Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Small Is Beautiful: Computing Minimal Equivalent $\mathcal{EL}$ Concepts

Nadeschda Nikitina
University of Oxford
nadeschda.nikitina@cs.ox.ac.uk

Patrick Koopmann
University of Dresden
patrick.koopmann@tu-dresden.de

Abstract

In this paper, we present an algorithm and a tool for computing minimal, equivalent $\mathcal{EL}$ concepts wrt. a given ontology. Our tool can provide valuable support in manual development of ontologies and improve the quality of ontologies automatically generated by processes such as uniform interpolation, ontology learning, rewriting ontologies into simpler DLs, abduction and knowledge revision. Deciding whether there exist equivalent $\mathcal{EL}$ concepts of size less than k is known to be an NP-complete problem. We propose a minimisation algorithm that achieves reasonable computational performance also for larger ontologies and complex concepts. We evaluate our tool on several bio-medical ontologies with promising results.

Introduction

Logics allow equivalent facts to be expressed in many different ways. The fact that ontologies are developed by a number of different people and grow over time can lead to concepts that are more complex than necessary. For example, below is a simplified definition of the medical concept `Clotting` from the Galen ontology (Rector et al. 1994):

$$\begin{aligned}\texttt{Clotting} \equiv\ & \exists\,\texttt{actsSpecificallyOn}.\\ & (\texttt{Blood} \sqcap \exists\,\texttt{hasPhysicalState}.\\ & \quad (\texttt{PhysicalState} \sqcap \exists\,\texttt{hasState}.\texttt{Liquid})) \sqcap\\ & \exists\,\texttt{hasOutcome}.\texttt{SolidBlood})\end{aligned}$$

Galen also defines concepts `LiquidBlood` and `LiquidState` by means of the following axioms:

$$\begin{aligned}\texttt{LiquidBlood} &\equiv \texttt{Blood} \sqcap \exists\,\texttt{hasPhysicalState}.\texttt{LiquidState}\\ \texttt{LiquidState} &\equiv \texttt{PhysicalState} \sqcap \exists\,\texttt{hasState}.\texttt{Liquid}\end{aligned}$$

Using these two concepts, we can find a more concise definition for `Clotting` and replace the initial one while preserving all logical consequences of the ontology:

$$\begin{aligned}\texttt{Clotting} \equiv\ & \exists\,\texttt{actsSpecificallyOn}.\texttt{LiquidBlood} \sqcap\\ & \exists\,\texttt{hasOutcome}.\texttt{SolidBlood}\end{aligned}$$

The two definitions of `Clotting` differ in various aspects. In addition to the fact that they use different sets of terms, we observe that the first is *structurally more complex* – it contains more occurrences of logical constructs such as intersection and existential quantification. Furthermore, we notice that the first definition introduces redundancy within the ontology. Both unnecessary structural complexity and redundancy complicate the maintenance of the ontology and hinder understanding.

Unnecessary structural complexity and redundancy are not unique to hand-crafted ontologies. On the contrary, they are a common side effect of processes that generate ontologies and concept expressions automatically. Examples of such processes include

computing least common subsumers (Turhan and Zarrieß 2013), *concept unification* (Baader, Borgwardt, and Morawska 2012), *uniform interpolation* (Nikitina and Rudolph 2012; Lutz, Seylan, and Wolter 2012), *ontology learning* (Konev, Ozaki, and Wolter 2016; Lehmann and Hitzler 2010), *rewriting ontologies into less expressive logics* (Carral et al. 2014; Lutz, Piro, and Wolter 2011), *abduction* (Du, Wang, and Shen 2015; Klarman, Endriss, and Schlobach 2011), and *knowledge revision* (Grau, Kharlamov, and Zheleznyakov 2012; Qi, Liu, and Bell 2006). Usually, such tools rely on heuristics that reduce the amount of redundancy within generated concepts. In this paper, we present a method that can entirely eliminate redundancy from $\mathcal{EL}$ concepts by computing equivalent concepts of minimal size, where concept size is defined as the number of occurrences of concept and role symbols. While approaches to related problems exist, including computing minimal subsets of ontologies (Grimm and Wissmann 2011), rewriting $\mathcal{ALE}$ and $\mathcal{ALN}$ concepts using terminologies (Baader, Küsters, and Molitor 2000) and computing minimal ontologies in a fragment of $\mathcal{EL}$ (Nikitina and Schewe 2013b), to the best of our knowledge, this is the first method computing minimal equivalent concepts wrt. $\mathcal{EL}$ ontologies.

While it is theoretically possible to compute minimal equivalent concepts using a naive brute-force approach that evaluates all possible concepts with the qualifying signature, it is challenging to compute the solution efficiently. In order to make concept minimisation feasible in practice, it is necessary to restrict the set of candidate concepts to those that are semantically related to the initial concept. The foundation of our method is a simple approach based on regular tree grammars in which we restrict candidate concepts to

subsumers of the concept in question. Thereby, we significantly reduce the number of candidates. We further narrow down the search space by using *dynamic derivation rules*—derivation rules that evolve during minimisation. Rather than applying the same set of rules to the same non-terminal over the entire course of minimisation, we compute an updated set of derivation rules on-demand for a particular context and a particular non-terminal based on the information gathered throughout the minimisation process. During the evaluation, we found that the average number of derivation rules applied to each non-terminal over the course of minimisation decreased by 8 orders of magnitude due to the use of dynamic derivation rules.

The evaluation confirms the feasibility of concept minimisation in practice. The computation of minimal equivalent concepts took on average 5 minutes for concepts from Snomed CT (Stearns et al. 2001), and just a few seconds for concepts from other ontologies including NCI Thesaurus (Sioutos et al. 2007) and Galen. We conclude that our tool would be a valuable new feature within ontology editors such as Protégé (Musen 2013).

Preliminaries

In this section, we formally introduce the description logic $\mathcal{EL}$. Let N_C and N_R be countably infinite and mutually disjoint sets called *concept symbols* and *role symbols*, respectively. $\mathcal{EL}$ *concepts* C are defined by

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

where A and r range over $N_C \cup \{\top\}$ and N_R, respectively. In the following, C, D, E, F and G can denote arbitrary concepts, while A, B can only denote concept symbols or $\top$. An *ontology* consists of *concept inclusion* axioms $C \sqsubseteq D$ and *concept equivalence* axioms $C \equiv D$, the latter used as a shorthand for the mutual inclusion $C \sqsubseteq D$ and $D \sqsubseteq C$.[1] The *signature* of an $\mathcal{EL}$ concept C, an axiom α or an ontology $\mathcal{O}$, denoted by $\mathrm{sig}(C)$, $\mathrm{sig}(\alpha)$ or $\mathrm{sig}(\mathcal{O})$, respectively, is the set of concept and role symbols occurring in it. To distinguish between the set of concept symbols and the set of role symbols, we use $\mathrm{sig}_C(\cdot)$ and $\mathrm{sig}_R(\cdot)$, respectively.

Further, we use the notation $\mathrm{sub}(C)$ to denote the set of all *subconcepts* of C, defined as follows. For $C \in N_C \cup \{\top\}$, we have $\mathrm{sub}(C) = \emptyset$. For $C = C_1 \sqcap \cdots \sqcap C_n$ with C_i being a non-conjunction, we have $\mathrm{sub}(C) = \{C_1, \cdots, C_n\} \cup \mathrm{sub}(C_1) \cup \cdots \cup \mathrm{sub}(C_n)$. For $C = \exists r.C_1$, we have $\mathrm{sub}(C) = \{C_1\} \cup \mathrm{sub}(C_1)$. We extend the above notion to axioms and ontologies as follows. For two concepts C_1, C_2, we have $\mathrm{sub}(C_1 \sqsubseteq C_2) = \{C_1, C_2\} \cup \mathrm{sub}(C_1) \cup \mathrm{sub}(C_2)$. For $\mathcal{O} = \{\alpha_1, \cdots, \alpha_n\}$, we have $\mathrm{sub}(\mathcal{O}) = \bigcup \mathrm{sub}(\alpha_i)$.

Next, we recall the semantics of the description logic constructs introduced above, which is defined by the means of interpretations. An *interpretation* $\mathcal{I}$ is given by a set $\Delta^{\mathcal{I}}$, called the *domain*, and an *interpretation function* $\cdot^{\mathcal{I}}$ assigning to each concept $A \in N_C$ a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and to each role $r \in N_R$ a subset $r^{\mathcal{I}}$ of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The interpretation of $\top$ is fixed to $\Delta^{\mathcal{I}}$. The interpretation of arbitrary $\mathcal{EL}$ concepts is defined inductively via $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$ and $(\exists r.C)^{\mathcal{I}} = \{x \mid (x, y) \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}} \text{ for some } y\}$. An interpretation $\mathcal{I}$ *satisfies* an axiom $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. $\mathcal{I}$ is a *model* of an ontology $\mathcal{O}$, if it satisfies all axioms in $\mathcal{O}$. We say that $\mathcal{O}$ entails an axiom α (in symbols, $\mathcal{O} \models \alpha$), if α is satisfied by all models of $\mathcal{O}$. For $\mathcal{EL}$ concepts C, D such that $\mathcal{O} \models C \sqsubseteq D$, we call C a *subsumee* of D and D a *subsumer* of C.

Regular Tree Grammars Next, we recall regular tree grammars on ranked ordered trees. A *ranked alphabet* $\mathcal{F}$ is a set of pairs of alphabet symbols and arities from $\mathbb{N}$. We use superscripts to denote the corresponding arity of an alphabet symbol (if it is not 0), e.g., $f^2(g^1(a), a)$. For the sake of simplicity, we allow the same alphabet symbol to assume different arities within the same alphabet, e.g., $\mathcal{F} = \{g^2, g^1, \cdots\}$. The set of ground terms over the alphabet $\mathcal{F}$ (which are also simply referred to as *trees*) is denoted by $T(\mathcal{F})$. Let $\mathcal{X}$ be a set of variables. Then, $T(\mathcal{F}, \mathcal{X})$ denotes the set of terms over the alphabet $\mathcal{F}$ and the set of variables $\mathcal{X}$. A term $\mathcal{C} \in T(\mathcal{F}, \mathcal{X})$ containing each variable from $\mathcal{X}$ at most once is called a *context*. A *regular tree grammar* $G = (\mathfrak{n}_S, \mathcal{N}, \mathcal{F}, R)$ is composed of a *start symbol* $\mathfrak{n}_S$, a set $\mathcal{N}$ of *non-terminal symbols* (of arity 0) with $\mathfrak{n}_S \in \mathcal{N}$, a ranked alphabet $\mathcal{F}$ of *terminal symbols* such that $\mathcal{F} \cap \mathcal{N} = \emptyset$, and a set R of derivation rules, each of which is of the form $\mathfrak{n} \to t$ where $\mathfrak{n}$ is a non-terminal from $\mathcal{N}$ and t is a term from $T(\mathcal{F} \cup \mathcal{N})$. Let $\mathcal{X}$ be a set of variables disjoint from the ranked alphabet $\mathcal{F} \cup \mathcal{N}$ with $X \in \mathcal{X}$. Given a regular tree grammar $G = (\mathfrak{n}_S, \mathcal{N}, \mathcal{F}, R)$, the derivation relation $\to_G$ associated with G is a relation on terms from $T(\mathcal{F} \cup \mathcal{N})$ such that $s \to_G t$ if and only if there is a rule $\mathfrak{n} \to t' \in R$ and there is a context $\mathcal{C} \in T(\mathcal{F} \cup \mathcal{N}, \{X\})$ such that $s = \mathcal{C}[\mathfrak{n}/X]$ and $t = \mathcal{C}[t'/X]$. The subset of $T(\mathcal{F} \cup \mathcal{N})$ which can be generated by successive derivations starting with the start symbol is denoted by $L_u(G) = \{t \in T(\mathcal{F} \cup \mathcal{N}) \mid \mathfrak{n}_S \to_G^+ t\}$ where $\to_G^+$ is the transitive closure of $\to_G$. We omit the subscript G when the grammar G is clear from the context or is arbitrary. The language generated by G is denoted by $L(G) = T(\mathcal{F}) \cap L_u(G)$. For further details on regular tree grammars, we refer the reader to (Comon et al. 2008).

Minimising Concepts

We define the *size* of concepts in an ontology $\mathcal{O}$ as the number of role and concept symbol occurrences:

- $\mathrm{s}(A) = 1$ for $A \in \mathrm{sig}_C(\mathcal{O}) \cup \{\top\}$;
- $\mathrm{s}(\exists r.C) = \mathrm{s}(C) + 1$ for a concept C and a role $r \in \mathrm{sig}_R(\mathcal{O})$;
- $\mathrm{s}(C_1 \sqcap C_2) = \mathrm{s}(C_1) + \mathrm{s}(C_2)$ for concepts C_1, C_2.

Given an $\mathcal{EL}$ ontology $\mathcal{O}$ and a concept C, deciding whether a concept of size less than k equivalent to C wrt. $\mathcal{O}$ exists, is an NP-complete problem (Nikitina and Schewe 2013a). Thus, while it is possible to find a simple approach that works in theory, minimising $\mathcal{EL}$ concepts wrt. an ontology within a reasonable time is challenging. Consider the following example:

[1] While ontologies in general can also include a specification of individuals with the corresponding concept and role assertions, in this paper we concentrate on concept inclusion and equivalence axioms.

Example 1. *The ontology $\mathcal{O}_{ex}$ consists of the following axioms:*

$$A_1 \sqcap A_2 \sqcap A_3 \sqsubseteq \exists r.(\exists s.A_3 \sqcap A_4) \quad (1)$$

$$\exists r.A_4 \sqsubseteq A_1 \quad (2)$$

$$A_1 \sqsubseteq A_3 \quad (3)$$

Let $C_{ex} = A_2 \sqcap \exists r.(\exists s.A_3 \sqcap A_4)$. If we look for a minimal concept D_{ex} equivalent to C_{ex} wrt. $\mathcal{O}_{ex}$ for the purpose of substituting C_{ex} in an axiom $\alpha \notin \mathcal{O}_{ex}$, we find $A_1 \sqcap A_2$.

In order to find D_{ex}, we could theoretically generate all concepts C' of size up to $\mathrm{s}(C_{\text{ex}}) - 1$ from the ontology signature $\mathrm{sig}(\mathcal{O}_{\text{ex}})$ and test for each of those 265 concepts whether $\mathcal{O}_{\text{ex}} \models C_{\text{ex}} \equiv C'$. While this would work in the case of our simple example concept and ontology, for larger concepts and ontologies with large signatures, this approach is not feasible in practice.

In order to achieve higher efficiency, we need to restrict the set of candidate concepts to those that are semantically related to C_{ex}. Next, we discuss an approach in which every candidate concept is a subsumer of the concept to be minimised.

Computing Subsumers

Our computation of subsumers is inspired by the approach to uniform interpolation presented by Nikitina et al. (Nikitina and Rudolph 2014). We construct *subsumer grammars* —a set of regular tree grammars that generate subsumers of subconcepts of $\mathcal{O}$. Since equivalent concepts have the same subsumers, we group subconcepts of $\mathcal{O}$ into *equivalence classes* $\mathcal{E}_\mathcal{O} = \{E \subseteq \mathrm{sub}(\mathcal{O}) \mid \forall C_1, C_2 \in E : \mathcal{O} \models C_1 \equiv C_2\}$ such that for each $D \in \mathrm{sub}(\mathcal{O})$ there is an $E \in \mathcal{E}_\mathcal{O}$ with $D \in E$. We then assign a single non-terminal $\mathfrak{n}_E$ and a subsumer grammar, denoted by $G^{\sqsubseteq}(\mathcal{O}, E)$, to each equivalence class E from $\mathcal{E}_\mathcal{O}$. We denote the entire set of non-terminals used in all subsumer grammars of $\mathcal{O}$ by $\mathcal{N}^\mathcal{O} = \{\mathfrak{n}_E \mid E \in \mathcal{E}_\mathcal{O}\}$.

Within the subsumer grammars, we use the ranked alphabet $\mathcal{F}^{\mathcal{EL}} = \mathrm{sig}_C(\mathcal{O}) \cup \{\top\} \cup \{\exists r^1 \mid r \in \mathrm{sig}_R(\mathcal{O})\} \cup \{\sqcap^i \mid 2 \leq i\}$, where $\top$ and concept symbols in $\mathrm{sig}_C(\mathcal{O})$ are constants, $\exists r^1$ for $r \in \mathrm{sig}_R(\mathcal{O})$ are unary functions and $\sqcap^i$ are functions of arity greater than 2. For brevity, we omit the arity of the above functions if it is clear from the context. We use the notation $\bigsqcap(S)$ to refer to terms constructed from a set $S \subseteq \mathcal{E}_\mathcal{O}$ as follows: $\bigsqcap(S) = \top$ if $S = \emptyset$, $\bigsqcap(S) = \mathfrak{n}_E$ if $S = \{E\}$ and $\bigsqcap(S) = \sqcap(\mathfrak{n}_{E_1}, \cdots, \mathfrak{n}_{E_n})$ if $S = \{E_1, \cdots, E_n\}$.

In the subsequent definition, we use the following two sets for constructing each subsumer grammar $G^{\sqsubseteq}(\mathcal{O}, E)$:

1. The set $\mathcal{S}^{\not\exists}_E$ of *subsumer successors* —a set of equivalence classes that contain subsumers of concepts from E. Since equivalence classes consisting of a single conjunction are not required to compute all subsumers (as shown later on in Lemma 1), but make our computations more expensive, we exclude those equivalence classes from $\mathcal{S}^{\not\exists}_E$. The set $\mathcal{S}^{\not\exists}_E$ is given by $\{E' \in \mathcal{E}_\mathcal{O} \mid E \neq E', E' \neq \{C_1 \sqcap C_2\}$ for some concepts C_1, C_2, and there exist $C \in E, C' \in E'$ such that $\mathcal{O} \models C \sqsubseteq C'\}$. In Table 1, we show the values of $\mathcal{S}^{\not\exists}_E$ for each $E \in \mathcal{E}_{\mathcal{O}'_{\text{ex}}}$ with $\mathcal{O}'_{\text{ex}} = \mathcal{O}_{\text{ex}} \cup \{C_{\text{ex}} \sqsubseteq \top\}$ from Example 1.
2. The set Ex_E of *existentially qualified successors*—a set of role and equivalence class pairs representing each existentially qualified expression from E. Ex_E is given by $\{\langle r, E'\rangle \mid r \in \mathrm{sig}_R(\mathcal{O}), E' \in \mathcal{E}_\mathcal{O}$ such that there exists a concept $C' \in E'$ with $\exists r.C' \in E\}$. In Example 1, there are three non-empty sets Ex_{E_i}, namely $\mathrm{Ex}_{E_6} = \{\langle r, E_3\rangle\}$, $\mathrm{Ex}_{E_8} = \{\langle r, E_7\rangle\}$ and $\mathrm{Ex}_{E_9} = \{\langle r, E_4\rangle\}$.

E	elements of E	$\mathcal{S}^{\not\exists}_E$
$E_\top$	$\{\top\}$	$\emptyset$
E_1	$\{A_1\}$	$E_\top, E_3$
E_2	$\{A_2\}$	$E_\top$
E_3	$\{A_3\}$	$E_\top$
E_4	$\{A_4\}$	$E_\top$
E_5	$\{C_{\text{ex}}, A_1 \sqcap A_2 \sqcap A_3\}$	$E_\top, E_1, E_2, E_3, E_8, E_9$
E_6	$\{\exists s.A_3\}$	$E_\top$
E_7	$\{\exists s.A_3 \sqcap A_4\}$	$E_\top, E_4, E_6$
E_8	$\{\exists r.(\exists s.A_3 \sqcap A_4)\}$	$E_\top, E_1, E_3, E_9$
E_9	$\{\exists r.A_4\}$	$E_\top, E_1, E_3$

Table 1: Values of $\mathcal{S}^{\not\exists}_E$ for $E \in \mathcal{E}_{\mathcal{O}'_{\text{ex}}}$ with $\mathcal{O}'_{\text{ex}} = \mathcal{O}_{\text{ex}} \cup \{C_{\text{ex}} \sqsubseteq \top\}$.

$$\mathfrak{n}_{E_5} \to \sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}) \quad (4)$$

$$\sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}) \to \sqcap(A_1, \mathfrak{n}_{E_2}) \quad (5)$$

$$\sqcap(A_1, \mathfrak{n}_{E_2}) \to \sqcap(A_1, A_2) \quad (6)$$

Figure 1: Derivation of the term $t_{D_{\text{ex}}} = \sqcap(A_1, A_2)$ from $\mathfrak{n}_{E_5}$ given in Example 1.

We construct subsumer grammars from $\mathcal{S}^{\not\exists}_E$ and Ex_E for a particular $\mathcal{EL}$ ontology $\mathcal{O}$ as follows:

Definition 1. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology and $E_0 \in \mathcal{E}_\mathcal{O}$. A* subsumer grammar *$G^{\sqsubseteq}(\mathcal{O}, E_0)$ for E_0 wrt. $\mathcal{O}$ is given by $(\mathfrak{n}_{E_0}, \mathcal{N}^\mathcal{O}, \mathcal{F}^{\mathcal{EL}}, R^{\sqsubseteq})$, where $R^{\sqsubseteq}$ includes the following rules for each $E \in \mathcal{E}_\mathcal{O}$:*

(R1) $\mathfrak{n}_E \to A$ *for each* $A \in E \cap (\mathrm{sig}_C(\mathcal{O}) \cup \{\top\})$;

(R2) $\mathfrak{n}_E \to \bigsqcap(S)$ *for each* $S \subseteq \mathcal{S}^{\not\exists}_E$ *with* $S \neq \emptyset$;

(R3) $\mathfrak{n}_E \to \exists r(\mathfrak{n}_{E_1})$ *for each* $\langle r, E_1\rangle \in \mathrm{Ex}_E$.

Rules of type R1 have a concept symbol or $\top$ on the right-hand side and are used for deriving ground terms. Within R2, we introduce a rule for each element of $2^{\mathcal{S}^{\not\exists}_E} \setminus \{\emptyset\}$, thereby covering all possibilities to introduce a conjunction within a subsumer term or simply replace a non-terminal by another representing a more general term. Rules of type R3 generate existentially qualified terms for each element of E that is an existentially qualified expression. If we construct the set of derivation rules $R^{\sqsubseteq}$ from the values of $\mathcal{S}^{\not\exists}_{E_i}$ and Ex_{E_i} given in Example 1, we can derive the term $t_{D_{\text{ex}}} = \sqcap(A_1, A_2)$ using the grammar $G^{\sqsubseteq}(\mathcal{O}'_{\text{ex}}, E_5)$ with $\mathcal{O}'_{\text{ex}} = \mathcal{O}_{\text{ex}} \cup \{C_{\text{ex}} \sqsubseteq \top\}$ as shown in Fig. 1.

As stated in the lemma below, this grammar-based generation of subsumers for subconcepts of $\mathcal{EL}$ ontologies is sound and complete.

Lemma 1. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology, $E \in \mathcal{E}_\mathcal{O}$ and D a concept with $D \in E$.*

1. *Let D' be a subsumer of D wrt. $\mathcal{O}$. Then, there exists a syntactic variant[2] D'' of D' with the corresponding term representation $t_{D''}$ such that $t_{D''} \in L(G^{\sqsubseteq}(\mathcal{O}, E))$.*
2. *Let D' be an $\mathcal{EL}$ concept with the corresponding term representation $t_{D'}$ such that $t_{D'} \in L(G^{\sqsubseteq}(\mathcal{O}, E))$. Then, D' is a subsumer of D wrt. $\mathcal{O}$.*

Dynamic Derivation Rules

Since the above grammars compute all subsumers of a concept D, a large proportion of derived concepts is neither equivalent to D nor minimal in size. We can further reduce the number of candidates and the number of rules applied by making the set of derivation rules *dynamic*— allowing it to evolve during minimisation. Rather than applying the same set of rules to the same non-terminal over the entire course of minimisation, we compute a set of rules that is specific to a particular context and non-terminal and incorporates our requirements concerning size and equivalence to D.

For convenience, we extend the notion of size to terms and contexts as follows: $\mathrm{s}(\mathfrak{n}_E) = 1$ for a non-terminal $\mathfrak{n}_E$, $\mathrm{s}(X) = 1$ for a variable X, $\mathrm{s}(\sqcap(t_1, \cdots, t_n)) = \sum_{1 \leq i \leq n} \mathrm{s}(t_i)$ for terms t_i and $\mathrm{s}(\exists r(t)) = \mathrm{s}(t) + 1$ for a role r and a term t.

Additionally, we use the notation $\mathrm{con}^f(t)$ to refer to the concept representation of a term t wrt. a representative selection function $f : \mathcal{E}_\mathcal{O} \to \mathrm{sub}(\mathcal{O}) \cup \{\top\}$ that assigns a representative $C \in E$ to each $E \in \mathcal{E}_\mathcal{O}$. If the choice of representatives from equivalence classes is irrelevant in a particular context, we omit the superscript f to indicate that the concept representation $\mathrm{con}(t)$ is based on an arbitrary representative selection function.

Dynamic derivation rules are motivated by the following observations regarding subsumer grammars:

1. Terms never become smaller over the course of derivation. Thus, once an non-ground term t reaches an unacceptable size, we can discard all terms that can be derived from t due to their size. We refer to this property of subsumer grammars as *size monotonicity* and use it to filter out rules where the term on the right-hand side is too large. For instance, once we have derived the term $t = \sqcap(\mathfrak{n}_{E_2}, \exists r(\sqcap(\mathfrak{n}_{E_6}, \mathfrak{n}_{E_4})))$ from $\mathfrak{n}_{E_5}$ in Example 1, we can skip the rule $\mathfrak{n}_{E_6} \to \exists r(\mathfrak{n}_{E_3})$, since the resulting term $t' = \sqcap(\mathfrak{n}_{E_2}, \exists r(\sqcap(\exists r(\mathfrak{n}_{E_3}), \mathfrak{n}_{E_4})))$ would become as large as C_{ex}.
2. Terms never become more specific over the course of derivation. Thus, if we find that $\mathcal{O} \not\models D \equiv \mathrm{con}(t)$ for the concept $D \in E$ to be minimised and some term t with $\mathfrak{n}_E \to^+ t$, we can discard all terms t' derived from t, since $\mathcal{O} \not\models D \equiv \mathrm{con}(t')$. We refer to this property as *subsumption monotonicity* and use it to filter out rules where the term on the right-hand side is too general. For instance, we can skip the rule $\mathfrak{n}_{E_5} \to \mathfrak{n}_{E_3}$ in Example 1, since $\mathcal{O} \not\models \mathrm{con}(\mathfrak{n}_{E_5}) \equiv \mathrm{con}(\mathfrak{n}_{E_3})$.

We formalise the above monotonicity properties of subsumer grammars as follows:

Lemma 2. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology and $E \in \mathcal{E}_\mathcal{O}$. Further, let t_1, t_2 be two terms such that $\mathfrak{n}_E \to^+_{G^{\sqsubseteq}(\mathcal{O},E)} t_1 \to^+_{G^{\sqsubseteq}(\mathcal{O},E)} t_2$. The following is true:*

1. $\mathrm{s}(t_1) \leq \mathrm{s}(t_2)$.
2. *If $\mathcal{O} \not\models \mathit{con}(\mathfrak{n}_E) \equiv \mathit{con}(t_1)$, then also $\mathcal{O} \not\models \mathit{con}(\mathfrak{n}_E) \equiv \mathit{con}(t_2)$.*

Looking back at subsumer grammars, we can further observe that a large proportion of rules of type R2 introduce *redundant conjuncts*—conjuncts that are not necessary for preserving the equivalence between the concept D to be minimised and the derived concept D'. When computing minimal equivalent concepts, such rules can be skipped as they never lead to minimal terms preserving equivalence to D. For instance, when applying rules to the term $\mathfrak{n}_{E_5}$ in Example 1, we can skip the rule $\mathfrak{n}_{E_5} \to \sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}, \mathfrak{n}_{E_8})$, since $\mathcal{O}'_{\mathrm{ex}} \models \mathrm{con}(\sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}, \mathfrak{n}_{E_8})) \equiv \mathrm{con}(\sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}))$. We formalise this observation within the following definition of *irreducible* sets of equivalence classes—sets that do not contain redundant elements:

Definition 2. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology and S a subset of $\mathcal{E}_\mathcal{O}$. The set S is* irreducible wrt. *$\mathcal{O}$ if and only if there is no subset S' of S such that $\mathcal{O} \models \mathit{con}(\bigsqcap(S)) \equiv \mathit{con}(\bigsqcap(S'))$.*

We now incorporate the above observations into a definition of dynamic derivation rules by imposing suitable restrictions onto the set of rules $R^{\sqsubseteq}$ given in Definition 1.

Definition 3. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology, $E_0 \in \mathcal{E}_\mathcal{O}$ and t a term such that $\mathfrak{n}_{E_0} \to^+_{G^{\sqsubseteq}(\mathcal{O},E_0)} t$. Let further $k \geq 0$ and let $\mathcal{C} \in T(\mathcal{F}^{\mathcal{EL}} \cup \mathcal{N}^{\mathcal{O}}, \{X\})$ be a context containing the variable X such that $\mathcal{C}[\mathfrak{n}_E/X] = t$ for some non-terminal $\mathfrak{n}_E$ occurring in t. The* dynamic set of derivation rules *$R^{E,k,\mathcal{C},\mathcal{O}}$ for E with size limit k preserving equivalence within $\mathcal{C}$ wrt. $\mathcal{O}$ is then given by $\emptyset$ in case $k = 0$ and, otherwise, as follows:*

(DR1) $\mathfrak{n}_E \to A$ *for each* $A \in E \cap (\mathit{sig}_C(\mathcal{O}) \cup \{\top\})$*;*

(DR2) $\mathfrak{n}_E \to \sqcap(S)$ *for each* $S \subseteq \mathcal{S}_E^{\not\equiv}$ *such that* $S \neq \emptyset$, $|S| \leq k$, $\mathcal{O} \models \mathit{con}(t) \equiv \mathit{con}(\mathcal{C}[\bigsqcap(S)/X])$ *and, unless* $S = \{E_\top\}$*, S is irreducible wrt. $\mathcal{O}$;*

(DR3) $\mathfrak{n}_E \to \exists r(\mathfrak{n}_{E_1})$ *for each* $\langle r, \mathfrak{n}_{E_1} \rangle \in \mathit{Ex}_{\mathfrak{n}_E}$ *in case* $k \geq 2$.

Algorithm 1 shows the computation of ground terms representing minimal equivalent concepts based on dynamic derivation rules. Given an $\mathcal{EL}$ ontology $\mathcal{O}$ and some concept $D \in E_0$ for some $E_0 \in \mathcal{E}_\mathcal{O}$, we call the recursive function MINIMISE with $\mathcal{O}$ as the ontology, the term representation t_D of D as the smallest known ground term $t_{\min}$, and $\mathfrak{n}_{E_0}$ as the starting point for further derivations t. In every call of MINIMISE, we first test whether the current term

[2] Within this context, we are referring to syntactic variations due to the associativity and commutativity of $\sqcap$ as well as the possibility of multiple occurrences of the same conjunct within conjunctions.

Algorithm 1: MINIMISE function computing a term representing a minimal equivalent concept.

Input: Ontology $\mathcal{O}$, smallest known ground term t_{min}, starting point for further derivations t

1 **if** t *is ground* **then**
2 return t;
3 $\langle \mathfrak{n}_E, \mathcal{C} \rangle \leftarrow$ pick a non-terminal occurring in t and generate the corresponding context;
4 **for** *rule* $\in R^{E,\mathrm{s}(t_{min})-\mathrm{s}(\mathcal{C}),\mathcal{C},\mathcal{O}}$ **do**
5 $t' \leftarrow$ apply rule to $\mathcal{C}$;
6 $t_{\text{min}} \leftarrow$ MINIMISE$(\mathcal{O}, t_{\text{min}}, t')$;
7 return t_{min};

t is ground, in which case we return t as the new smallest known ground term. If t is not ground, we randomly pick a non-terminal $\mathfrak{n}_E$ occurring in t and obtain the corresponding context $\mathcal{C}$ by replacing $\mathfrak{n}_E$ with a variable X. We then compute the dynamic set of derivation rules R for E. The size limit $k = \mathrm{s}(t_{\text{min}}) - \mathrm{s}(\mathcal{C})$ ensures that terms produced by rules from R are always smaller than t_{min}. Since the values E and $\mathcal{C}$ are used as arguments when computing R, all rules from R preserve equivalence to $\mathrm{con}(t)$. Since we start the computation with $t = t_D$, all generated terms have concept representations equivalent to D wrt. $\mathcal{O}$. In lines 4-6, we apply each derivation rule from R and call the function MINIMISE with a potentially updated value of t_{min} and the new term t' as the starting point for further derivations.

We obtain the following result for computing terms representing minimal equivalent concepts using the function MINIMISE:

Theorem 1. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology and D an $\mathcal{EL}$ concept represented by a term t_D. Further, let $\mathcal{O}' = \mathcal{O} \cup \{D \sqsubseteq \top\}$ and $E \in \mathcal{E}_{\mathcal{O}'}$ such that $D \in E$.* MINIMISE$(\mathcal{O}', t_D, \mathfrak{n}_E)$ *computes a minimal ground term t such that $\mathcal{O} \models D \equiv$ con(t).*

Computing Dynamic Derivation Rules

While computing rules of types DR1 and DR3 is straightforward and the corresponding computational effort is negligible, the challenging task is to efficiently compute rules of type DR2. Algorithm 2 shows the computation of the latter type of rules. Within the algorithm, we first compute for each rule the corresponding non-empty set of equivalence classes that form the right-hand side of that rule. The algorithm iteratively builds subsets of $\mathcal{S}_E^{\sqcap}$, as required by Definition 3, starting with the smallest and extending the size of considered sets by one in each iteration. For each set, we enforce the size requirement in lines 8 and 12, the equivalence requirement in line 12, and the irreducibility requirement in line 26. The algorithm contains several optimisations based on observations from our experiments. These optimisations aim to reduce the number of considered sets as early within the computation process as possible:

1. In many cases, no equivalence-preserving subset of $\mathcal{S}_E^{\sqcap}$ exists. In order to avoid unnecessary computation, we test in lines 3-4 of Algorithm 2 whether the entire set $\mathcal{S}_E^{\sqcap}$ of conjuncts is sufficient to preserve equivalence within $\mathcal{C}$ wrt. $\mathcal{O}$ and, otherwise, return an empty set.
2. There is often a *required* set of conjuncts—a set of conjuncts that is shared among all conjunctions preserving equivalence. By computing it in line 5 and using it as the starting point for the iterative part of the algorithm, we avoid subsets of $\mathcal{S}_E^{\sqcap}$ that clearly do not preserve equivalence.
3. A large number of elements from $\mathcal{S}_E^{\sqcap}$ are subsumer successors of other elements within $\mathcal{S}_E^{\sqcap}$ or conjunctions

Algorithm 2: COMPUTERULES function computing rules of type DR2 for $R^{E,k,\mathcal{C},\mathcal{O}}$

Input: Equivalence class E, size limit $k \geq 1$, context $\mathcal{C} \in T(\mathcal{F}^{\mathcal{EL}} \cup \mathcal{N}^{\mathcal{O}}, \{X\})$ containing the variable X, ontology $\mathcal{O}$

1 $D \leftarrow \mathrm{con}(\mathcal{C}[\mathfrak{n}_E/X])$;
2 $M^{\text{res}} \leftarrow \emptyset$;
3 **if** $\mathcal{O} \not\models D \equiv$ *con*$(\mathcal{C}[\bigsqcap(\mathcal{S}_E^{\sqcap})/X])$ **then**
4 return $\emptyset$;
5 $S^{\text{req}} \leftarrow$ compute the required subset of $\mathcal{S}_E^{\sqcap}$;
6 $M^{\text{test}} \leftarrow \{S^{\text{req}}\}$;
7 $s \leftarrow |S^{\text{req}}|$;
8 **while** $s \leq k$ *and* $M^{test} \neq \emptyset$ **do**
9 $s \leftarrow s + 1$;
10 $M^{\text{expand}} \leftarrow \emptyset$;
11 **for** $S \in M^{test}$ **do**
12 **if** $S \neq \emptyset$ *and* $\mathcal{O} \models D \equiv$ *con*$(\mathcal{C}[\bigsqcap(S)/X])$ **then**
13 $M^{\text{res}} \leftarrow M^{\text{res}} \cup \{S\}$;
14 **else**
15 reducible $\leftarrow$ false;
16 **for** S' *with* $\langle S', S \rangle \in$ SUCC **do**
17 **if** $\mathcal{O} \models$ *con*$(\bigsqcap(S)) \equiv$ *con*$(\bigsqcap(S'))$ **then**
18 reducible $\leftarrow$ true;
19 **for** S'' *with* $\langle S', S'' \rangle \in$ SUCC **do**
20 EXCL $\leftarrow$ EXCL $\cup \{\langle S'', E' \rangle \mid E' \in S \setminus S'\}$;
21 **if** *reducible* = *false* **then**
22 $M^{\text{expand}} \leftarrow M^{\text{expand}} \cup \{S\}$;
23 $M^{\text{test}} \leftarrow \emptyset$;
24 **for** $S \in M^{expand}$ **do**
25 **for** $E' \in \mathcal{S}_E^{\sqcap}$ **do**
26 **if** $\langle S, E' \rangle \notin$ EXCL *and* $E' \notin S$ *and there is no* $S' \in M^{res}$ *with* $S' \subseteq S \cup \{E'\}$ **then**
27 $M^{\text{test}} \leftarrow M^{\text{test}} \cup \{S \cup \{E'\}\}$;
28 SUCC $\leftarrow$ SUCC $\cup \{\langle S, S \cup \{E'\} \rangle\}$;
29 EXCL $\leftarrow$ EXCL $\cup \{\langle S \cup \{E'\}, E'' \rangle \mid E'' \in \mathcal{S}_{E'}^{\sqcap}$ or $\langle S, E'' \rangle \in$ EXCL$\}$;
30 return $\{\mathfrak{n}_E \rightarrow \mathcal{C}[\bigsqcap(S)/X] \mid S \in M^{\text{res}}\}$;

Ontology	#Ax.	$\mathrm{s}(C)$	$\mathrm{s} \geq 2$			$\mathrm{s} \geq 5$			$\mathrm{s} \geq 10$		
			Time(s)	Red.	Red.by	Time(s)	Red.	Red.by	Time(s)	Red.	Red.by
Snomed CT	320,335	5.4	268.0	36%	54%	333.3	50%	28%	704.1	33%	32%
Galen	51,320	3.1	1.8	27%	43%	7.0	86%	47%	15.2	100%	72%
Genomic CDS	4,322	14.2	0.2	14%	76%	0.4	32%	90%	0.5	37%	90%
FYPO	12,265	2.9	1.4	16%	48%	5.4	76%	48%	1.0	0%	0%
NCIT	204,976	3.0	2.6	5%	26%	5.5	33%	26%	4.1	40%	14%

Table 2: Evaluation results.

thereof. Thus, adding those to the corresponding sets makes those sets reducible. In order to avoid creating sets that cannot be extended into irreducible ones, we record the corresponding relationships between subsets of $\mathcal{S}_E^{\not\sqsupseteq}$ and equivalence classes by means of the relation $\texttt{EXCL} \subseteq 2^{\mathcal{S}_E^{\not\sqsupseteq}} \times \mathcal{E}_{\mathcal{O}}$. This relation is gradually constructed in lines 20 and 29. In line 26, we use known EXCL relationships to avoid creating subsets of $\mathcal{S}_E^{\not\sqsupseteq}$ that will lead to reducible sets only.

We obtain the following result for Algorithm 2:

Theorem 2. *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology, $E_0 \in \mathcal{E}_{\mathcal{O}}$ and t a term such that $\mathfrak{n}_{E_0} \rightarrow^{+}_{G^{\sqsubseteq}(\mathcal{O},E_0)} t$. Further, let $k \geq 1$ and let $\mathcal{C} \in T(\mathcal{F}^{\mathcal{EL}} \cup \mathcal{N}^{\mathcal{O}}, \{X\})$ be a context containing the variable X such that $\mathcal{C}[\mathfrak{n}_E/X] = t$ for some non-terminal $\mathfrak{n}_E$ occurring in t. The function* ComputeRules$(E, k, \mathcal{C}, \mathcal{O})$ *computes rules of type DR2 for $R^{E,k,\mathcal{C},\mathcal{O}}$ in time exponential in the size of $\mathcal{S}_E^{\not\sqsupseteq}$ in the worst case.*

Evaluation

We evaluate our method on concepts from Snomed Clinical Terms (Snomed CT) (Stearns et al. 2001), National Cancer Institute Thesaurus (NCIT) (Sioutos et al. 2007), Galen (Rector et al. 1994), Fission Yeast Phenotype Ontology (FYPO) (Harris et al. 2013) and Genomic Clinical Decision Support Ontology (Genomic CDS) (Samwald 2013). For each ontology $\mathcal{O}$, we selected 100 $\mathcal{EL}$ concepts occurring in the axioms of $\mathcal{O}$ with the size at least 2. The order of the axioms was determined by the iterator over the set of axioms retrieved by the OWL API (Horridge and Bechhofer 2011). We then minimised each concept D with respect to the ontology $\mathcal{O}\backslash\{\alpha\}$, where α is the axiom in which the concept D occurred. We set a timeout of 5 minutes for all ontologies except Snomed CT, for which we increase the timeout to 30 minutes due to longer reasoner response times.[3]

In Table 2, we first list for each ontology the total number of axioms (*#Ax.*) and the average size of the 100 evaluated concepts ($\mathrm{s}(C)$). We then separately show evaluation results for concepts of size at least 2, at least 5 and at least 10. For each size category, we include the average processing time in seconds (*Time*), the percentage of concepts for which a smaller equivalent concept existed (*Red.*), and the average achieved size difference for those concepts (*Red.by*).

We can see that, while the number of axioms and the average size of evaluated concepts differ significantly, we could find a smaller equivalent concept for a notable proportion of concepts from all ontologies. As expected, this effect is more prominent in the size categories $\mathrm{s} \geq 5$ and $\mathrm{s} \geq 10$. An exception is FYPO, which had only 1 concept of size at least 10, for which no smaller representation existed. In many cases, a notable reduction in size has been achieved.

We measured how the number of applied rules changes when the computation is based on a dynamic rather than static set of derivation rules. We found that, on average, the number of applied rules per non-terminal decreased by 8 orders of magnitude. We also found that, on average, the optimisations included within the function ComputeRules reduced the number of considered subsets of $\mathcal{S}_E^{\not\sqsupseteq}$ by 5 orders of magnitude.

In terms of computation time, we observe that, while a timeout occurred for 1 concept in NCIT and 4 concepts in Snomed CT, on average, concept minimisation takes just a few seconds for all ontologies except Snomed CT. We conclude that, for ontologies of an average size and complexity, concept minimisation could be made available as a feature within interactive ontology editors such as Protégé (Musen 2013).

Discussion and Outlook

This work can be extended in various directions. For instance, one open question is whether concepts expressed in more expressive DLs can be minimised efficiently as well. This work addresses the problem to a certain extent—the presented method can be used to compute small equivalent concepts for EL concepts within ontologies expressed in more expressive DLs without a guarantee of minimality. However, different optimisations might be more effective and different methods are required to achive an optimal result. We plan to investigate this in future work.

Another open question is how and to what extent the presented results can be generalised to support other refactoring tasks. The presented algorithm can be made more flexible, e.g. in order to compute all equivalent concepts up to a certain size and, thereby, enable the user to choose the most appropriate meaning-preserving concept. The presented results are also directly relevant for minimising $\mathcal{EL}$ ontologies as a whole. A systematic analysis of benefits for supporting various refactoring tasks has been left for future work.

[3]The ELK reasoner (Kazakov, Krötzsch, and Simančík 2014) used in our evaluation took 6 times longer to update classification results for Snomed CT in comparison to the average time it took for other ontologies. Therefore, we use a sixfold timeout in case of Snomed CT.

References

Baader, F.; Borgwardt, S.; and Morawska, B. 2012. Extending unification in EL towards general TBoxes. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning (KR 2012)*, 568–572.

Baader, F.; Küsters, R.; and Molitor, R. 2000. Rewriting concepts using terminologies. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR 2000)*, 297–308.

Carral, D.; Feier, C.; Grau, B. C.; Hitzler, P.; and Horrocks, I. 2014. $\mathcal{EL}$-ifying ontologies. In *Proceedings of the 10th International Joint Conference on Automated Reasoning (IJCAR 2014)*, 464–479.

Comon, H.; Jacquemard, F.; Dauchet, M.; Gilleron, R.; Lugiez, D.; Loding, C.; Tison, S.; and Tommasi, M. 2008. Tree automata techniques and applications.

Du, J.; Wang, K.; and Shen, Y. 2015. Towards tractable and practical ABox abduction over inconsistent description logic ontologies. In *Proceedings of the 29th National Conference on Artificial Intelligence (AAAI 2015)*.

Grau, B. C.; Kharlamov, E.; and Zheleznyakov, D. 2012. How to contract ontologies. In *Proceedings of OWL: Experiences and Directions (OWLED 2012)*.

Grimm, S., and Wissmann, J. 2011. Elimination of redundancy in ontologies. In *Proceedings of the 8th Extended Semantic Web Conference (ESWC 2011)*, 260–274.

Harris, M. A.; Lock, A.; Bähler, J.; Oliver, S. G.; and Wood, V. 2013. FYPO: the fission yeast phenotype ontology. *Bioinformatics* 29(13):1671–1678.

Horridge, M., and Bechhofer, S. 2011. The OWL API: A Java API for OWL ontologies. *Semantic Web* 2(1):11–21.

Kazakov, Y.; Krötzsch, M.; and Simančík, F. 2014. The incredible ELK. *Journal of Automated Reasoning* 53(1):1–61.

Klarman, S.; Endriss, U.; and Schlobach, S. 2011. ABox abduction in the description logic $\mathcal{ALC}$. *Journal of Automated Reasoning* 46(1):43–80.

Konev, B.; Ozaki, A.; and Wolter, F. 2016. A model for learning description logic ontologies based on exact learning. In *Proceedings of the 30th National Conference on Artificial Intelligence (AAAI 2016)*, 1008–1015.

Lehmann, J., and Hitzler, P. 2010. Concept learning in description logics using refinement operators. *Machine Learning* 78(1-2):203–250.

Lutz, C.; Piro, R.; and Wolter, F. 2011. Description logic TBoxes: Model-theoretic characterizations and rewritability. In *Proceedings of the 22th International Joint Conference on Artificial Intelligence (IJCAI 2011)*, 983–988.

Lutz, C.; Seylan, I.; and Wolter, F. 2012. An automata-theoretic approach to uniform interpolation and apprximation in the description logic el. In *Proceedings of the 13th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2012)*.

Musen, M. A. 2013. Protégé ontology editor. *Encyclopedia of Systems Biology* 1763–1765.

Nikitina, N., and Rudolph, S. 2012. ExpExpExplosion: Uniform interpolation in general EL terminologies. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, 618–623.

Nikitina, N., and Rudolph, S. 2014. (Non-)Succinctness of Uniform Interpolants of General Terminologies in the Description Logic EL. *Artificial Intelligence* 215(0):120–140.

Nikitina, N., and Schewe, S. 2013a. More is Sometimes Less: Succinctness in $\mathcal{EL}$. In *Proceedings of the 26th International Workshop on Description Logics (DL 2013)*, 403–414.

Nikitina, N., and Schewe, S. 2013b. Simplifying Description Logic Ontologies. In *Proceedings of the 12th International Semantic Web Conference (ISWC 2013)*, 411–426.

Qi, G.; Liu, W.; and Bell, D. A. 2006. Knowledge base revision in description logics. In *Proceedings of the 10th European Conference on Logics in Artificial Intelligence (JELIA 2006)*, 386–398.

Rector, A.; Gangemi, A.; Galeazzi, E.; Glowinski, A.; and Rossi-Mori, A. 1994. The GALEN CORE model schemata for anatomy: Towards a re-usable application-independent model of medical concepts. In *Proceedings of the 12th International Congress of the European Federation for Medical Informatics (MIE 1994)*, 229–233.

Samwald, M. 2013. Genomic CDS: an example of a complex ontology for pharmacogenetics and clinical decision support. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE 2013)*, 128–133.

Sioutos, N.; Coronado, S. d.; Haber, M. W.; Hartel, F. W.; Shaiu, W.-L.; and Wright, L. W. 2007. NCI Thesaurus: A semantic model integrating cancer-related clinical and molecular information. *Journal of Biomedical Informatics* 40(1):pp.30–43.

Stearns, M. Q.; Price, C.; Spackman, K. A.; and Wang, A. Y. 2001. SNOMED clinical terms: overview of the development process and project status. In *Proceedings of the American Medical Informatics Association Annual Symposium (AMIA 2001)*, 662–666.

Turhan, A., and Zarrieß, B. 2013. Computing the LCS w.r.t. general $\mathcal{EL}+$-TBoxes. In *Proceedings of the 26th International Workshop on Description Logics (DL 2013)*, 477–488.