Tracking Semantic Evolutionary Changes in Large-Scale Ontological Knowledge Bases*

Zhao Liu[1,2], Chang Lu[1,2], Ghadah Alghamdi[3], Renate A. Schmidt[3], and Yizheng Zhao**[1,2]

[1] National Key Laboratory for Novel Software Technology, Nanjing Univeristy, China
[2] School of Artificial Intelligence, Nanjing University, China
[3] Department of Computer Science, The University of Manchester, UK

Abstract. This paper is concerned with the problem of computing the semantic difference between different versions of large-scale ontologies using a uniform interpolation (UI) approach. The semantic difference between two versions of an ontology are the axioms entailed by one version but not the other, reflecting the semantic evolutionary changes of the ontology. We develop a novel, tailor-made UI method for the task of computing semantic difference in large-scale ontologies, which often specified in the description logic $\mathcal{ELH}$. The method is terminating and sound, and can always compute results of UI when such results exist. A case study on different versions of the SNOMED CT terminology shows that the new method has overcome major drawbacks and limitations of existing methods, and has provided a feasible approach to the task of computing semantic difference in large-scale ontologies.

1 Introduction

In Computer Science & Artificial Intelligence (AI), *ontologies* are a formal description of knowledge as a set of concepts within a domain and the relationships that hold between the concepts.

Since ontologies are dynamic entities that are constantly evolving, computing the semantic difference between two versions of ontologies can be a critical task: to track what has changed in a new version of an ontology, to ensure that the changes are safe in the sense of the new version being a conservative extension of a preceding version [4,12], and to identify unexpected consequences in versions of an ontology. This provides effective means for discovering issues in ontologies and enhances quality control during the ontology evolution process. Being able to compute the semantic difference between ontologies is also important when merging and aligning ontologies from different sources [7,17].

A straightforward way to compute $\mathsf{Diff}(\mathcal{T}_1, \mathcal{T}_2)$ is to first compute all logical entailments $\mathcal{V}_2$ of $\mathcal{T}_2$, and then collect from $\mathcal{V}_2$ the axioms not entailed by $\mathcal{T}_1$.

** Yizheng Zhao is the corresponding author of this paper.

However, $\mathcal{V}_2$ can be infinite and computing $\mathcal{V}_2$ is not always computationally feasible [8]. Konev et al. [8] has proposed an approach to compute finite representations of the semantic difference between two ontologies. The idea is that, rather than computing all entailments of one ontology not entailed by the other ontology, which would be computationally infeasible, only the strongest entailments not entailed by the other ontology are computed. Then all logical entailments can in principle be computed from the *deductive closure* of the strongest entailments. This approach computes the strongest entailments of an ontology using an abstraction technique called *uniform interpolation* (UI), which seeks to create views of ontologies while preserves the logical models (the semantics) of the views [19,11].

Existing UI methods are designed for DLs that are either more expressive or less expressive than real-world large-scale ontologies, which are often expressed in the DL $\mathcal{ELH}$. This means that the computed views of the ontologies will contain language constructs that are outside of the language of $\mathcal{ELH}$ or they do not support language constructs of $\mathcal{ELH}$. To be useful, views must be in the language of the input ontology and also satisfy the modeling guidelines of the development community.

Contributions. We introduce a novel, tailor-made UI method for the task of computing semantic difference for ontologies expressed in the DL $\mathcal{ELH}$ and as large as SNOMED CT [18]. Our UI method is terminating, sound, and can always compute a uniform interpolant when such a uniform interpolant exists. An empirical evaluation with a prototype implementation shows very good success rates and performance results on a large corpus of real-world ontologies taken from the Oxford ISG Library. A case study on different releases of SNOMED CT shows that the new method has overcome major drawbacks and limitations of existing methods.

The source code, the long version of this paper and all test data are distributed at github.com/anonymous-ai-researcher/DL2021. User-friendly web access to try out these tools is possible at http://www.forgettingshow.info/.

2 Preliminaries

Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be disjoint and countably infinite sets of *concept names* and *role names*, respectively. *$\mathcal{ELH}$-concepts* are inductively constructed based on the following syntax rule:

$$C, D \longrightarrow \top \mid A \mid C \sqcap D \mid \exists r.C,$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and C and D range over concepts. Let $\mathcal{T}$ be an $\mathcal{ELH}$-TBox and $A, B_i \in \mathsf{N_C}$ $(1 \leq i \leq n)$ be atomic concepts in $\mathcal{T}$. We say that A *directly depends on* B ($A \prec B$) iff the clausal form of $\mathcal{T}$ 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 $A \prec B_1 \prec \ldots \prec B_n$. $\mathcal{T}$ is *acyclic* if there is no concept name in $\mathcal{T}$ that depends on itself; otherwise it

is *cyclic*. In the remainder of this paper, the terms *TBox* and *ontology* are used interchangeably.

The semantics of $\mathcal{ELH}$ 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 follows:

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \qquad (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$$
$$(\exists r.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y.(x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$$

A *signature* $\mathsf{sig} \subseteq \mathsf{N_C} \cup \mathsf{N_R}$ is a finite set of concept and role names. By $\mathsf{sig_C}(X)$ and $\mathsf{sig_R}(X)$ we denote the sets of respectively the concept names and role names occurring in X, where X ranges over concepts, axioms and ontologies. We let $\mathsf{sig}(X) = \mathsf{sig_C}(X) \cup \mathsf{sig_R}(X)$. An axiom α with $\mathcal{S} \in \mathsf{sig}(\alpha)$ is called an $\mathcal{S}$*-axiom*.

Definition 1 (Semantic Difference). *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two $\mathcal{ELH}$-ontologies. Let Σ be a subset of the shared signature of $\mathcal{T}_1$ and $\mathcal{T}_2$. The* semantic difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ *for Σ is the set $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of all $\mathcal{ELH}$-axioms α such that (i) $\mathsf{sig}(\alpha) \subseteq \Sigma$, (ii) $\mathcal{T}_2 \models \alpha$, but (iii) $\mathcal{T}_1 \not\models \alpha$. An axiom α satisfying these conditions is a* witness *of a difference in $\mathcal{T}_2$ w.r.t. $\mathcal{T}_1$.*

We note that the witness set $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ computes the *information gain* from $\mathcal{T}_1$ to $\mathcal{T}_2$ and the *information loss* from $\mathcal{T}_2$ to $\mathcal{T}_1$ for the signature Σ, where $\Sigma \subseteq \mathsf{sig}(\mathcal{T}_1) \cap \mathsf{sig}(\mathcal{T}_2)$. To compute all witnesses in $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ it is necessary to compute all Σ-entailments of $\mathcal{T}_2$ which are not entailed by $\mathcal{T}_1$. If $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \neq \emptyset$, it is typically infinite and, therefore, cannot be presented to the user as such [10]. However, a finite representation of $\mathsf{Diff}(\mathcal{T}_1, \mathcal{T}_2)$ can be computed via a Uniform Interpolation (UI) approach.

Definition 2 (Uniform Interpolation). *Let $\mathcal{T}$ be an $\mathcal{ELH}$-ontology. Let $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$ be a set of concept and role names. An $\mathcal{ELH}$-ontology $\mathcal{V}$ is a* Σ-uniform interpolant *of $\mathcal{T}$ iff the following conditions hold: (i) $\mathsf{sig}(\mathcal{V}) \subseteq \Sigma$ and (ii) for any $\mathcal{ELH}$-axiom α with $\mathsf{sig}(\alpha) \subseteq \Sigma$, $\mathcal{V} \models \alpha$ iff $\mathcal{T} \models \alpha$. In this case, the set Σ is called the* interpolation signature*.*

This means that uniform interpolants $\mathcal{V}$ have the same logical entailments as the given ontologies $\mathcal{T}$ up to Σ, and thus are the strongest Σ-entailments of $\mathcal{T}$. The problem of semantic difference can be related to that of UI as follows: $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) = \emptyset$ iff $\mathcal{T}_1 \models \mathcal{V}_2$, where $\mathcal{V}_2$ is a Σ-uniform interpolant of $\mathcal{T}_2$, for $\Sigma \subseteq \mathsf{sig}(\mathcal{T}_1) \cap \mathsf{sig}(\mathcal{T}_2)$. On the other hand, if $\mathcal{T}_1 \not\models \mathcal{V}_2$, this means that $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \neq \emptyset$, and every $\alpha \in \mathcal{V}_2$ with $\mathcal{T}_1 \not\models \alpha$ is a witness of $\mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$.

Definition 3 (UI-based Semantic Difference). *Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two $\mathcal{ELH}$-ontologies. Let Σ be a subset of the shared signature of $\mathcal{T}_1$ and $\mathcal{T}_2$. The* UI semantic difference *between $\mathcal{T}_1$ and $\mathcal{T}_2$ is the set $\mathsf{UI\text{-}Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of all $\mathcal{ELH}$-axiom α such that (i) $\mathsf{sig}(\alpha) \subseteq \Sigma$, (ii) $\alpha \in \mathcal{V}_2$ and (iii) $\mathcal{T}_1 \not\models \alpha$, where $\mathcal{V}_2$ is a Σ-uniform interpolant of $\mathcal{T}_2$. An axiom α satisfying these conditions is a* UI-witness *of a difference in $\mathcal{T}_2$ w.r.t. $\mathcal{T}_1$.*

Since any $\alpha \in \mathcal{V}_2$ is a logical entailment of $\mathcal{T}_2$, every UI-witness is a witness and $\mathsf{UI\text{-}Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2) \subseteq \mathsf{Diff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$. Since all the witnesses can in principle be computed from the deductive closure of a Σ-uniform interpolant $\mathcal{V}_2$ of $\mathcal{T}_2$, we can think of $\mathsf{UI\text{-}Diff}(\mathcal{T}_1, \mathcal{T}_2)$ as a representation of $\mathsf{Diff}(\mathcal{T}_1, \mathcal{T}_2)$. For $\mathcal{ELH}$, $\mathsf{UI\text{-}Diff}(\mathcal{T}_1, \mathcal{T}_2)$ is a *finite* representation of $\mathsf{Diff}(\mathcal{T}_1, \mathcal{T}_2)$. It follows from these considerations that the set $\mathsf{UI\text{-}Diff}(\mathcal{T}_1, \mathcal{T}_2)$ of UI-witnesses can be computed using the following algorithm:

Step (1): compute the Σ-uniform interpolant $\mathcal{V}_2$ of $\mathcal{T}_2$, for $\Sigma \subseteq \mathsf{sig}(\mathcal{T}_1) \cap \mathsf{sig}(\mathcal{T}_2)$, and then

Step (2): collect the axioms $\alpha \in \mathcal{V}_2$ not entailed by $\mathcal{T}_1$.

The first step can be done using a UI method/tool, and the second step can be done using an external DL reasoner.

3 Limitations of Existing Uniform Interpolation Methods

A few methods have been developed for various DLs. These methods include NUI [9], LETHE [14], UI-FAME [21] and the method developed by [10]. They are however designed for DLs that are either more expressive or less expressive than $\mathcal{ELH}$, the underlying language of typical large-scale ontologies, and therefore are not ideal tools to perform **Step (1)** of the above algorithm.

NUI [9] handles $\mathcal{ELH}$-ontologies restricted to terminologies. Hence, NUI is not an ideal tool to perform the UI step, given that many $\mathcal{ELH}$-ontologies are general $\mathcal{ELH}$-ontologies containing GCIs.[4]

LETHE, UI-FAME, as well as the method of [10] take the description logic $\mathcal{ALCH}$ (or some extensions of $\mathcal{ALCH}$) as the source and target languages. This means that, given any $\mathcal{ELH}$-ontology and an interpolation signature, the uniform interpolant computed by these methods always uses $\mathcal{ALCH}$-axioms (or the extensions), but not $\mathcal{ELH}$-axioms. The target language is different from the source language. This may lead to non-UI-witnesses being mistakenly collected into the UI-witness set. Hence, LETHE, UI-FAME, and [10] are not ideal tools to perform the UI step either.

The evaluations in [9] and [3] showed that NUI had excellent performance when interpolating for very small signatures, but it became problematic when applied to SNOMED CT, where often a large signature was considered. It was also shown in [9] that failures appeared with more frequency as the signature grew, and all failures were due to the memory overflow. In fact, performance issues are a common problem among all existing UI methods [3].

4 A Novel UI Method for $\mathcal{ELH}$

In this section we introduce a tailor-made method for computing uniform interpolants of $\mathcal{ELH}$-ontologies for **Step (1)**. In particular, we develop a new method for forgetting concept and role names from $\mathcal{ELH}$-ontologies.

[4] SNOMED CT used to be an acyclic $\mathcal{ELH}$-terminology, but began to include GCIs from its International 2019 January release.

Central to the forgetting method are two mutually independent calculi, namely a calculus for concept name elimination and a calculus for role name elimination. The process of computing uniform interpolants is to firstly deal with concept forgetting, then role forgetting, then treat introduced definers as regular concept names to carry out forgetting. In the remainder of this paper, we refer to the concept or role name under current consideration for forgetting as the *pivot*. The notation Σ and $\mathcal{F}$ is uniformly used to denote the interpolation signature and the forgetting signature, respectively. Let $\mathcal{T}$ denote an $\mathcal{ELH}$-ontology.

4.1 Calculus for Concept Name Elimination

Let $\mathsf{A} \in \mathsf{sig_C}(\mathcal{T})$ be the pivot concept. The calculus for eliminating A from $\mathcal{T}$ includes two steps that are executed in sequence. The first step is to transform $\mathcal{T}$ into A-*reduced form* (normalization), which is a specialized normal form to which some of the inference rules in the calculus (presented later) are applicable. The second step is to apply these inference rules to $\mathcal{T}$ to eliminate A.

Definition 4 (A-Reduced Form). *An $\mathcal{ELH}$ GCI is in* A-reduced form *if it has one of the following forms, where (i) $r, s \in \mathsf{N_R}$, (ii) C, D, E and F are concepts not containing A, (iii) $G \neq \mathsf{A}$ is an atomic concept or a concept of the form $\exists r.X$ for X a concept not containing A. An $\mathcal{ELH}$-ontology $\mathcal{T}$ is in* A-reduced form *if every A-axiom in $\mathcal{T}$ is a GCI in positive or negative A-reduced form.*

$$\text{I: } C \sqsubseteq \mathsf{A} \quad \text{II: } C \sqsubseteq \exists r.(\mathsf{A} \sqcap D) \quad \text{III: } \mathsf{A} \sqcap F \sqsubseteq G \quad \text{IV: } \exists s.(\mathsf{A} \sqcap E) \sqcap F \sqsubseteq G$$

Given any $\mathcal{ELH}$-ontology $\mathcal{T}$, one can compute in polynomial time an equisatisfiable $\mathcal{ELH}$-ontology in A-reduced form by applying exhaustively the following rules to the A-axioms in $\mathcal{T}$:

1. replace each $C \equiv C_1$ with $C \sqsubseteq C_1$ and $C_1 \sqsubseteq C$;
2. replace each $C \sqsubseteq C_1 \sqcap C_2$ with $C \sqsubseteq C_1$ and $C \sqsubseteq C_2$;
3. if $\exists s.C$ occurs somewhere on the left-hand side of a GCI in $\mathcal{T}$, where $s \in \mathsf{N_R}$ and C is a concept containing A, replace C with a fresh concept name $X \in \mathsf{N_C}$ and add $C \sqsubseteq X$ to $\mathcal{T}$;
4. if $\exists s.C$ occurs somewhere on the right-hand side of a GCI in $\mathcal{T}$, where $s \in \mathsf{N_R}$ and C is a concept containing A, replace C with a fresh concept name $X \in \mathsf{N_C}$ and add $X \sqsubseteq C$ to $\mathcal{T}$;
5. if A occurs in Z, where Z is a placeholder for the concept "C", "E", or "F" in the above A-reduced form, replace Z with a fresh concept name $X \in \mathsf{N_C}$ and add $Z \sqsubseteq X$ to $\mathcal{T}$;
6. if A occurs in Z, where Z is a placeholder for the concept "D" or "G"in the above A-reduced form, replace Z with a fresh concept name $X \in \mathsf{N_C}$ and add $X \sqsubseteq Z$ to $\mathcal{T}$.

The fresh concept names $X \in \mathsf{N_C}$ introduced in the above rules are called *definer names* or simply *definers* [15]. Definers are auxiliary symbols externally

introduced to facilitate the normalization of $\mathcal{T}$. We notice that if an axiom contains one $\exists$-restriction, at most one definer needs to be introduced, and if an axiom contains n $\exists$-restriction, at most n definer needs to be introduced. Therefore, the number of definers for the reduction is bounded by $O(n)$, for n the number of $\exists$-restriction in $\mathcal{T}$. Indeed, our definer introduction amounts to structural transformation[16,13].

Lemma 1. *For any $\mathcal{ELH}$-ontology $\mathcal{T}$, one can construct in polynomial time a normalized $\mathcal{ELH}$-ontology $\mathcal{T}'$ of polynomial size in $|\mathcal{T}|$ such that (i) $\mathsf{sig}(\mathcal{T}) \subseteq \mathsf{sig}(\mathcal{T}')$ and (ii) $\mathcal{T}' \models \mathcal{T}$, and for every model $\mathcal{I}$ of $\mathcal{T}$ there exists a model $\mathcal{J}$ of $\mathcal{T}'$ such that $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$ and $X^{\mathcal{I}} = X^{\mathcal{J}}$ for every $X \in \mathsf{sig}(\mathcal{T})$. $\mathcal{T}'$ is acyclic if $\mathcal{T}$ is acyclic.*

Lemma 1 states that definer introduction preserves the underlying logical models of the concepts and roles in the original axioms (Definition 2 holds). Normalized $\mathcal{ELH}$-ontologies in this sense are modifications of normalized terminologies as defined by [2]. For space reasons, we have to omit the proof of this standard operation. This lemma holds also for normalized $\mathcal{ELH}$-ontologies in role name elimination, described in the next subsection.

a. $C \sqsubseteq \mathsf{A}, \mathsf{A} \sqcap F \sqsubseteq G \implies C \sqcap F \sqsubseteq G$

b. $C \sqsubseteq \mathsf{A}, \exists s.(\mathsf{A} \sqcap E) \sqcap F \sqsubseteq G \implies \exists s.(C \sqcap E) \sqcap F \sqsubseteq G$

c. $C \sqsubseteq \exists r.(\mathsf{A} \sqcap D), \mathsf{A} \sqcap F_1 \sqsubseteq G_1, \ldots, \mathsf{A} \sqcap F_n \sqsubseteq G_n$
$\implies C \sqsubseteq \exists r.(G_1 \sqcap \ldots \sqcap G_n \sqcap D)$
provided that: $F_i \equiv \top$ for $1 \leq i \leq n$
$C \sqsubseteq \exists r.(\mathsf{A} \sqcap D), \mathsf{A} \sqcap F \sqsubseteq G_1, \ldots, \mathsf{A} \sqcap F \sqsubseteq G_n$
$\implies C \sqsubseteq \exists r.D, C \sqsubseteq \exists r.(G_1 \sqcap \ldots \sqcap G_n)$
provided that: $F \not\equiv \top$ and $\mathcal{T} \models D \sqsubseteq F$
$C \sqsubseteq \exists r.(\mathsf{A} \sqcap D), \mathsf{A} \sqcap F \sqsubseteq G \implies C \sqsubseteq \exists r.D$
provided that: $\mathcal{T} \not\models D \sqsubseteq F$

d. $C \sqsubseteq \exists r.(\mathsf{A} \sqcap D), \exists s.(\mathsf{A} \sqcap E) \sqcap F \sqsubseteq G$
$\implies C \sqsubseteq \exists r.D, C \sqcap F \sqsubseteq G$
provided that: $\mathcal{T} \models D \sqsubseteq E$ and $\mathcal{T} \models r \sqsubseteq s$
$C \sqsubseteq \exists r.(\mathsf{A} \sqcap D), \exists s.(\mathsf{A} \sqcap E) \sqcap F \sqsubseteq G \implies C \sqsubseteq \exists r.D$
provided that: $\mathcal{T} \not\models D \sqsubseteq E$ or $\mathcal{T} \not\models r \sqsubseteq s$

Fig. 1: Inference rules for concept name elimination

Once $\mathcal{T}$ is in A-reduced form, the second step is to eliminate A from $\mathcal{T}$ using the inference rules shown in Figure 1. The elimination of A is based on an exhaustive application of the inference rules to all (reduced) A-axioms in $\mathcal{T}$ to derive new logical entailments on A (and add them to $\mathcal{T}$) until $\mathcal{T}$ is saturated w.r.t. A, and then remove all A-axioms from $\mathcal{T}$. Inferences to reveal logical entailments of a name is often based on combining positive and negative occurrences

of the name, which is also reflected in the inference rules of this calculus. In the context of forgetting, ontologies are assumed to be consistent, so no contradiction would be derived. Also, termination is guaranteed because A do not occur in newly-derived entailments, meaning that there would be no recursive derivations and saturation can be reached in finite steps.

Specifically, Rule (a) and Rule (b) combines each GCI of Form I, which contains a positive occurrence of A, with each GCI of Form III and Form IV, respectively, which contain a negative occurrence of A; Rule (c) and Rule (d) combines each GCI of Form II, which contains a positive occurrence of A, with each GCI of Form III and Form IV, respectively.

Lemma 2. *The calculus for concept name elimination is sound.*

Proof (sketch). To prove that the calculus is sound is to prove that the output $\mathcal{OUT}$ of the calculus has the same logical entailments as its input $\mathcal{IN}$ up to the signature $\mathsf{sig}(\mathcal{IN})\backslash\{\mathsf{A}\}$.

The calculus for concept name elimination includes the normalization step and the inference step, so its soundness follows from the rules used in both steps. The first two normalization rules are standard transformations preserving logical equivalence. The latter four normalization rules are the structural transformation [15,20] which preserves all logical entailments in the signature $\mathsf{sig}(\mathcal{T})$ of the given ontology $\mathcal{T}$. We can also regard the latter four rules as the reverse operation of a monotonicity property called *Ackermann's Lemma* [1], which preserves equivalence up to the definers.The first two inference rules for concept name elimination are basically the binary resolution inference, so automatically we have that the conclusion of each rule has the same logical entailments as its premises up to the signature of the premises excluding the pivot. We prove soundness of the last two inference rules. To show Rule (c) is sound, we can appeal to the technique of unfolding [2] used in Tableaux reasoning. We prove the second case of Rule (c). $C \sqsubseteq \exists r.(\mathsf{A} \sqcap D)$ is directly equivalent to $C \sqsubseteq \exists r.D$ up to A. According to the side condition, we redefine D as $F \sqcap X$, where X is a fresh concept name. Then we have the premises $C \sqsubseteq \exists r.(\mathsf{A} \sqcap F \sqcap X)$ and $\mathsf{A} \sqcap F \sqsubseteq G$. Taking $\mathsf{A} \sqcap F$ as a whole, we obtain (via resolution or Ackermann's Lemma) $C \sqsubseteq \exists r.(G \sqcap X)$, which is equivalent to $C \sqsubseteq \exists r.G$ up to X. The other cases of Rule (c) and the two cases of Rule (d) can be proved similarly.

4.2 Calculus for Role Name Elimination

Let $\mathsf{r} \in \mathsf{sig}_{\mathsf{R}}(\mathcal{T})$ be the pivot role name. The calculus for role name elimination, as with that for concept name elimination, includes two steps executed in sequence. The first step is to transform $\mathcal{T}$ into another specialized normal form, namely r-*reduced form*, which generalizes all elementary forms of a concept/role inclusion where a role name r could occur.

Definition 5 (r-Reduced Form). *An inclusion is in* r-reduced form *if it has one of the following forms, where (i)* $s, r \in \mathsf{N_R}$*, (ii)* C*,* E*,* F*,* D *and* G *are*

concepts not containing r. *An* $\mathcal{ELH}$ *ontology* $\mathcal{T}$ *is in* r-reduced form *if every* r*-axiom in* $\mathcal{T}$ *is an inclusion in* r*-reduced form.*

$$\text{I: } s \sqsubseteq \mathsf{r} \quad \text{II: } C \sqsubseteq \exists \mathsf{r}.D \quad \text{III: } \mathsf{r} \sqsubseteq t \quad \text{IV: } F \sqcap \exists \mathsf{r}.E \sqsubseteq G$$

Observe that an r-role inclusion is naturally in r-reduced form. r-clauses not in r-reduced form can be transformed into the form by a trivial adaptation of the definer introduction for concept elimination, which inherits all its properties including Lemma 1.

e. $s \sqsubseteq \mathsf{r}, \mathsf{r} \sqsubseteq t \implies s \sqsubseteq t$
f. $s \sqsubseteq \mathsf{r}, F \sqcap \exists \mathsf{r}.E \sqsubseteq G \implies F \sqcap \exists s.E \sqsubseteq G$
g. $C \sqsubseteq \exists \mathsf{r}.D, \mathsf{r} \sqsubseteq t \implies C \sqsubseteq \exists t.D$
h. $C \sqsubseteq \exists \mathsf{r}.D, F \sqcap \exists \mathsf{r}.E \sqsubseteq G \implies F \sqcap C \sqsubseteq G$
provided that: $\mathcal{T} \models D \sqsubseteq E$

Fig. 2: Inference rules for role name elimination

Once $\mathcal{T}$ is in r-reduced form, the second step is to apply exhaustively the inference rules shown in Figure 2 to eliminate r from $\mathcal{T}$. The idea is analogous to that in concept name elimination, that is, to derive from the r-axioms all logical entailments not involving r, add them to $\mathcal{T}$, and then remove from $\mathcal{T}$ the r-axioms.

Lemma 3. *The calculus for role name elimination is sound.*

Proof (sketch). The first three inferences rules are generalizations of Ackermann's Lemma which preserve equivalence up to the interpolation signature. The last rule can be solved using unfolding [2]. From the side condition, we know that D can be defined with $E \sqcap X$, where X is a fresh concept name. Then D is replaced by $E \sqcap X$ in the premises, and now it is obvious that $\exists r.(E \sqcap X)$ and $\exists r.E$ are equivalent up to E. Because every instance of C is also an instance of $\exists r.D$ and $\exists r.E$, we have the conclusion $F \sqcap C \sqsubseteq G$, which is the strongest entailment of the premises for the remaining signature, as models of the concepts in the remaining signature are preserved.

Theorem 1. *For any* $\mathcal{ELH}$*-ontology* $\mathcal{T}$ *and an interpolation signature* $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$*, our UI method always terminates and returns an* $\mathcal{ELH}$ *ontology* $\mathcal{V}$*. If* $\mathcal{V}$ *does not contain any definers, then our method* succeeds *and* $\mathcal{V}$ *is a* Σ*-uniform interpolant of* $\mathcal{T}$*. For any* acyclic $\mathcal{ELH}$*-ontology* $\mathcal{T}$ *and a signature* $\Sigma \subseteq \mathsf{sig}(\mathcal{T})$*, our method always terminates and returns a* Σ*-uniform interpolant* $\mathcal{V}$ *of* $\mathcal{T}$*.*

Table 1: Statistics of Adapted Oxford-ISG

Oxford-ISG		Min	Max	Medium	Mean	90th Centile
I	$\lvert\mathsf{N_C}\rvert$	0	1582	86	191	545
	$\lvert\mathsf{N_R}\rvert$	0	332	10	29	80
	$\lvert\text{Onto}\rvert$	0	990	162	262	658
II	$\lvert\mathsf{N_C}\rvert$	200	5877	1665	1769	2801
	$\lvert\mathsf{N_R}\rvert$	0	887	11	34	61
	$\lvert\text{Onto}\rvert$	1008	4976	2282	2416	3937
III	$\lvert\mathsf{N_C}\rvert$	1162	9809	4042	5067	8758
	$\lvert\mathsf{N_R}\rvert$	1	158	4	23	158
	$\lvert\text{Onto}\rvert$	5112	9783	7277	7195	9179

5 Evaluation of the UI Method

To understand the practicality of our UI method, we implemented a prototype in Java using the OWL API Version 3.5.7,[5] and compared it with LETHE on a corpus of real-world ontologies.

We selected 488 ontologies from the Oxford ISG[6] snapshot with the size $|\text{Onto}|$ of TBox axioms not exceeding 10000. We further split the 488 ontologies into three subparts: PART I with $10 \leq |\text{Onto}| < 1000$, containing 355 ontologies, PART II with $1000 \leq |\text{Onto}| < 5000$, containing 108 ontologies, and PART III with $5000 \leq |\text{Onto}| \leq 10000$, containing 25 ontologies. This would provide clear clues for better understanding of the performance of our UI method for realistic ontologies of different size ranges. The selected ontologies were restricted to their $\mathcal{ELH}$-fragments by removing from them those axioms not expressible in $\mathcal{ELH}$. On average 8.9% of the axioms were thus dropped from the ontologies. Statistical information about the adapted ontologies is shown in Table 1. We repeat the tests three times for each ontology.

To fit with real-world applications, the experiments were conducted for two settings: forgetting respectively 10% and 30% of the concept and role names in the signature of each ontology. The forgetting signature was *randomly* chosen. The experiments were conducted on a laptop with an Intel Core i7-9750H processor, 6 cores running at up to 2.70 GHz, and 12 GB of DDR4-1600 MHz RAM. The timeout was limited to 300 seconds and heap space to 9GB.

The *success for forgetting* was defined as: (i) eliminating all the names in $\mathcal{F}$, (ii) not leaving any definers in the solutions, (iii) finished in the time and space limit. The results, shown in Table 2, are quite revealing in several ways. The most encouraging result is that our UI prototype succeeded in almost all test cases, and in most of these successful cases the forgetting was finished in an instant. In particular, compared to LETHE, our prototype fared considerably better w.r.t. success rates. Failures of our prototype were due to cyclic dependencies (over the names in $\mathcal{F}$) being present in the original ontologies and the timeout. Most

[5] http://owlcs.github.io/owlapi/
[6] http://krr-nas.cs.ox.ac.uk/ontologies/lib/

Table 2: Experimental results (Time (seconds): Time Consumption, Mem (MB): Memory Consumption, S-R: Success Rate, TO-R: Timeout Rate, MO-R: Memory Overflow Rate, Cyclic: Cyclic Dependency over $\mathcal{F}$)

			Time	Mem	S-R	TO-R	MO-R	Cyclic
LETHE	0.1	I	3.07	230.68	92.68	6.85	0.00	0.38
		II	7.12	957.65	68.21	31.17	0.00	0.31
		III	24.86	911.55	74.67	18.67	0.00	2.67
		Avg.	5.08	426.44	86.34	12.83	0.00	0.48
	0.3	I	1.50	230.14	84.60	13.62	0.00	1.78
		II	14.50	867.70	56.50	42.28	0.00	0.30
		III	57.30	1039.06	66.67	21.33	0.00	12
		Avg.	7.23	412.68	77.46	20.35	0.00	1.97
Our Prototype	0.1	I	0.08	75.56	99.53	0.00	0.00	0.47
		II	1.11	279.73	93.83	5.25	0.00	0.93
		III	1.02	275.77	92.01	1.33	0.00	2.67
		Avg.	**0.35**	**131.00**	**97.88**	**1.23**	0.00	0.68
	0.3	I	0.36	132.10	97.00	1.03	0.00	1.97
		II	2.75	466.60	91.36	7.72	0.00	0.00
		III	9.48	714.75	81.30	4.00	0.00	14.6
		Avg.	**1.36**	**235.97**	**94.95**	**2.66**	0.00	2.18

failures of LETHE were due to timeout. Our prototype outperformed LETHE in speed performance, and was at least five to ten times faster than LETHE. We conjecture (also evidenced by the results of memory consumption) this is most likely because LETHE uses an expensive definer introduction algorithm to flatten complex clauses, where the number of introduced definers is bounded by $O(2^n)$, for n the number of input clauses [15], whereas our UI method introduces definers when really necessary and introduces only linearly many definers.

The best way to verify our conjecture was to track the working process of LETHE but this was infeasible due to the unavailability of the source of LETHE. We then designed an alternative experiment; see Figure 3. In the Cartesian coordinate system, the x-axis denotes the number of GCIs in a test ontology fragment (e.g., the number of A-clauses when forgetting A), which reflects the *size* of the fragment, and the y-axis denotes the number of definers our UI prototype introduced in a test case, which reflects the *flatness* of the test ontology. Less definers normally came with flatter ontologies. The blue points mark the cases where our UI tool succeeded but LETHE failed. The orange points mark those where both succeeded. Now it is clear that the size of ontologies was not a problem for LETHE, but a complex internal structure with a large number of definers in demand for normalization would increase the probability of failure for LETHE.

6 Case Studies

SNOMED CT is presently the most comprehensive, multilingual clinical healthcare ontology in the world, and has been integrated into the knowledge base of many e-health vendors. SNOMED International[7] owns and maintains SNOMED

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

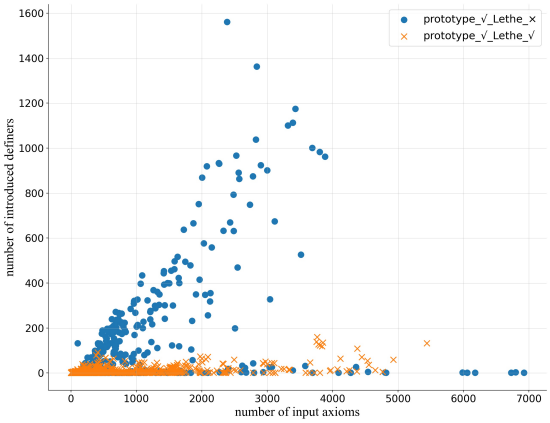

Fig. 3: Distributions of successful and failure cases

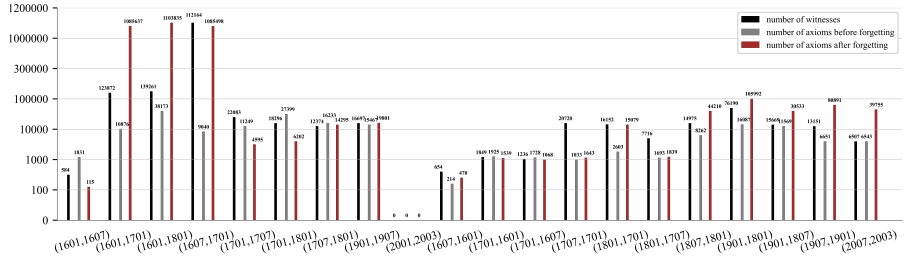

Fig. 4: Information gain and loss between different releases of SNOMED CT

CT on an ongoing basis, issuing releases of its International Edition at the end of January and July each year. In this section we studied how our UI method performs in practice for the task of tracking the semantic evolutionary changes in different versions of SNOMED CT. We computed the semantic difference between 15 consecutive international releases of SNOMED CT, as well as 5 non-consecutive international releases using a UI-Diff tool that employed our UI tool to perform **Step (1)** and the DL reasoner HermiT [5] to perform **Step (2)** of the UI-Diff algorithm.

Figure 4, where the x axis denotes different SNOMED CT versions combinations of which we want to compute semantic difference (1601 denotes 2016 January release, others are in a same way), and the y axis denotes the number of axioms, reflects the semantic changes over the evolution of the International SNOMED CT edition, i.e., the information gain $\mathsf{UI\text{-}Diff}_{\Sigma}(\mathcal{T}, \mathcal{T}')$ and the information loss $\mathsf{UI\text{-}Diff}_{\Sigma}(\mathcal{T}', \mathcal{T})$, where $\Sigma = \mathsf{sig}(\mathcal{T}) \cap \mathsf{sig}(\mathcal{T}')$. With the UI-witness set being successfully generated in all of the 20 comparison cases, our UI tool demonstrated superb performance for forgetting. Table 3 summarizes the metrics of the forgetting task corresponding to each comparison case, where $|\mathcal{F}_{\mathsf{C}}|$ and $|\mathcal{F}_{\mathsf{R}}|$ denote respectively the number of concept and role names to be forgotten, $|\sqsubseteq|$ the number of axioms in the given normalized ontology, and T(s) the time duration for forgetting. To the best of our knowledge, our UI prototype

Table 3: Metrics of the forgetting tasks on SNOMED CT

Tasks	$\lvert\mathcal{F}_{\mathsf{C}}\rvert$	$\lvert\mathcal{F}_{\mathsf{R}}\rvert$	$\lvert \sqsubseteq \rvert$	T(s)	Tasks	$\lvert\mathcal{F}_{\mathsf{C}}\rvert$	$\lvert\mathcal{F}_{\mathsf{R}}\rvert$	$\lvert \sqsubseteq \rvert$	T(s)
(1601,1607)	2599	1	614K	29	(1701,1601)	1921	1	610K	330
(1601,1701)	7695	3	620K	1894	(1701,1607)	1795	1	616K	306
(1601,1801)	23.4K	33	643K	2919	(1707,1701)	614	0	629K	89
(1607,1701)	5115	2	623K	2151	(1801,1701)	1378	1	628K	173
(1701,1707)	12.3K	18	637K	438	(1801,1707)	779	3	648K	100
(1701,1801)	17.2K	30	651K	1161	(1807,1801)	6955	0	655K	351
(1707,1801)	4980	14	664K	534	(1901,1801)	9389	1	652K	1784
(1901,1907)	4423	1	755K	879	(1901,1807)	3034	1	693K	861
(2001,2003)	0	0	770K	0	(1907,1901)	3142	0	731K	740
(1607,1601)	145	0	612K	190	(2007,2003)	2074	0	768K	637

is so far the only tool capable of forgetting 10K+ number of concepts and roles from ontologies as large as containing 600K+ logical statements. All forgetting tasks were finished within a reasonable period of time. This provides ontology engineers with a powerful tooling support to create views of industrial-scale ontologies.

7 Conclusion and Future Work

We developed a novel, tailor-made UI method. On a large corpus of adapted Oxford ISG ontologies, our UI method has shown superb performance and superiority over LETHE (the state-of-the-art UI tool) by a large margin. Case studies on different versions of SNOMED CT have verified the viability of our semantic difference algorithm as back-end technology in e-health vendors' knowledge base interface for their main concerns of content analysis and quality assurance.

An immediate future step is to find justifications [6] for the UI-witnesses so as to pinpoint an (ideally minimum) set of axioms that accounts for each witness. [3] have shown that incorporating the technique of modularization in the forgetting procedure could improve the performance of UI. We then mark an attempt of this incorporation as future work for improvement.

Acknowledgements

The authors would like to thank the reviewers for their insightful comments and good suggestions. This work was supported by National Natural Science Foundation of China (grant 62006114) and Open Research Projects of Zhejiang Lab (grant 2021KE0AB08). Ghadah Alghamdi's Ph.D. program is supported by scholarship funding from the Saudi Arabian Cultural Bureau (SACB) (grant 1068191434).

References

1. Wilhelm Ackermann. Untersuchungen úber das Eliminationsproblem der mathematischen Logik. *Mathematische Annalen*, 110(1):390–413, 1935.

2. Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.
3. Jieying Chen, Ghadah Alghamdi, Schmidt Renate A., Dirk Walther, and Yongsheng Gao. Ontology extraction for large ontologies via modularity and forgetting. In *K-CAP'19*, pages 45–52. ACM, 2019.
4. Silvio Ghilardi, Carsten Lutz, and Frank Wolter. Did I Damage My Ontology? A Case for Conservative Extensions in Description Logics. In *Proc. KR'06*, pages 187–197. AAAI Press, 2006.
5. Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. HermiT: An OWL 2 Reasoner. *J. Autom. Reasoning*, 53(3):245–269, 2014.
6. Matthew Horridge. *Justification based explanation in ontologies*. PhD thesis, The University of Manchester, UK, 2011.
7. Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Supporting concurrent ontology development: Framework, algorithms and tool. *Data Knowl. Eng.*, 70(1):146–164, 2011.
8. Boris Konev, Michel Ludwig, Dirk Walther, and Frank Wolter. The Logical Difference for the Lightweight Description Logic $\mathcal{EL}$. *J. Artif. Intell. Res.*, 44:633–708, 2012.
9. Boris Konev, Dirk Walther, and Frank Wolter. Forgetting and Uniform Interpolation in Large-Scale Description Logic Terminologies. In *Proc. IJCAI'09*, pages 830–835. IJCAI/AAAI Press, 2009.
10. Michel Ludwig and Boris Konev. Practical Uniform Interpolation and Forgetting for $\mathcal{ALC}$ TBoxes with Applications to Logical Difference. In *Proc. KR'14*. AAAI Press, 2014.
11. C. Lutz and F. Wolter. Foundations for Uniform Interpolation and Forgetting in Expressive Description Logics. In *Proc. IJCAI'11*, pages 989–995. IJCAI/AAAI Press, 2011.
12. Carsten Lutz and Frank Wolter. Deciding inseparability and conservative extensions in the description logic $\mathcal{EL}$. *J. Symb. Comput.*, 45(2):194–228, 2010.
13. Boris Motik. *Reasoning in description logics using resolution and deductive databases*. PhD thesis, Karlsruhe Institute of Technology, Germany, 2006.
14. Koopmann P. and Schmidt R. A. LETHE: Saturation-based reasoning for non-standard reasoning tasks. In *Proc. DL'15*, volume 1387 of *CEUR Workshop Proceedings*, pages 23–30. CEUR-WS.org, 2015.
15. Koopmann Patrick. *Practical Uniform Interpolation for Expressive Description Logics*. PhD thesis, The University of Manchester, UK, 2015.
16. John Alan Robinson and Andrei Voronkov, editors. *Handbook of Automated Reasoning (in 2 volumes)*. Elsevier and MIT Press, 2001.
17. Alessandro Solimando, Ernesto Jiménez-Ruiz, and Giovanna Guerrini. Minimizing conservativity violations in ontology alignments: algorithms and evaluation. *Knowl. Inf. Syst.*, 51(3):775–819, 2017.
18. Kent A. Spackman. SNOMED RT and SNOMED CT. promise of an international clinical ontology. *M.D. Computing 17*, 2000.
19. Albert Visser. *Bisimulations, Model Descriptions and Propositional Quantifiers*. Logic Group Preprint Series. Utrecht University, 1996.
20. Yizheng Zhao. *Automated Semantic Forgetting for Expressive Description Logics*. PhD thesis, The University of Manchester, UK, 2018.
21. Yizheng Zhao, Ghadah Alghamdi, Schmidt Renate A., Hao Feng, Giorgos Stoilos, Damir Juric, and Mohammad Khodadadi. Tracking Logical Difference in Large-Scale Ontologies: A Forgetting-Based Approach. In *Proc. AAAI'19*, pages 3116–3124. AAAI Press, 2019.