2004.

[Cha] SNOMED International Youtube Channel. Invariants: how to test SNOMED CT content development - McMurtrie, Cordell and Lawley (202040) : `https://tinyurl.com/4wv2xa8e`. Accessed: 2022-06-20.

[Che18] Jieying Chen. *Knowledge Extraction from Description Logic Terminologies*. PhD thesis, University of Paris-Saclay, France, 2018.

[CLMW17] Jieying Chen, Michel Ludwig, Yue Ma, and Dirk Walther. Zooming in on Ontologies: Minimal Modules and Best Excerpts. In *Proceedings of ISWC'17*, pages 173–189, 2017.

[CLMW19] Jieying Chen, Michel Ludwig, Yue Ma, and Dirk Walther. Computing Minimal Projection Modules for $\mathcal{ELH}^r$-Terminologies. In *Proceedings of JELIA'19*, pages 355–370, 2019.

[CLW18] Jieying Chen, Michel Ludwig, and Dirk Walther. Computing Minimal Subsumption Modules of Ontologies. In *GCAI-2018, 4th Global Conference on Artificial Intelligence, Luxembourg, September 18-21, 2018*, volume 55 of *EPiC Series in Computing*, pages 41–53. EasyChair, 2018.

[CMG+22] Andrew Le Clair, Alicia Marinache, Haya El Ghalayini, Wendy Maccaull, and Ridha Khedri. A Review on Ontology Modularization Techniques - A Multi-Dimensional Perspective. *IEEE Transactions on Knowledge & Data Engineering*, (01):pre–print, feb 2022.

[CR80] Roger A. Cote and Stanley Robboy. Progress in Medical Information Management. Systematized Nomenclature of Medicine (SNOMED). *L'unión médicale du Canada*, 109:1243–52, 10 1980.

[CS09] Ronald Cornet and Stefan Schulz. Relationship Groups in SNOMED CT. In *Medical Informatics in a United and Healthy Europe - Proceedings of MIE 2009, The XXIInd International Congress of the European Federation for Medical Informatics, Sarajevo, Bosnia and Herzegovina, August 30 - September 2, 2009*, volume 150 of *Studies in Health Technology and Informatics*, pages 223–227. IOS Press, 2009.

[Cum13] Louise Cummings. Public Health Reasoning: Much More Than Deduction. *Archives of Public Health*, 71:25, 09 2013.

[DBF+14] Heiko Dietze, Tanya Z Berardini, Rebecca E Foulger, David P Hill, Jane Lomax, David Osumi-Sutherland, Paola Roncaglia, and Christopher J Mungall. TermGenie–a Web-Application for Pattern-based Ontology Class Generation. *Journal of biomedical semantics*, 5(1):1–13, 2014.

[DBLM15] T. Alexander Dececchi, James P. Balhoff, Hilmar Lapp, and Paula M. Mabee. Toward Synthesizing our Knowledge of Morphology: Using Ontologies and Machine Reasoning to Extract Presence/Absence Evolutionary Phenotypes across Studies. *Systematic biology*, 64(6):936–952, 2015.

[DdME+08] Kirill Degtyarenko, Paula de Matos, Marcus Ennis, Janna Hastings, Martin Zbinden, Alan McNaught, Rafael Alcántara, Michael Darsow, Mickaël Guedj, and Michael Ashburner. ChEBI: a Database and Ontology for Chemical Entities of Biological Interest. *Nucleic Acids Research*, 36(Database-Issue):344–350, 2008.

[Dig] NHS Digital. General Practitioner/Family Practitioner (GP/FP) Reasons for Encounter/Health Issues SNOMED CT package Release Notes - July 2019: `https://hscic.kahootz.com/connect.ti/t_c_home/view?objectId=299027#Subsets`. Accessed: 2022-03-03.

[DS17] Christophe Dessimoz and Nives Skunca. *The Gene Ontology Handbook*. Springer Nature, New york, 2017.

[DSM02] Robert H Dolin, Kent A Spackman, and David Markwell. Selective Retrieval of Pre- and Post-Coordinated SNOMED Concepts. *Proceedings. AMIA Symposium*, 2002.

[dSM06] Mathieu d'Aquin, Marta Sabou, and Enrico Motta. Modularization: a Key for the Dynamic Selection of Relevant Knowledge Components. In *Proceedings of the 1st International Workshop on Modular Ontologies, WoMO'06, co-located with the International Semantic Web Conference, ISWC'06 November 5, 2006, Athens, Georgia, USA*, volume 232 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.

[dSSS07] Mathieu d'Aquin, Anne Schlicht, Heiner Stuckenschmidt, and Marta Sabou. Ontology Modularization for Knowledge Selection: Experiments and Evaluations. In *Database and Expert Systems Applications, 18th International Conference, DEXA 2007, Regensburg, Germany, September 3-7, 2007, Proceedings*, volume 4653 of *Lecture Notes in Computer Science*, pages 874–883. Springer, 2007.

[dSSS09] Mathieu d'Aquin, Anne Schlicht, Heiner Stuckenschmidt, and Marta Sabou. Criteria and Evaluation for Ontology Modularization Techniques. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*, pages 67–89. Springer, 2009.

[DTPP09] Paul Doran, Valentina A. M. Tamma, Terry R. Payne, and Ignazio Palmisano. An Entropy Inspired Measure for Evaluating Ontology Modularization. In *Proceedings of K-CAP'09*, pages 73–80, 2009.

[EIS+06] Thomas Eiter, Giovambattista Ianni, Roman Schindlauer, Hans Tompits, and Kewen Wang. Forgetting in Managing Rules and Ontologies. In *2006 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2006), 18-22 December 2006, Hong Kong, China*, pages 411–419. IEEE Computer Society, 2006.

[ELM+05] Karen Eilbeck, Suzanna Lewis, Christopher Mungall, Mark Yandell, Lincoln Stein, Richard Durbin, and Michael Ashburner. The Sequence Ontology: a Tool for the Unification of Genome Annotations. *Genome Biology*, 6:R44, 2005.

[Eng] NHS England. Diagnostic Imaging Dataset `https://www.england.nhs.uk/statistics/statistical-work-areas/diagnostic-imaging-dataset/`. Accessed: 2022-03-03.

[Fal] Sean Falconer. OntoGraf Plugin for Protégé: `http://protegewiki.stanford.edu/wiki/OntoGraf`. Accessed: 2022-03-22.

[GAB] Yongsheng Gao, Ian Arrowsmith, and Maria Braithwaite. SNOMED CT Implementation Showcase 2013 - Presentation or Poster Abstract: SNOMED CT Clinical Imaging Procedure Codes for Radiology Information Systems and National Dataset in UK `https://www.england.nhs.uk/statistics/statistical-work-areas/diagnostic-imaging-dataset/`. Accessed: 2022-03-03.

[GEE17] K. S. Gayathri, K. S. Easwarakumar, and Susan Elias. Probabilistic Ontology Based Activity Recognition in Smart Homes Using Markov Logic Network. *Knowledge Based Systems*, 121:173–184, 2017.

[GFH+03] Jennifer Golbeck, Gilberto Fragoso, Frank W. Hartel, James A. Hendler, Jim Oberthaler, and Bijan Parsia. The National Cancer Institute's Thésaurus and Ontology. *Journal of Web Semantics*, 1(1):75–80, 2003.

[GHKS07] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Just the Right Amount: Extracting Modules from Ontologies. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 717–726. ACM, 2007.

[GHKS08] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Modular Reuse of Ontologies: Theory and Practice. *Journal of Artificial Intelligence Research (JAIR)*, 31:273–318, 2008.

[GHM+08] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter F. Patel-Schneider, and Ulrike Sattler. OWL 2: The Next Step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.

[GHM10] Birte Glimm, Ian Horrocks, and Boris Motik. Optimized Description Logic Reasoning via Core Blocking. In *Automated Reasoning, 5th International Joint Conference, IJCAR 2010, Edinburgh, UK, July 16-19, 2010. Proceedings*, volume 6173 of *Lecture Notes in Computer Science*, pages 457–471. Springer, 2010.

[GHM+14] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: An OWL 2 Reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[GLW06] Silvio Ghilardi, Carsten Lutz, and Frank Wolter. Did I Damage My Ontology? A Case for Conservative Extensions in Description Logics. In *Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom, June 2-5, 2006*, pages 187–197. AAAI Press, 2006.

[GM14] Bernardo Cuenca Grau and Boris Motik. Reasoning Over Ontologies with Hidden Content: The Import-by-Query Approach. *Journal of Artificial Intelligence Research*, abs/1401.5853, 2014.

[GPS11] Rafael S Gonçalves, B. Parsia, and U. Sattler. Categorising Logical Differences Between OWL Ontologies. In *CIKM '11*, 2011.

[GPS12a] Rafael S. Gonçalves, Bijan Parsia, and Ulrike Sattler. Concept-Based Semantic Difference in Expressive Description Logics. In *The Semantic Web - ISWC 2012 - 11th International Semantic Web Conference, Boston, MA, USA, November 11-15, 2012, Proceedings, Part I*, volume 7649 of *Lecture Notes in Computer Science*, pages 99–115. Springer, 2012.

[GPS12b] Rafael S. Gonçalves, Bijan Parsia, and Ulrike Sattler. Ecco: A Hybrid Diff Tool for OWL 2 Ontologies. In *Proceedings of OWL: Experiences and Directions Workshop 2012, Heraklion, Crete, Greece, May 27-28, 2012*, volume 849 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[HAG+16] Robert Hoehndorf, Mona Alshahrani, Georgios V. Gkoutos, George Gosline, Quentin Groom, Thomas Hamann, Jens Kattge, Sylvia Mota de Oliveira, Marco Schmidt, Soraya Sierra, Erik Smets, Rutger A. Vos, and Claus Weiland. The Flora Phenotype Ontology (FLOPO): Tool for Integrating Morphological Traits and Phenotypes of Vascular Plants. *Journal of Biomedical Semantics*, 7:65:1–65:11, 2016.

[Har] Monica Harry. Primitive parent modelling. `https://confluence.ihtsdotools.org/display/DOCEG/Proximal+Primitive+Modeling`. SNOMED International, Accessed: 2021-05-23.

[HB09] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for Working with OWL 2 Ontologies. In *Proceedings of the 6th International Conference on OWL: Experiences and Directions*, volume 2, pages 11–21, 2009.

[HBBB09] Rinke Hoekstra, Joost Breuker, Marcello Di Bello, and Alexander Boer. LKIF Core: Principled Ontology Development for the Legal

Domain. In *Law, Ontologies and the Semantic Web - Channelling the Legal Information Flood*, volume 188 of *Frontiers in Artificial Intelligence and Applications*, pages 21–52. IOS Press, 2009.

[HDG12] Robert Hoehndorf, Michel Dumontier, and Georgios V. Gkoutos. Identifying Aberrant Pathways Through Integrated Analysis of Knowledge in Pharmacogenomics. *Bioinformatics*, 28(16):2169–2175, 2012.

[HG12] A Randorff Højen and K Rosenbeck Gøeg. SNOMED CT Implementation. Mapping Guidelines Facilitating Reuse of Data. *Methods of Information in Medicine*, 51 6:529–38, 2012.

[HHH+12] Robert Hoehndorf, Midori A. Harris, Heinrich Herre, Gabriella Rustici, and Georgios V. Gkoutos. Semantic Integration of Physiology Phenotypes with an Application to the Cellular Phenotype Ontology. *Bioinformatics*, 28(13):1783–1789, 2012.

[HJ15] Yongqun He and Guoqian Jiang. Bridging Vaccine Ontology and NCIt Vaccine Domain for Cancer Vaccine Data Integration and Analysis. In *Proceedings of the International Conference on Biomedical Ontology, ICBO 2015, Lisbon, Portugal, July 27-30, 2015*, volume 1515 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[HJRS+17] Ian Harrow, Ernesto Jiménez-Ruiz, Andrea Splendiani, Martin Romacker, Peter Woollard, Scott Markel, Yasmin Alam-Faruque, Martin Koch, James Malone, and Arild Waaler. Matching Disease and Phenotype Ontologies in the Ontology Alignment Evaluation Initiative. *Journal of biomedical semantics*, 8(1):1–13, 2017.

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible $\mathcal{SROIQ}$. In *Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning, Lake District of the United Kingdom, June 2-5, 2006*, pages 57–67. AAAI Press, 2006.

[HLB+13] Midori A. Harris, Antonia Lock, Jürg Bähler, Stephen G. Oliver, and Valerie Wood. FYPO: the Fission Yeast Phenotype Ontology. *Bioinformatics*, 29(13):1671–1678, 2013.

[HMT+13] Matthew Horridge, Jonathan Mortensen, Tania Tudorache, Jennifer Vendetti, Csongor Nyulas, Mark A. Musen, and Natalya Fridman Noy. Introducing WebProtégé 2 as a Collaborative Platform for Editing Biomedical Ontologies. In *Proceedings of the 4th International Conference on Biomedical Ontology, ICBO 2013, Montreal, Canada, July 7-12, 2013*, volume 1060 of *CEUR Workshop Proceedings*, pages 138–139. CEUR-WS.org, 2013.

[HOD+16] Janna Hastings, Gareth Owen, Adriano Dekker, Marcus Ennis, Namrata Kale, Venkatesh Muthukrishnan, Steve Turner, Neil Swainston, Pedro Mendes, and Christoph Steinbeck. ChEBI in 2016: Improved Services and an Expanding Collection of Metabolites. *Nucleic Acids Research*, 44:pp. D1214 – D1219, 2016.

[Hor11] Matthew Horridge. *Justification based explanation in ontologies*. PhD thesis, University of Manchester, UK, 2011.

[HPS15] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. The OWL Explanation Workbench: A Toolkit for Working with Justifications for Entailments in OWL Ontologies. *The Semantic Web Journal - Tool/System Report*, 2015.

[HS99] Geoffrey Hinton and Terrence J. Sejnowski. *Unsupervised Learning: Foundations of Neural Computation*. Computational Neuroscience Series. MIT Press, 1999.

[HS18] Pascal Hitzler and Cogan Shimizu. Modular Ontologies as a Bridge Between Human Conceptualization and Data. In *Proceedings of ICCS'18*, pages 3–6, 2018.

[HS20] Ian Hyland and Renate A. Schmidt. Protege-TS: An OWL Ontology Term Selection Tool. In *Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), Online Event [Rhodes, Greece], September 12th to 14th, 2020*, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

[HZSBHM15] Maria Herrero-Zazo, Isabel Segura-Bedmar, Janna Hastings, and Paloma Martinez. DINTO: Using OWL ontologies and SWRL Rules to Infer Drug–Drug Interactions and their Mechanisms. *Journal of Chemical Information and Modeling*, 55(8):1698–1707, 2015.

[Inta] SNOMED International. SNOMED International Organisation `snomed.org`. Accessed: 2022-03-27.

[Intb] SNOMED International. SNOMED CT - OWL Refsets Preview: `https://confluence.ihtsdotools.org/mag/owl-refsets-preview?searchId=DBZB1IAP6`. Accessed: 2022-02-26.

[Intc] SNOMED International. SNOMED CT Basics: `https://confluence.ihtsdotools.org/display/DOCSTART/4.+SNOMED+CT+Basics`. Accessed: 2022-01-07.

[Intd] SNOMED International. SNOMED CT - Concept Model `https://confluence.ihtsdotools.org/display/DOCSTART/6.+SNOMED+CT+Concept+Model`. Accessed: 2022-02-26.

[Inte] SNOMED International. SNOMED CT Logical Model: `https://confluence.ihtsdotools.org/display/DOCSTART/5.+SNOMED+CT+Logical+Model`. Accessed: 2022-02-20.

[Intf] SNOMED International. Authoring - Description Logic (DL) Support Features: `https://confluence.ihtsdotools.org/pages/viewpage.action?pageId=61145112`. Accessed: 2022-03-08.

[Intg] SNOMED International. SNOMED CT Browsers: `https://confluence.ihtsdotools.org/display/DOCNRCG/10.+SNOMED+CT+browsers`. Accessed: 2022-03-21.

[Inth] SNOMED International. SNOMED CT Release Types: `https://confluence.ihtsdotools.org/display/DOCRELFMT/3.2+Release+Types`. Accessed: 2022-02-24.

[Inti] SNOMED International. SNOMED CT - Release Format 1: https://confluence.ihtsdotools.org/display/DOCGLOSS/RF1. Accessed: 2022-02-28.

[Intj] SNOMED International. Authoring: https://confluence.ihtsdotools.org/display/DOCEG/Authoring. Accessed: 2022-02-28.

[Intk] SNOMED International. Generating Necessary Normal Form Relationships from the OWL Refsets: https://confluence.ihtsdotools.org/display/DOCOWL/2.5.+Generating+Necessary+Normal+Form+Relationships+from+the+OWL+Refsets. Accessed: 2022-02-28.

[Intl] SNOMED International. SNOMED CT - Practical Guide to Reference Sets: https://confluence.ihtsdotools.org/display/DOCRFSPG. Accessed: 2022-02-28.

[Intm] SNOMED International. SNOMED CT - Magnetic Resonance Imaging - MRI: https://confluence.ihtsdotools.org/display/DOCEG/Magnetic+Resonance+Imaging+-+MRI. Accessed: 2022-02-20.

[Intn] SNOMED International. General Practitioner/Family Practitioner (GP/FP) Reasons for Encounter/Health Issues SNOMED CT package Release Notes - July 2019: https://confluence.ihtsdotools.org/pages/viewpage.action?pageId=94409573. Accessed: 2021-10-12.

[Into] SNOMED International. ICNP to SNOMED CT (Diagnoses) Equivalency Table Release Notes - January 2018: https://confluence.ihtsdotools.org/display/RMT/ICNP+to+SNOMED+CT+%28Diagnoses%29+Equivalency+Table+Release+Notes+-+January+2018. Accessed: 2021-10-08.

[Intp] SNOMED International. ICNP to SNOMED CT (Interventions) Equivalency table Release Notes - January 2018: https://confluence.ihtsdotools.org/display/RMT/ICNP+to+SNOMED+CT+%28Interventions%29+Equivalency+table+Release+Notes+-+January+2018. Accessed: 2021-10-08.

[Intq] SNOMED International. SNOMED CT General Dentistry Diagnostic Refset Package Release Notes - July 2021: https://confluence.ihtsdotools.org/display/RMT/SNOMED+CT+General+Dentistry+Diagnostic+refset+package+Release+Notes+-+July+2021. Accessed: 2021-10-08.

[Intr] SNOMED International. March 2020 SNOMED CT International Edition Interim Release: Updated COVID-19 Content Available: https://www.snomed.org/news-and-events/articles/march-2020-interim-snomedct-release-COVID-19. Accessed: 2022-03-08.

[Ints] SNOMED International. SNOMED CT - Versioning: https://confluence.ihtsdotools.org/display/DOCANLYT/11.4+Versioning. Accessed: 2022-02-28.

[Intt] SNOMED International. SNOMED CT - Versioning - Retrospective Delta View: https://confluence.ihtsdotools.org/display/DOCGLOSS/retrospective+delta+view. Accessed: 2022-03-01.

[Intu] SNOMED International. Quality Initiative - Project Update - Peter G. Williams, Monica Harry and Cathy Richardson (202027) : shorturl.at/nvxET. Accessed: 2022-03-20.

[JBD+19] Rebecca C Jackson, James P Balhoff, Eric Douglass, Nomi L Harris, Christopher J Mungall, and James A Overton. ROBOT: a Tool for Automating Ontology Workflows. *BMC bioinformatics*, 20(1):1–10, 2019.

[JGS+08] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support. In *Proceedings of ESWC'08*, pages 185–199, 2008.

[JSH12] Simon Jupp, Robert Stevens, and Robert Hoehndorf. Logical Gene Ontology Annotations (GOAL): Exploring Gene Ontology Annotations with OWL. *Journal of Biomedical Semantics*, 3(S-1):S3, 2012.

[KC17] Patrick Koopmann and Jieying Chen. Computing $\mathcal{ALCH}$-Subsumption Modules Using Uniform Interpolation. In *Proceedings of SOQE'17*, pages 51–66, 2017.

[KK14] Yevgeny Kazakov and Pavel Klinov. Goal-Directed Tracing of Inferences in $\mathcal{EL}$ Ontologies. In *The Semantic Web – ISWC 2014*, pages 196–211. Springer International Publishing, 2014.

[KKS12a] Yevgeny Kazakov, Markus Kroetzsch, and Frantisek Simancik. Practical Reasoning with Nominals in the $\mathcal{EL}$ Family of Description Logics. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference, KR 2012, Rome, Italy, June 10-14, 2012*. AAAI Press, 2012.

[KKS12b] Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. ELK Reasoner: Architecture and Evaluation. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012*, volume 858 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[KKS14] Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. The Incredible ELK - From Polynomial Procedures to Efficient Reasoning with $\mathcal{EL}$ Ontologies. *Journal of Automated Reasoning*, 53:1–61, 2014.

[KLWW08] Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Logical Difference and Module Extraction with CEX and MEX. In *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.

[KLWW09] Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. *Formal Properties of Modularisation*, pages 25–66. Springer Verlag, Berlin, Heidelberg, 2009.

[KLWW12] Boris Konev, Michel Ludwig, Dirk Walther, and Frank Wolter. The Logical Difference for the Lightweight Description Logic $\mathcal{EL}$. *Journal of Artificial Intelligence Research*, 44:633–708, 2012.

[KLWW13] Boris Konev, Carsten Lutz, Dirk Walther, and Frank Wolter. Model-Theoretic Inseparability and Modularity of Description Logic Ontologies. *Artificial Intelligence*, 203:66–103, 2013.

[Knu] Holger Knublauch. Travel.owl : `http://protege.cim3.net/file/pub/ontologies/travel/travel.owl#`. Accessed: 2022-03-20.

[Koo15] Patrick Koopmann. *Practical Uniform Interpolation for Expressive Description Logics*. PhD thesis, University of Manchester, UK, 2015.

[Koo20] Patrick Koopmann. LETHE: Forgetting and Uniform Interpolation for Expressive Description Logics. *Künstliche Intelligenz*, 34(3):381–387, 2020.

[KPHS07] Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding All Justifications of OWL DL Entailments. In *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.

[KS13a] Patrick Koopmann and Renate A. Schmidt. Forgetting Concept and Role Symbols in $\mathcal{ALCH}$ -Ontologies. In *Logic for Programming, Artificial Intelligence, and Reasoning - 19th International Conference, LPAR-19, Stellenbosch, South Africa, December 14-19, 2013. Proceedings*, volume 8312 of *Lecture Notes in Computer Science*, pages 552–567. Springer, 2013.

[KS13b] Patrick Koopmann and Renate A. Schmidt. Implementation and Evaluation of Forgetting in $\mathcal{ALC}$-Ontologies. In *Proceedings of the*

7th International Workshop on Modular Ontologies co-located with the 12th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR 2013), Corunna, Spain, September 15, 2013, volume 1081 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013.

[KS13c] Patrick Koopmann and Renate A. Schmidt. Uniform Interpolation of $\mathcal{ALC}$-Ontologies Using Fixpoints. In *Proceedings of FroCoS'13*, volume 8152, pages 87–102, 2013.

[KS14] Patrick Koopmann and Renate A. Schmidt. Count and Forget: Uniform Interpolation of $\mathcal{SHQ}$ -Ontologies. In *Automated Reasoning - 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings*, volume 8562 of *Lecture Notes in Computer Science*, pages 434–448. Springer, 2014.

[KS15] Patrick Koopmann and Renate A. Schmidt. Uniform Interpolation and Forgetting for $\mathcal{ALC}$ Ontologies with ABoxes. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 175–181. AAAI Press, 2015.

[KSK11] Petr Kremen, Marek Smid, and Zdenek Kouba. OWLDiff: A Practical Tool for Comparison and Merge of OWL Ontologies. In *Proceedings of the 2011 22nd International Workshop on Database and Expert Systems Applications*, pages 229–233. IEEE Computer Society, 2011.

[KWW08] Boris Konev, Dirk Walther, and Frank Wolter. The Logical Difference Problem for Description Logic Terminologies. In *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, volume 5195 of *Lecture Notes in Computer Science*, pages 259–274. Springer, 2008.

[KWW09] Boris Konev, Dirk Walther, and Frank Wolter. Forgetting and Uniform Interpolation in Large-Scale Description Logic Terminologies. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July (IJCAI)*, pages 830–835, 2009.

[KWZ10] Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. Logic-based Ontology Comparison and Module Extraction, with an Application to DL-Lite. *Artificial Intelligence*, 2010.

[LBIS12] Pablo López-García, Martin Boeker, Arantza Illarramendi, and Stefan Schulz. Usability-Driven Pruning of Large Ontologies: the Case of SNOMED CT. *Journal of the American Medical Informatics Association*, 19, 2012.

[LCL11] Dennis Lee, Ronald Cornet, and Francis Lau. Implications of SNOMED CT Versioning. *International Journal of Medical Informatics*, 80(6):442–453, 2011.

[LdKLC14] Dennis Lee, Nicolette F. de Keizer, Francis Y. Lau, and Ronald Cornet. Literature Review of SNOMED CT Use. *Journal of the American Medical Informatics Association : JAMIA*, 21 e1:e11–9, 2014.

[LK14] Michel Ludwig and Boris Konev. Practical Uniform Interpolation and Forgetting for $\mathcal{ALC}$ TBoxes with Applications to Logical Difference. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria*. AAAI Press, 2014.

[LLA+21a] Zhao Liu, Chang Lu, Ghadah Alghamdi, Renate A. Schmidt, and Yizheng Zhao. Tracking Semantic Evolutionary Changes in Large-Scale Ontological Knowledge Bases. In *CIKM'21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 1130–1139. ACM, 2021.

[LLA+21b] Zhao Liu, Chang Lu, Ghadah Alghamdi, Renate A. Schmidt, and Yizheng Zhao. Tracking Semantic Evolutionary Changes in Large-Scale Ontological Knowledge Bases. In *Proceedings of the 34th International Workshop on Description Logics (DL 2021) part of Bratislava Knowledge September (BAKS 2021), Bratislava, Slovakia, September 19th to 22nd, 2021*, volume 2954 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

[LLM03] Jérôme Lang, Paolo Liberatore, and Pierre Marquis. Propositional Independence: Formula-Variable Independence and Forgetting. *Journal of Artificial Intelligence Research (JAIR)*, 18:391–443, 2003.

[LNB14] Steffen Lohmann, Stefan Negru, and David Bold. The ProtégéVOWL Plugin: Ontology Visualization for Everyone. In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, volume 8798 of *Lecture Notes in Computer Science*, pages 395–400. Springer, 2014.

[LPW10] Carsten Lutz, Robert Piro, and Frank Wolter. Enriching $\mathcal{EL}$-Concepts with Greatest Fixpoints. In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, pages 41–46. IOS Press, 2010.

[LR94] Fangzhen Lin and Ray Reiter. Forget it! In *Proceedings of the AAAI Fall Symposium on Relevance*, pages 154–159, 1994.

[LS16] Pablo López-García and Stefan Schulz. Can SNOMED CT be Squeezed Without Losing its Shape? *Journal of Biomedical Semantics*, 7:56, 2016.

[LSW12] Carsten Lutz, Inanç Seylan, and Frank Wolter. An Automata-Theoretic Approach to Uniform Interpolation and Approximation in the Description Logic $\mathcal{EL}$. In *Proceedings of KR'12*, 2012.

[Lut99] Carsten Lutz. Reasoning with Concrete Domains. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages*, pages 90–95. Morgan Kaufmann, 1999.

[Lut02] Carsten Lutz. Description Logics with Concrete Domains-A Survey. In *Advances in Modal Logic 4, papers from the fourth conference on "Advances in Modal logic," held in Toulouse, France, 30 September - 2 October 2002*, pages 265–296. King's College Publications, 2002.

[LW11] Carsten Lutz and Frank Wolter. Foundations for Uniform Interpolation and Forgetting in Expressive Description Logics. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain*, pages 989–995. IJCAI/AAAI, 2011.

[LW14] Michel Ludwig and Dirk Walther. The Logical Difference for $\mathcal{ELH}^r$-Terminologies Using Hypergraphs. In *ECAI 2014 - 21st 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.

[LW16] Michel Ludwig and Dirk Walther. Towards a Practical Decision Procedure for Uniform Interpolants of $\mathcal{EL}$-TBoxes – a Proof-Theoretic Approach. In *Proceedings of GCAI'16*, 2016.

[MCNK15] Till Mossakowski, Mihai Codescu, Fabian Neuhaus, and Oliver Kutz. The Distributed Ontology, Modeling and Specification Language – DOL. In *The Road to Universal Logic*, volume 2, pages 489–520. Birkhäuser, 2015.

[MGH+12] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Achille Fokoue, and Carsten Lutz, editors. *OWL 2 Web Ontology Language Profiles (Second Edition)*. Available at: `https://www.w3.org/TR/owl2-profiles/`, 2012.

[MH08a] Boris Motik and Ian Horrocks. Individual Reuse in Description Logic Reasoning. In *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, volume 5195 of *Lecture Notes in Computer Science*, pages 242–258. Springer, 2008.

[MH08b] Boris Motik and Ian Horrocks. OWL Datatypes: Design and Implementation. In *The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*, volume 5318 of *Lecture Notes in Computer Science*, pages 307–322. Springer, 2008.

[MHA+10] James Malone, Ele Holloway, Tomasz Adamusiak, Misha Kapushesky, Jie Zheng, Nikolay Kolesnikov, Anna Zhukova, Alvis Brazma, and Helen E. Parkinson. Modeling Sample Variables with an Experimental Factor Ontology. In *Bioinformatics*, 2010.

[MSH14] Boris Motik, Robert D. C. Shearer, and Ian Horrocks. Hypertableau Reasoning for Description Logics. *CoRR*, abs/1401.3485, 2014.

[Mus15] Mark A. Musen. The Protégé Project: a Look Back and a Look Forward. *AI Matters*, 1(4):4–12, 2015.

[NCS+18] Marlies Noordzij, Ronald Cornet, Keith Simpson, Kitty J Jager, and Charles R V Tomson. An Update of the ERA-EDTA Registry Primary Renal Disease Coding System: What's New? *Nephrology Dialysis Transplantation*, 34(6):896–898, 11 2018.

[NG12] Nadeschda Nikitina and Birte Glimm. Hitting the Sweetspot: Economic Rewriting of Knowledge Bases. In *The Semantic Web – ISWC 2012*, pages 394–409. Springer Berlin Heidelberg, 2012.

[nhs] NHS Digital - SNOMED CT subset members. `https://isd.digital.nhs.uk/trud/users/guest/filters/0/categories/40`. Accessed: 2022-02-20.

[Nik12] Nadeschda Nikitina. *Reasoning-Supported Quality Assurance for Knowledge Bases*. PhD thesis, Karlsruhe Institute of Technology, 2012.

[NK17] Nadeschda Nikitina and Patrick Koopmann. Small Is Beautiful: Computing Minimal Equivalent $\mathcal{EL}$ Concepts. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1206–1212. AAAI Press, 2017.

[NM09] Natalya F. Noy and Mark A. Musen. *Traversing Ontologies to Extract Views*, page 245–260. Springer-Verlag, Berlin, Heidelberg, 2009.

[NR14] Nadeschda Nikitina and Sebastian Rudolph. (Non-)Succinctness of Uniform Interpolants of General Terminologies in the Description Logic $\mathcal{EL}$. *Artificial Intelligence*, 215:120–140, 2014.

[NTW+11] Jithun Nair, Tania Tudorache, Trish Whetzel, Natalya Fridman Noy, and Mark A. Musen. The BioPortal Import Plugin for Protégé. In *Proceedings of the 2nd International Conference on Biomedical Ontology, Buffalo, NY, USA, July 26-30, 2011*, volume 833 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

[OCP16] Christopher Ochs, James T. Case, and Yehoshua Perl. Tracking the Remodeling of SNOMED CT's Bacterial Infectious Diseases. In *AMIA 2016, American Medical Informatics Association Annual Symposium, Chicago, IL, USA, November 12-16, 2016*. AMIA, 2016.

[OCP17] Christopher Ochs, James T. Case, and Yehoshua Perl. Analyzing Structural Changes in SNOMED CT's Bacterial Infectious Diseases Using a Visual Semantic Delta. *Journal of Biomedical Informatics*, 67:101–116, 2017.

[OGP+15] Christopher Ochs, James Geller, Yehoshua Perl, Yan Chen, Junchuan Xu, Hua Min, James T. Case, and Zhi Wei. Scalable Quality Assurance for Large SNOMED CT Hierarchies Using Subject-based Subtaxonomies. *Journal of the American Medical Informatics Association*, 22(3):507–518, 2015.

[oPa] Chartered Society of Physiotherapy. Chartered Society of Physiotherapy - SNOMED CT subsets: `https://www.csp.org.uk/documents/snomed-ct-subsets`. Accessed: 2022-03-04.

[oPb] Chartered Society of Physiotherapy. Chartered Society of Physiotherapy - SNOMED CT subsets: `https://www.csp.org.uk/system/files/csp_snomed_ct_subsets_20160414_v1.pdf`. Accessed: 2021-05-23.

[OPEC16] Christopher Ochs, Yehoshua Perl, Gai Elhanan, and James T. Case. A Descriptive Delta for Identifying Changes in SNOMED CT. In *Proceedings of the Joint International Conference on Biological Ontology and BioCreative, Corvallis, Oregon, United States, August 1-4, 2016*, volume 1747 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.

[ORM+12] David Osumi-Sutherland, Simon Reeve, Christopher J. Mungall, Fabian Neuhaus, Alan Ruttenberg, Gregory S. X. E. Jefferis, and J. Douglas Armstrong. A Strategy for Building Neuroanatomy Ontologies. *Bioinformatics*, 28(9):1262–1269, 2012.

[OWL] OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition). `https://www.w3.org/TR/owl2-syntax/`. Accessed: 2021-10-08.

[PJC09] Jyotishman Pathak, Thomas M. Johnson, and Christopher G. Chute. Survey of Modular Ontology Techniques and their Applications in the Biomedical Domain. *Integrated Computer-Aided Engineering*, 16(3):225–242, 2009.

[Pra] BMJ Best Practice. `https://bestpractice.bmj.com/info/about-us/`. Accessed: 2022-03-07.

[PS08] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF. `https://www.w3.org/TR/rdf-sparql-query/`, 2008.

[QMMAN+18] Manuel Quesada-Martínez, Mar Marcos, Francisco Abad-Navarro, Begoña Martínez-Salvador, and Jesualdo Tomás Fernández-Breis. Towards the Semantic Enrichment of Computer Interpretable Guidelines: a Method for the Identification of Relevant Ontological Terms. In *AMIA Annual Symposium Proceedings*, volume 2018, page 922. American Medical Informatics Association, 2018.

[Rec95] Alan Rector. Coordinating Taxonomies: Key to Re-Usable Concept Representations. In *Proceedings of Artificial Intelligence Medicine, 5th Conference on Artificial Intelligence in Medicine in Europe, AIME'95, Pavia, Italy*, volume 934, pages 17–28. Springer, 1995.

[RGHJ13] Ana Armas Romero, Bernardo Cuenca Grau, Ian Horrocks, and Ernesto Jiménez-Ruiz. MORe: a Modular OWL Reasoner for Ontology Classification. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, volume 1015 of *CEUR Workshop Proceedings*, pages 61–67. CEUR-WS.org, 2013.

[RNM07] Daniel Rubin, Natasha Noy, and Mark Musen. Protégé: A Tool for Managing and Using Terminology in Radiology Applications. *Journal of digital imaging : the official journal of the Society for Computer Applications in Radiology*, 20 Suppl 1:34–46, 12 2007.

[Roc75] Marc J. Rochkind. The Source Code Control System. *IEEE Transactions on Software Engineering*, SE-1:364–370, 1975.

[RU15] Marie-Christine Rousset and Federico Ulliana. Extracting Bounded-Level Modules from Deductive RDF Triplestores. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[SCC97] Kent A. Spackman, Keith Campbell, and RA Cote. SNOMED RT: a Reference Terminology for Health Care. In *Proceedings of AMIA'97*, 1997.

[SDMW02] Kent A. Spackman, Robert Dionne, Eric Mays, and Jason Weis. Role grouping as an extension to the description logic of ontology, motivated by concept modeling in SNOMED. In *Proceedings of AMIA'02*, 2002.

[SGD+18] Brian J Stucky, Rob Guralnick, John Deck, Ellen G Denny, Kjell Bolmgren, and Ramona Walls. The Plant Phenology Ontology: a New Informatics Resource for Large-scale Integration of Plant Phenology Data. *Frontiers in Plant Science*, 9:517, 2018.

[SGH19] Fatima Zohra Smaili, Xin Gao, and Robert Hoehndorf. OPA2Vec: Combining Formal and Informal Content of Biomedical Ontologies to Improve Similarity-based Prediction. *Bioinformatics*, 35(12):2133–2140, 2019.

[SGSK18] Giorgos Stoilos, David Geleta, Jetendr Shamdasani, and Mohammad Khodadadi. A Novel Approach and Practical Algorithms for Ontology Integration. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*, volume 11136 of *Lecture Notes in Computer Science*, pages 458–476. Springer, 2018.

[SGW+18] Giorgos Stoilos, David Geleta, Szymon Wartak, Sheldon Hall, Mohammad Khodadadi, Yizheng Zhao, Ghadah Alghamdi, and Renate A. Schmidt. Methods and Metrics for Knowledge Base Engineering and Integration. In *Proceedings of the 9th Workshop on Ontology Design and Patterns (WOP 2018) co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 9th, 2018*, volume 2195 of *CEUR Workshop Proceedings*, pages 72–86. CEUR-WS.org, 2018.

[SK04] Heiner Stuckenschmidt and Michel C. A. Klein. Structure-Based Partitioning of Large Concept Hierarchies. In *The Semantic Web - ISWC 2004: Third International Semantic Web Conference,Hiroshima, Japan, November 7-11, 2004. Proceedings*, volume 3298 of *Lecture Notes in Computer Science*, pages 289–303. Springer, 2004.

[Spa00a] Kent A. Spackman. Managing Clinical Terminology Hierarchies Using Algorithmic Calculation of Subsumption: Experience with SNOMED RT. *Journal of the American Medical Informatics Association*, 2000.

[Spa00b] Kent A. Spackman. SNOMED RT and SNOMED CT. Promise of an International Clinical Ontology. *M.D. computing : computers in medical practice*, 17:29, 2000.

[Spa01] Kent A. Spackman. Normal Forms for Description Logic Expressions of Clinical Concepts in SNOMED RT. In *AMIA 2001, American Medical Informatics Association Annual Symposium, Washington, DC, USA*. AMIA, 2001.

[SPG+07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A Practical OWL-DL Reasoner. *Journal of Web Semantics*, 5(2):51–53, 2007.

[SPS09] Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*. Springer Verlag, 01 2009.

[SR06] Julian Seidenberg and Alan Rector. Web Ontology Segmentation: Analysis, Classification and Use. In *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, page 13–22, New York, NY, USA, 2006. Association for Computing Machinery.

[SRP+13] Vasil Slavov, Praveen Rao, Srivenu Paturi, Tivakar Komara Swami, Michael Barnes, Deepthi Rao, and Raghuvarun Palvai. A New Tool for Sharing and Querying of Clinical Documents Modeled Using HL7 Version 3 Standard. *Computer Methods and Programs in Biomedicine*, 112(3):529–552, 2013.

[SS20] Thomas Schneider and Mantas Simkus. Ontologies and Data Management: A Brief Survey. *Künstliche Intelligenz.*, 34(3):329–353, 2020.

[SSZ09] Ulrike Sattler, Thomas Schneider, and Michael Zakharyaschev. Which Kind of Module Should I Extract? In *Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, July 27-30, 2009*, volume 477 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[TC09] Milorad Tosic and Marija Cubric. SeMCQ–Protégé Plugin for Automatic Ontology-Driven Multiple Choice Question Tests Generation. In *Proceedings of the 11th International Protege Conference*. Stanford Center for Biomedical Informatics Research, 2009.

[TH06] Dmitry Tsarkov and Ian Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Automated Reasoning, Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, volume 4130 of *Lecture Notes in Computer Science*, pages 292–297. Springer, 2006.

[The] The Gene Ontology Consortium. `http://geneontology.org/docs/go-subset-guide/`. Accessed: 2021-05-23.

[The18] The Gene Ontology Consortium. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research*, 49:gky1055, 2018.

[Tic96] Walter Tichy. RCS—A System for Version Control. 06 1996.

[Tsa12] Dmitry Tsarkov. Improved Algorithms for Module Extraction and Atomic Decomposition. In *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012*, volume 846 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[Ulr] Brandon Ulrich. `https://bit.ly/2RqRA9A`. Accessed: 2021-06-06.

[vDQCF18] Philip van Damme, Manuel Quesada-Martínez, Ronald Cornet, and Jesualdo Tomás Fernández-Breis. From Lexical Regularities to Axiomatic Patterns for the Quality Assurance of Biomedical Terminologies and Ontologies. *Journal of Biomedical Informatics*, 84:59–74, 2018.

[VKP+13] Chiara Del Vescovo, Pavel Klinov, Bijan Parsia, Ulrike Sattler, Thomas Schneider, and Dmitry Tsarkov. Empirical Study of Logic-Based Modules: Cheap Is Cheerful. In *Proceedings of the Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia*, volume 8218, pages 84–100. Springer, 2013.

[VRTG+12] Gopalakrishnan Venkat-Raman, Charles R.V. Tomson, Yongsheng Gao, Ronald Cornet, Benedicte Stengel, Carola Gronhagen-Riska, Chris Reid, Christian Jacquelinet, Elke Schaeffner, Els Boeschoten, Francesco Casino, Frederic Collart, Johan De Meester, Oscar Zurriaga, Reinhard Kramar, Kitty J. Jager, and Keith Simpson. New Primary Renal Diagnosis Codes for the ERA-EDTA. *Nephrology Dialysis Transplantation*, 27(12):4414–4419, 11 2012.

[W3C12] W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview (Second Edition) - W3C Recommendation 11 December 2012, December 2012.

[WBB+01] Amy Y. Wang, James W. Barrett, Timothy E. Bentley, David Markwell, Colin Price, Kent A. Spackman, and Michael Q. Stearns. Mapping between SNOMED RT and Clinical Terms Version 3: a Key Component of the SNOMED CT Development Process. In *AMIA*

2001, American Medical Informatics Association Annual Symposium, Washington, DC, USA, November 3-7, 2001. AMIA, 2001.

[WDL+20] Xuan Wu, Wenxing Deng, Chang Lu, Hao Feng, and Yizheng Zhao. UI-FAME: A High-Performance Forgetting System for Creating Views of Ontologies. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 3473–3476. ACM, 2020.

[WLZ+20] Xuan Wu, Chang Lu, Yizheng Zhao, Renate A. Schmidt, and Hao Feng. UI-FAME: A Deductive Forgetting Tool for Creating Views of ALC-TBoxes. In *Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020), Online Event [Rhodes, Greece], September 12th to 14th, 2020*, volume 2663 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2020.

[WNS+11] Patricia L. Whetzel, Natalya Fridman Noy, Nigam H. Shah, Paul R. Alexander, Csongor Nyulas, Tania Tudorache, and Mark A. Musen. BioPortal: Enhanced Functionality via New Web Services from the National Center for Biomedical Ontology to Access and Use Ontologies in Software Applications. *Nucleic Acids Research*, 39(Web-Server-Issue):541–545, 2011.

[WSB11] Pinar Wennerberg, Klaus U. Schulz, and Paul Buitelaar. Ontology Modularization to Improve Semantic Medical Image Annotation. *Journal of Biomedical Informatics*, 44(1):155–162, 2011.

[WWTP10] Zhe Wang, Kewen Wang, Rodney W. Topor, and Jeff Z. Pan. Forgetting for Knowledge Bases in DL-Lite. *Annals of Mathematics and Artificial Intelligence*, 58(1-2):117–151, 2010.

[ZAS+19a] Yizheng Zhao, Ghadah Alghamdi, Renate A. Schmidt, Hao Feng, Giorgos Stoilos, Damir Juric, and Mohammad Khodadadi. Tracking Logical Difference in Industrial-Scale Ontologies. In *Proceedings of*

the 32nd International Workshop on Description Logics, Oslo, Norway, June 18-21, 2019, volume 2373 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

[ZAS+19b] Yizheng Zhao, Ghadah Alghamdi, Renate A. Schmidt, Hao Feng, Giorgos Stoilos, Damir Juric, and Mohammad Khodadadi. Tracking Logical Difference in Large-Scale Ontologies: A Forgetting-Based Approach. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3116–3124. AAAI Press, 2019.

[ZGN+15] Yujiao Zhou, Bernardo Cuenca Grau, Yavor Nenov, Mark Kaminski, and Ian Horrocks. PAGOdA: Pay-as-you-go Ontology Query Answering Using a Datalog Reasoner. *Journal of Artificial Intelligence Research*, 54:309–367, 2015.

[Zha18] Yizheng Zhao. *Automated semantic forgetting for expressive description logics*. PhD thesis, 2018.

[ZS18a] Yizheng Zhao and Renate A. Schmidt. FAME: an automated tool for semantic forgetting in expressive description logics. In *Automated Reasoning - 9th International Joint Conference, IJCAR 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings*, volume 10900 of *Lecture Notes in Computer Science*, pages 19–27. Springer, 2018.

[ZS18b] Yizheng Zhao and Renate A. Schmidt. On Concept Forgetting in Description Logics with Qualified Number Restrictions. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden*, pages 1984–1990. ijcai.org, 2018.

[ZS19] Yizheng Zhao and Renate A. Schmidt. FAME(Q): an automated tool for forgetting in description logics with qualified number restrictions. In *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.

[ZWZ+21] Xinhao Zhu, Xuan Wu, Ruiqing Zhao, Yu Dong, and Yizheng Zhao. Metadata-based term selection for modularization and uniform interpolation of OWL ontologies. In *Proceedings of the Second Workshop on Second-Order Quantifier Elimination and Related Topics (SOQE 2021) associated with the 18th International Conference on Principles of Knowledge Representation and Reasoning (KR 2021), Online Event, November 4, 2021*, volume 3009 of *CEUR Workshop Proceedings*, pages 122–134. CEUR-WS.org, 2021.

Appendix A

Appendix

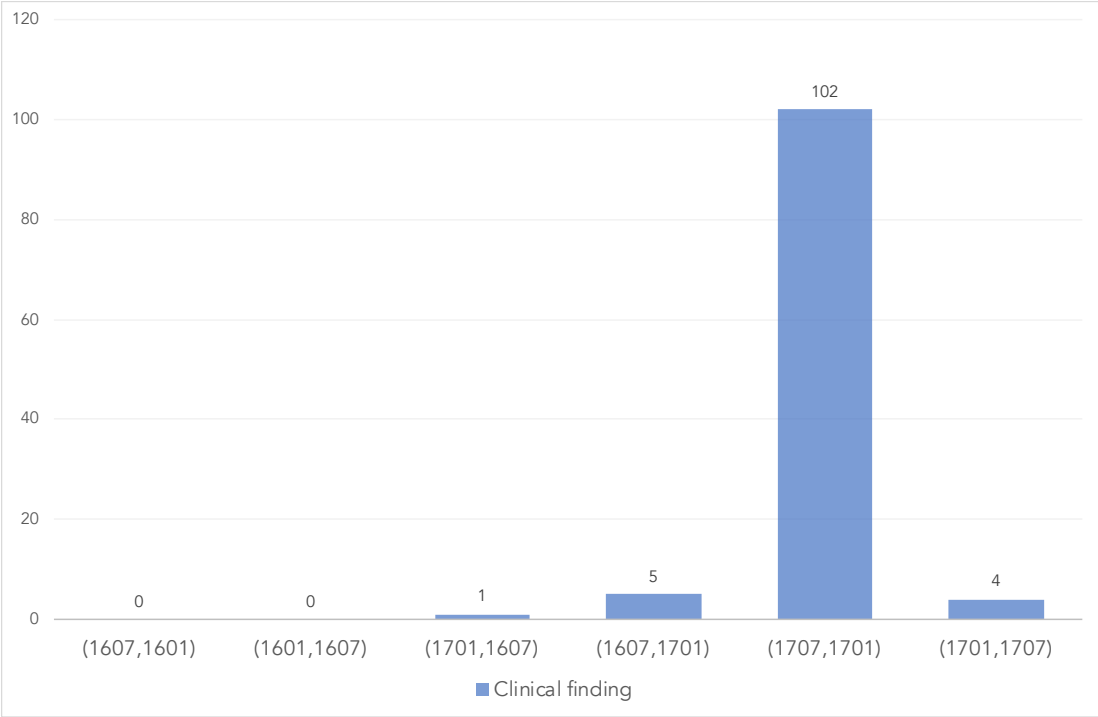

Figure A.1: Focus witnesses concept hierarchy in ICNP comparisons

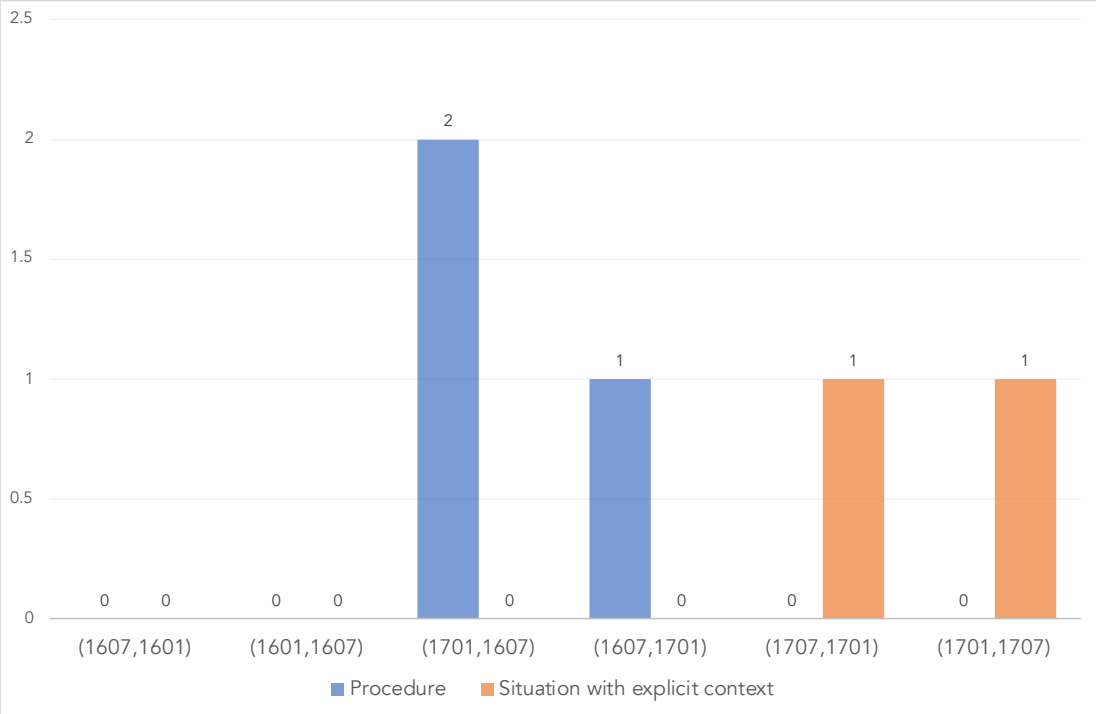

Figure A.2: Focus witnesses concept hierarchy in ICNP-Interventions comparisons

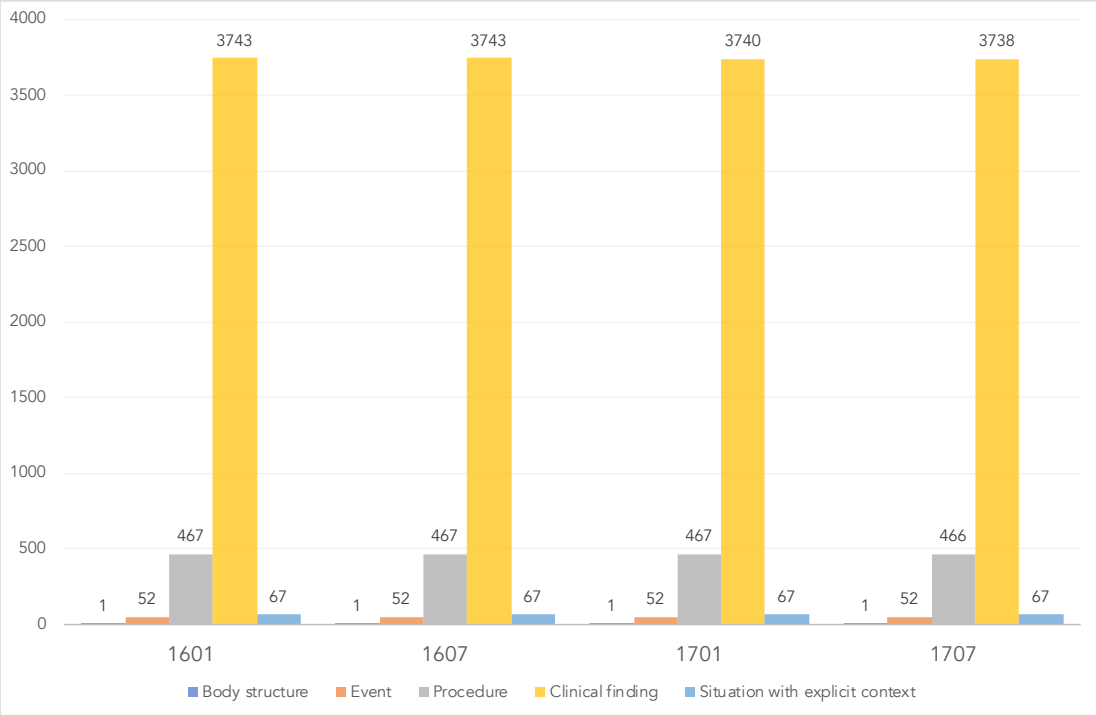

Figure A.3: Concepts in GPFP refset main concepts hierarchies

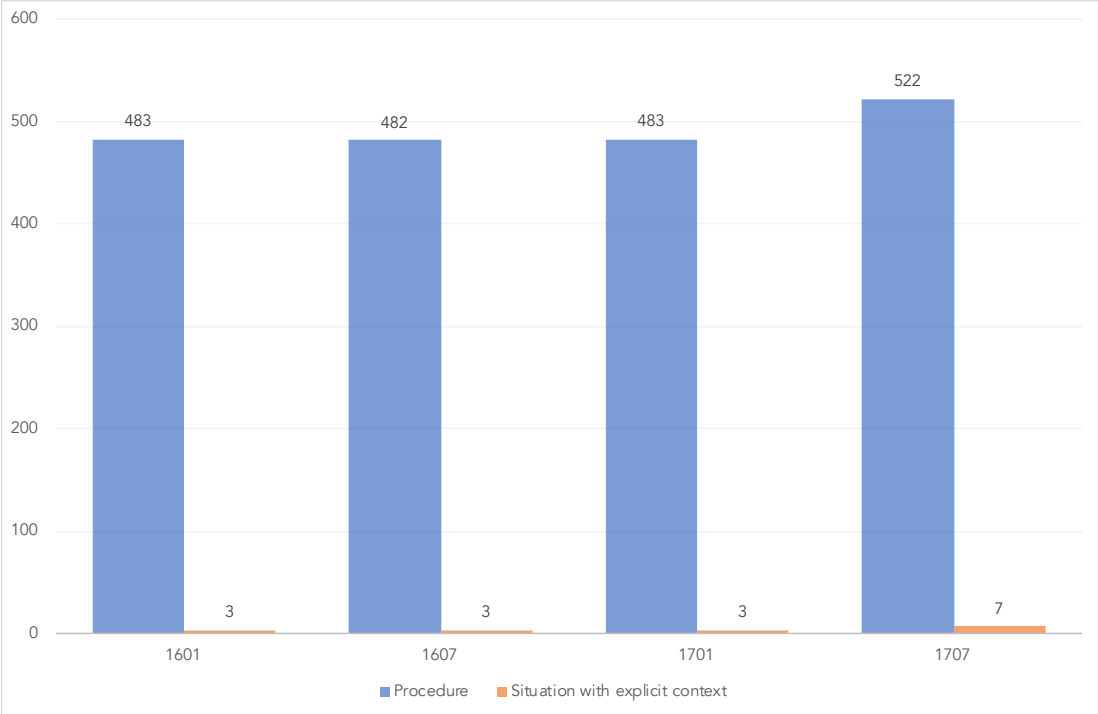

Figure A.4: Concepts in ICNP refset main concepts hierarchies

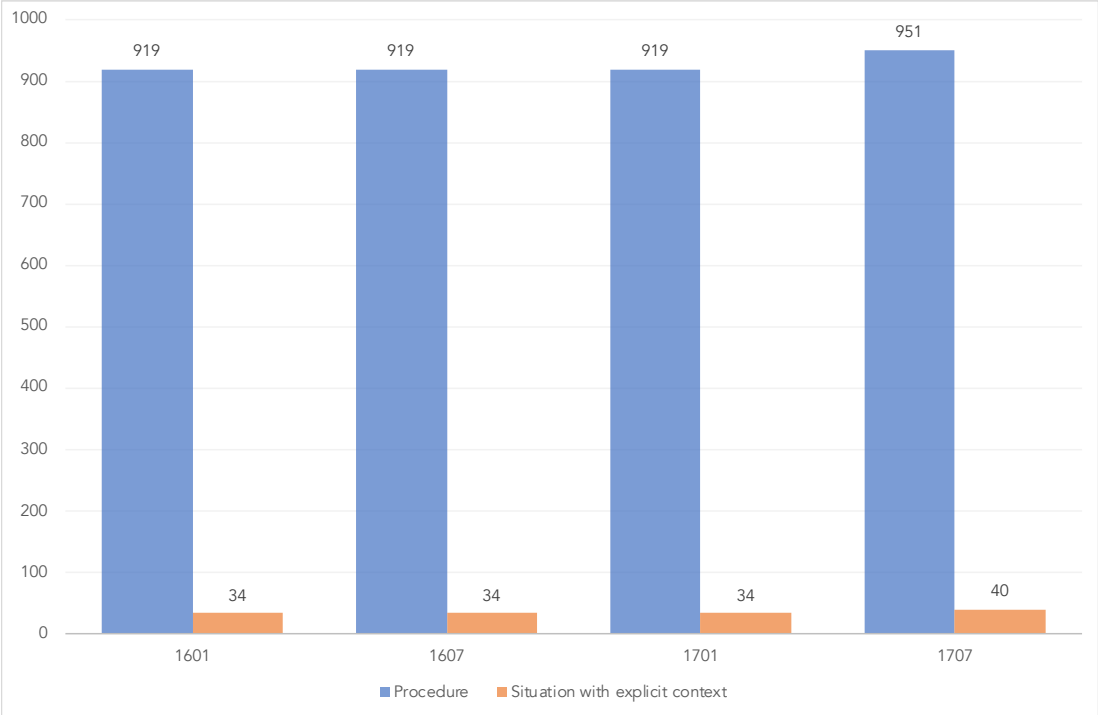

Figure A.5: Concepts in ICNP-Interventions refset main concepts hierarchies