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

283

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 *et al.* 2012), *uniform interpolation* (Nikitina and Rudolph 2012; Lutz *et al.* 2012), *ontology learning* (Konev *et al.* 2016; Lehmann and Hitzler 2010), *rewriting ontologies into less expressive logics* (Carral *et al.* 2014; Lutz *et al.* 2011), *abduction* (Du *et al.* 2015; Klarman *et al.* 2011), and *knowledge revision* (Grau *et al.* 2012; Qi *et al.* 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 *et al.* 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).

Further details and proofs can be found in the extended version [1] of this paper.

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 $D \sqsubseteq C$.[2] 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 $\mathrm{sub}(\mathcal{O})$ to denote the set of all subconcepts in $\mathcal{O}$ including $\top$.

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* is a pair ($\mathcal{F}$,*Arity*) where $\mathcal{F}$ is a finite set and *Arity* is a mapping from $\mathcal{F}$ into $\mathbb{N}$. We use superscripts to denote the arity of alphabet symbols (if it is not 0), e.g., $f^2(g^1(a), a)$. 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* with a fixed arity 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:

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

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

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

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

[1] https://dl.dropbox.com/u/10637748/AAAI2017a.pdf

[2] 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.

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 $\mathtt{s}(C_{\text{ex}}) - 1$ from the ontology signature $\text{sig}(\mathcal{O}_{\text{ex}})$ 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. For instance, in case of Snomed, generating concepts of size up to 10 increases the number of concepts within the ontology by 60 orders of magnitude. It would be impossible to classify such an ontology within a reasonable time using state-of-the-art reasoning software running on modern hardware.

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 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* $\mathcal{E}_{\mathcal{O}} = \{E \subseteq \text{sub}(\mathcal{O}) \cup \{\top\} \mid \forall C_1, C_2 \in E : \mathcal{O} \models C_1 \equiv C_2\}$ and assign a single non-terminal $\mathfrak{n}_E$ and a *subsumer grammar* $G^{\sqsubseteq}(\mathcal{O}, E)$ to each equivalence class 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 $\mathcal{F}^{\mathcal{EL}} = \text{sig}_C(\mathcal{O}) \cup \{\top\} \cup \{\exists r^1 \mid r \in \text{sig}_R(\mathcal{O})\} \cup \{\sqcap^i \mid 2 \leq i\}$, where $\top$ and concept symbols in $\text{sig}_C(\mathcal{O})$ are constants, $\exists r^1$ for $r \in \text{sig}_R(\mathcal{O})$ 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 $\mathcal{S}_E^{\not\sqsubseteq}$ 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 guarantee completeness but make our computations more expensive, we exclude those equivalence classes from $\mathcal{S}_E^{\not\sqsubseteq}$. The set $\mathcal{S}_E^{\not\sqsubseteq}$ 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 $\mathcal{S}_E^{\not\sqsubseteq}$ 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 $\{\langle r, E'\rangle \mid r \in \text{sig}_R(\mathcal{O}), E' \in \mathcal{E}_{\mathcal{O}}$ 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 $\text{Ex}_{E_6} = \{\langle r, E_3\rangle\}$, $\text{Ex}_{E_8} = \{\langle r, E_7\rangle\}$ and $\text{Ex}_{E_9} = \{\langle r, E_4\rangle\}$.

E	elements of E	$\mathcal{S}_E^{\not\sqsubseteq}$
$E_\top$	$\{\top\}$	$\emptyset$
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 $\mathcal{S}_E^{\not\sqsubseteq}$ 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 $\mathcal{S}_E^{\not\sqsubseteq}$ 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* $A \in E \cap (\text{sig}_C(\mathcal{O}) \cup \{\top\})$;

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

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

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 $2^{\mathcal{S}_E^{\not\sqsubseteq}} \setminus \{\emptyset\}$, 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 $\mathcal{S}_{E_i}^{\not\sqsubseteq}$ 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 and D a subconcept of $\mathcal{O}$ with $D \in E$ for some $E \in \mathcal{E}_\mathcal{O}$.*

1. *Let D' be a subsumer of D wrt. $\mathcal{O}$. Then, there exists a syntactic variant[3] 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 $E \in \mathcal{E}_\mathcal{O}$ with some $D \in E$. 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 $\mathcal{O}'_{\text{ex}} \models \mathrm{con}(\sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}, \mathfrak{n}_{E_8})) \equiv \mathrm{con}(\sqcap(\mathfrak{n}_{E_1}, \mathfrak{n}_{E_2}))$. 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 $k \geq 1$ 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* 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:*

(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 $S \subseteq \mathcal{S}^{\not\equiv}_E$ 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 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

[3] 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};

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 $\mathcal{S}_E^{\curlywedge}$, 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 $\mathcal{S}_E^{\curlywedge}$ exists. In order to avoid unnecessary computation, we test in lines 3-4 of Algorithm 2 whether the entire set $\mathcal{S}_E^{\curlywedge}$ of conjuncts is sufficient to preserve equivalence within $\mathcal{C}$ wrt. $\mathcal{O}$ and, otherwise, return an empty set.

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** $\mathcal{O} \not\models D \equiv con(\mathcal{C}[\bigsqcap(\mathcal{S}_E^{\curlywedge})/X])$ **then**
4 return $\emptyset$;
5 $S^{\text{req}} \leftarrow$ compute the required subset of $\mathcal{S}_E^{\curlywedge}$;
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** $E' \in \mathcal{S}_E^{\curlywedge}$ **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 $\cup \{\langle S \cup \{E'\}, E'' \rangle \mid E'' \in \mathcal{S}_{E'}^{\curlywedge}$ or $\langle S, E'' \rangle \in$ EXCL$\}$;
30 return $\{\mathfrak{n}_E \rightarrow \mathcal{C}[\bigsqcap(S)/X] \mid S \in M^{\text{res}}\}$;

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 $\mathcal{S}_E^{\curlywedge}$ that clearly do not preserve equivalence.
3. A large number of elements from $\mathcal{S}_E^{\curlywedge}$ are subsumer successors of other elements within $\mathcal{S}_E^{\curlywedge}$ or conjunctions 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

Ontology	#Ax.	s(C)	s ≥ 2			s ≥ 5			s ≥ 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.

of $\mathcal{S}_E^{\not\sqsupset}$ and equivalence classes by means of the relation $\texttt{EXCL} \subseteq 2^{\mathcal{S}_E^{\not\sqsupset}} \times \mathcal{E}_\mathcal{O}$. This relation is gradually constructed in lines 20 and 29. In line 26, we use known EXCL relationships to avoid creating subsets of $\mathcal{S}_E^{\not\sqsupset}$ 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 $\mathfrak{n}_{E_0} \rightarrow^+_{G^\sqsubseteq(\mathcal{O},E_0)} t$. Let further $k \geq 1$ 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 $\mathcal{S}_E^{\not\sqsupset}$ 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.[4]

In Table 2, we first list for each ontology the total number of axioms (*#Ax.*) and the average size of the 100 evaluated concepts (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 s ≥ 5 and s ≥ 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 $\mathcal{S}_E^{\not\sqsupset}$ 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).

Summary

In this paper, we discussed the computation of minimal equivalent concepts wrt. $\mathcal{EL}$ ontologies. We investigated the feasibility of concept minimisation in practice by exploring various ways of restricting the search space. We first discussed a grammar-based approach in which candidate concepts are restricted to subsumers of the concept in question. We then showed that we can further narrow down the search space by making the set of derivation rules dynamic—computing them on-demand for a particular context and a particular non-terminal while incorporating information gathered throughout the minimisation process.

We evaluated our method on 5 bio-medical ontologies and found that a smaller equivalent concept existed for a notable proportion of concepts from those ontologies. In many cases, a notable reduction in size has been achieved. For instance, for concepts of size at least 5, the achieved size reduction ranged between 26% and 90%. The evaluation also confirms the feasibility of concept minimisation in practice. For Snomed CT, most concepts could be processed within 30 minutes and, on average, within 5 minutes. For the other ontologies, the computation took on average just a few seconds. We conclude that our tool would be a valuable new feature for ontology editors.

[4]The ELK reasoner (Kazakov *et al.* 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. We therefore increase the timeout by 6.

References

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

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

David Carral, Cristina Feier, Bernardo Cuenca Grau, Pascal Hitzler, and Ian Horrocks. $\mathcal{EL}$-ifying ontologies. In *Proceedings of the 10th International Joint Conference on Automated Reasoning (IJCAR 2014)*, pages 464–479, 2014.

Hubert Comon, Florent Jacquemard, Max Dauchet, Remi Gilleron, Denis Lugiez, Christof Loding, Sophie Tison, and Marc Tommasi. Tree automata techniques and applications, 2008.

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

Bernardo Cuenca Grau, Evgeny Kharlamov, and Dmitriy Zheleznyakov. How to contract ontologies. In *Proceedings of OWL: Experiences and Directions (OWLED 2012)*, 2012.

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

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.

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

Yevgeny Kazakov, Markus Krötzsch, and František Simančík. The incredible ELK. *Journal of Automated Reasoning*, 53(1):1–61, 2014.

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

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

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

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

Carsten Lutz, Inanc Seylan, and Frank Wolter. 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)*, 2012.

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

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

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

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

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

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

A.L. Rector, A. Gangemi, E. Galeazzi, A.J. Glowinski, and A. Rossi-Mori. 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)*, pages 229–233, 1994.

Matthias Samwald. 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)*, pages 128–133, 2013.

Nicholas Sioutos, Sherri de Coronado, Margaret W. Haber, Frank W. Hartel, Wen-Ling Shaiu, and Lawrence W. Wright. NCI Thesaurus: A semantic model integrating cancer-related clinical and molecular information. *Journal of Biomedical Informatics*, 40(1):pp.30–43, February 2007.

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

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

Proofs

Proof for Lemma 1

Lemma 1 *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology and D a subconcept of $\mathcal{O}$ with $D \in E$ for some $E \in \mathcal{E}_\mathcal{O}$.*

1. *Let D' be a subsumer of D wrt. $\mathcal{O}$. Then, there exists a syntactic variant 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. $\mathcal{O}$.*

Proof. We investigate the above two claims separately:

1. The completeness is proved by induction on the role depth of D' given by

$$D' = \prod_{1 \leq j \leq n} A_j \sqcap \prod_{1 \leq k \leq m} \exists r_k.C_k$$

where A_j are concept symbols, r_k are role symbols and C_k are arbitrary $\mathcal{EL}$ concepts. W.l.o.g., we can assume that all A_j are pairwise different. For each A_j, we know that $\mathcal{O} \models D \sqsubseteq A_j$ and $A_j \in E_j$ for some $E_j \in \mathcal{E}_\mathcal{O}$. Thus, each E_j is in $\mathcal{S}_E^{\not\exists}$. Further, for each $\exists r_k.C_k$, we have $E'_k, E''_k \in \mathcal{E}_\mathcal{O}$ with $\exists r_k.C_k \in E'_k$, $C_k \in E''_k$. Since $\mathcal{O} \models D \sqsubseteq \exists r_k.C_k$, we have $E'_k \in \mathcal{S}_E^{\not\exists}$.

   - Assume a role depth of 0 so that $D' = \prod_{1 \leq j \leq n} A_j$. By (R2) of Definition 1, $\mathfrak{n}_E \to \prod(S) \in R^{\sqsubseteq}$ for $S = \{E_1, \cdots, E_n\}$. Further, by (R1) of Definition 1, $\mathfrak{n}_{E_j} \to A_j \in R^{\sqsubseteq}$ for each A_j. By applying (R1) to each E_j within the term $\prod(S)$, we obtain a term $t_{D'}$. Thus, there exists a concept D'' such that $D' = D''$ and $t_{D''} \in L(G^{\sqsubseteq}(\mathcal{O}, E))$. We have shown that the theorem holds for role depth 0.
   - Assume that the role depth is greater than 0 so that D' has the general form above. By (R2) of Definition 1, $\mathfrak{n}_E \to \prod(S) \in R^{\sqsubseteq}$ for any subset of S of $\mathcal{S}_E^{\not\exists}$. This includes the set $S_{D'} = \{E_1, \cdots, E_n, E'_1, \cdots, E'_m\}$. By (R3) of Definition 1, $\mathfrak{n}_{E_k} \to \exists r_k(\mathfrak{n}_{E'_k}) \in R^{\sqsubseteq}$. Thus, we can derive the term $\sqcap(\mathfrak{n}_{E_1}, ..., \mathfrak{n}_{E_n}, \exists r_1(\mathfrak{n}_{E'_1}), ..., \exists r_m(\mathfrak{n}_{E'_m}))$ from $\mathfrak{n}_{E_k}$. By induction hypothesis, $t_{C'_k} \in L(G^{\sqsubseteq}(\mathcal{O}, E'_k))$, where $t_{C'_k}$ represents some syntactic variant C'_k of C_k. Further, by applying (R1) to all E_j, we obtain the term $t_{D'}$. Thus, there exists a concept D'' such that $D' = D''$ and $t_{D''} \in L(G^{\sqsubseteq}(\mathcal{O}, E))$.
2. The proof of soundness of $G^{\sqsubseteq}(\mathcal{O}, E)$ can be done by induction on the maximal nesting depth of functions in $t_{D'}$:
   - Assume that the nesting depth of functions in $t_{D'}$ is 0, i.e., $t_{D'} = B$ is a concept symbol or $\top$. The term B can only be derived from $\mathfrak{n}_E$ by n transitions (R2) with a single non-terminal on the right-hand side, and, once $\mathfrak{n}_{E'}$ is derived with $B \in E'$, the rule (R1). Let $E_1, ..., E_n$ be such that $\mathfrak{n}_E \to \mathfrak{n}_{E_1} \to ... \to \mathfrak{n}_{E_n} \to \mathfrak{n}_{E'}$. Then, by Definition 1, for each pair E_i, E_{i+1} it holds that E_{i+1} is a subsumer successor of E_i. The same holds for pairs E, E_1 and E_n, E'. It follows that E' is a subsumer successor of E and thus $\mathcal{O} \models D \sqsubseteq B$.
   - Assume that $t_{D'} = \exists r(t')$ for some term t'. Then, the derivation of t from $\mathfrak{n}_E$ starts with n transitions (R2) with a single non-terminal on the right-hand side, such that $\mathfrak{n}_{E'}$ for some $E' \in \mathcal{E}_\mathcal{O}$ is derived. As argued in the previous part of the proof, E' is a subsumer successor of E. Further, with a subsequent application of a transition (R3), we derive a term $\exists r(\mathfrak{n}_{E''})$ for some $E'' \in \mathcal{E}_\mathcal{O}$ from $\mathfrak{n}_{E'}$. By Definition 1 (R3), there exists a concept C with $C \in E''$ such that $\exists r.C \in E'$. Therefore, $\mathcal{O} \models D \sqsubseteq \exists r.C$. Let $C' = \text{con}(t')$. Since $E'' \in \mathcal{E}_\mathcal{O}$ and $\mathfrak{n}_{E''} \to t'$, by induction hypothesis, $\mathcal{O} \models C \sqsubseteq C'$. Therefore, $\mathcal{O} \models D \sqsubseteq \exists r.C'$ with $\text{con}(t_{D'}) = \exists r.C'$.
   - Assume that $t_{D'} = \sqcap(t_1, ..., t_n)$ for a set of terms $t_1, ..., t_n$. Then, the derivation of t from $\mathfrak{n}_E$ starts with n transitions (R2) with a single non-terminal on the right-hand side, such that $\mathfrak{n}_{E'}$ for some $E' \in \mathcal{E}_\mathcal{O}$ is derived. As argued in the previous part of the proof, E' is a subsumer successor of E. Further, with a subsequent application of an (R2) rule with a conjunction on the right-hand side, we derive $\sqcap(\mathfrak{n}_{E_1}, ..., \mathfrak{n}_{E_n})$, where each t_i is in $L(G^{\sqsubseteq}(\mathcal{O}, \mathfrak{n}_{E_i}))$. By Definition 1, each E_i is a subsumer successor of E'. It follows that each E_i is a subsumer successor of E. Let $C_i \in E_i$ and $C'_i = \text{con}(t_i)$. Since each t_i is in $L(G^{\sqsubseteq}(\mathcal{O}, \mathfrak{n}_{E_i}))$, by induction hypothesis, $\mathcal{O} \models C_i \sqsubseteq C'_i$. Therefore, also $\mathcal{O} \models D \sqsubseteq C'_1 \sqcap ... \sqcap C'_n$ with $C_1 \sqcap ... \sqcap C_n = \text{con}(t_{D'})$.

□

Proofs for Lemma 2 and Theorem 1

Lemma 2 *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology and $E \in \mathcal{E}_\mathcal{O}$ with some $D \in E$. 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. *$\text{s}(t_1) \leq \text{s}(t_2)$.*
2. *If $\mathcal{O} \not\models$ con$(\mathfrak{n}_E) \equiv$ con(t_1), then also $\mathcal{O} \not\models$ con$(\mathfrak{n}_E) \equiv$ con(t_2).*

Proof Sketch. The first claim can be shown by induction on the length of the derivation of t_2 from t_1, in which we examine the possibly applied rules and observe that $\text{s}(t_1)$ is not greater than $\text{s}(t'_1)$ for each term t'_1 with $t_1 \to^+ t'_1$. For the second claim, we assume that $\mathcal{O} \models \text{con}(\mathfrak{n}_E) \equiv \text{con}(t_2)$. Since $\mathcal{O} \models \text{con}(\mathfrak{n}_E) \sqsubseteq \text{con}(t_1)$ and $\mathcal{O} \models \text{con}(t_1) \sqsubseteq \text{con}(t_2)$ and $\mathcal{O} \models \text{con}(t_2) \sqsubseteq \text{con}(\mathfrak{n}_E)$, we obtain $\mathcal{O} \models \text{con}(\mathfrak{n}_E) \equiv \text{con}(t_1)$. □

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).*

Proof. We need to show the three following properties of the computed term t: (a) t is ground; (b) $\mathcal{O} \models D \equiv \text{con}(t)$, (c) t is minimal.

(a). t is ground: Assuming that MINIMISE is called according to its specification, i.e., with t_{min} being a ground term, it can only return ground terms. This can be shown by induction on the nesting depth of function calls with two base cases: (1) the starting point for further derivations

Algorithm 3: COMPUTEALL function computing a superset of all minimal ground terms equivalent to t'' and smaller than t_{limit}.

Input: Ontology $\mathcal{O}$, size limit term t_{limit}, starting point for further derivations t'' with $\mathrm{s}(t'') < \mathrm{s}(_{\text{limit}})$

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 $M^{\text{res}} \leftarrow \emptyset$;
5 **for** *rule* $\in R^{E,\mathrm{s}(t_{limit})-\mathrm{s}(\mathcal{C}),\mathcal{C},\mathcal{O}}$ **do**
6 $t' \leftarrow$ apply rule to $\mathcal{C}$;
7 $M^{\text{res}} \leftarrow M^{\text{res}} \cup \text{COMPUTEALL}(\mathcal{O}, t_{\text{limit}}, t')$;
8 return M^{res};

t'' is ground, and (2) $R^{E,\mathrm{s}(t_{min})-\mathrm{s}(\mathcal{C}),\mathcal{C},\mathcal{O}} = 0$. For the first base case, the condition is true, since we return t'' in line 2. For the second case, the condition is true, since we return t_{min} in line 7. Assume that the nesting depth of function calls is greater 0. Then, by induction hypothesis, the function MINIMISE in line 6 returns a ground term so that the new value of t_{min} is ground and the condition is true due to the return statement in line 7.

(b). $\mathcal{O} \models D \equiv \text{con}(t)$: This is true if MINIMISE is called with terms t', t_{min} such that $\mathcal{O} \models D \equiv \text{con}(t_{\text{min}})$ and $\mathcal{O} \models D \equiv \text{con}(t')$, where t'' is the starting point for further derivations. It can be shown by induction on the nesting depth of function calls that, if MINIMISE is called with terms t', t_{min} such that $\mathcal{O} \models \text{con}(t') \equiv \text{con}(t_{\text{min}})$, then the function returns a term t such that $\mathcal{O} \models \text{con}(t) \equiv \text{con}(t_{\text{min}})$ and $\mathcal{O} \models \text{con}(t) \equiv \text{con}(t')$. We have two base cases: (1) the starting point for further derivations t'' is ground, and (2) $R^{E,\mathrm{s}(t_{min})-\mathrm{s}(\mathcal{C}),\mathcal{C},\mathcal{O}} = \emptyset$. For the first base case, the condition is true, since we return t'' in line 2. For the second case, the condition is true, since we return t_{min} in line 7. Assume that the nesting depth of function calls is greater 0. Then, as a consequence of Definition 3, for each term t' derived in line 5, we have $\mathcal{O} \models \text{con}(t_{\text{min}}) \equiv \text{con}(t')$. By induction hypothesis, the function MINIMISE in line 6 returns a new term t'_{min} such that $\mathcal{O} \models \text{con}(t_{\text{min}}) \equiv \text{con}(t'_{\text{min}})$. Thus, the condition is true due to the return statement in line 7. Since we initially call MINIMISE terms t', t_{min} such that $\mathcal{O} \models D \equiv \text{con}(t_{\text{min}})$ and $\mathcal{O} \models D \equiv \text{con}(t')$, where t'' is the starting point for further derivations, it follows that $\mathcal{O} \models D \equiv \text{con}(t)$.

(c). t is minimal: Here, we first show that, for an ontology $\mathcal{O}'$ and a term t'' such that $\mathfrak{n}_E \rightarrow_{(G^{\sqsubseteq}(\mathcal{O}',E))} t''$ for some $E \in \mathcal{E}_{\mathcal{O}'}$, the function COMPUTEALL$(t'', t_{\text{limit}}, \mathcal{O}')$ in Algorithm 3 computes a set of terms including all ground terms that are equivalent to t'' wrt. $\mathcal{O}'$, minimal in size and smaller than a given term t_{limit} (if the smallest ground term equivalent to t'' wrt. $\mathcal{O}'$ is not smaller than t_{limit}, it returns $\emptyset$). Let t' be such a term— a minimal ground term that is equivalent to t'' wrt. $\mathcal{O}'$ and that is smaller than a given term t_{limit}. Since $\mathfrak{n}_E \rightarrow_{(G^{\sqsubseteq}(\mathcal{O}',E))} t''$, it follows by Lemma 1 that also $t'' \rightarrow_{(G^{\sqsubseteq}(\mathcal{O}',E))} t'$, since $\mathcal{O}' \models \text{con}(t') \equiv \text{con}(t'')$, and, therefore $\mathcal{O}' \models \text{con}(\mathfrak{n}_E) \sqsubseteq \text{con}(t'')$. We consider the derivation of t' using $G^{\sqsubseteq}(\mathcal{O}', E)$. For every derivation step with a non-terminal $\mathfrak{n}_{E'}$ and a context $\mathcal{C}$, we show that the applied rule is contained in the set $R^{E',\mathrm{s}(t')-\mathrm{s}(\mathcal{C})+1,\mathcal{C},\mathcal{O}'}$. Let α be the rule applied to the non-terminal $\mathfrak{n}_{E'}$ within the context $\mathcal{C}$.

- Assume that α is of type (R1) with A on the right-hand side. Then, $A \in E' \cap (\text{sig}_C(\mathcal{O}') \cup \{\top\})$ so that $\alpha \in R^{E',\mathrm{s}(t')-\mathrm{s}(\mathcal{C})+1,\mathcal{C},\mathcal{O}'}$ by Definition 3 (DR1).
- Assume that α is of type (R2) with $\bigsqcap(S)$ on the right-hand side. Then, $S \subseteq \mathcal{S}^{\sqcap}_{E'}$ and $S \neq \emptyset$. Further, it follows by Lemma 2 Claim 1 that $\mathrm{s}(\mathcal{C}) \leq \mathrm{s}(t') - |S| + 1$ for any order of derivation steps within the derivation of t' including the case that $\mathfrak{n}_{E'}$ was the last non-terminal to be picked in line 3. Therefore, $\mathrm{s}(t') - \mathrm{s}(\mathcal{C}) + 1 \geq |S|$. The condition $\mathcal{O}' \models \text{con}(t') \equiv \text{con}(\mathcal{C}[\bigsqcap(S)/X])$ holds due to Lemma 2 Claim 2 and to $\mathcal{O}' \models \text{con}(t') \equiv \text{con}(t'')$. It remains to show that, unless $S = \{E_\top\}$, S is irreducible wrt. $\mathcal{O}'$. Assume that $S \neq \{E_\top\}$ and S is reducible wrt. $\mathcal{O}'$. Then, by Definition 2, there is a conjunct $E_i \in S$ such that $\mathcal{O}' \models \text{con}(\bigsqcap(S)) \equiv \text{con}(\bigsqcap(S \setminus \{E_i\}))$. In that case, there is a conjunct within t' that can be removed to obtain a smaller term t'' such that $\mathcal{O}' \models \text{con}(t') \equiv \text{con}(t'')$. This contradicts our assumption that t' is minimal. Thus, S has to be irreducible wrt. $\mathcal{O}'$. As a consequence of the five fulfilled conditions of Definition 3 (DR2), $\alpha \in R^{E',\mathrm{s}(t')-\mathrm{s}(\mathcal{C})+1,\mathcal{C},\mathcal{O}'}$.
- Assume that α is of type (R3) with $\exists r(\mathfrak{n}_{E_1})$ on the right-hand side. Then, $\langle r, \mathfrak{n}_{E_1} \rangle \in \text{Ex}_{\mathfrak{n}_{E'}}$ and, by Lemma 2 Claim 1, $\mathrm{s}(\mathcal{C}) \leq \mathrm{s}(t') - 1$ for any order of derivation steps within the derivation of t' including the case that $\mathfrak{n}_{E'}$ was the last non-terminal to be picked in line 3. Therefore, $\mathrm{s}(t') - \mathrm{s}(\mathcal{C}) + 1 \geq 2$ and, by Definition 3 (DR3), $\alpha \in R^{E',\mathrm{s}(t')-\mathrm{s}(\mathcal{C})+1,\mathcal{C},\mathcal{O}'}$.

We conclude that the function COMPUTEALL in Algorithm 3 computes a set containing all minimal ground terms that are equivalent to a term t'' wrt. $\mathcal{O}'$ and that are smaller than t_{limit}.

Now, we show that, assuming $\mathrm{s}(t'') < \mathrm{s}(t_{\text{min}})$, the function MINIMISE$(t'', t_{\text{min}}, \mathcal{O})$ returns t_{min} in case COMPUTEALL$(t'', t_{\text{min}}, \mathcal{O}) = \emptyset$ and, otherwise, a term from $\{t \mid t \in \text{COMPUTEALL}(t'', t_{\text{min}}, \mathcal{O})$, there is no $t' \in \text{COMPUTEALL}(t'', t_{\text{min}}, \mathcal{O})$ such that $\mathrm{s}(t') < \mathrm{s}(t)\}$. This is shown by induction on the nesting depth of function calls in MINIMISE$(t'', t_{\text{min}}, \mathcal{O})$. We have two base cases: (1) t'' is ground, and (2) $R^{E,\mathrm{s}(t_{min})-\mathrm{s}(\mathcal{C}),\mathcal{C},\mathcal{O}} = \emptyset$. In the first case, COMPUTEALL$(t'', t_{\text{limit}}, \mathcal{O})$ returns $\{t''\}$ in line 2 and MINIMISE$(t'', t_{\text{min}}, \mathcal{O})$ returns t'' in line 2 so that the induction hypothesis is true. In the second case, COMPUTEALL$(t'', t_{\text{min}}, \mathcal{O})$ returns $\emptyset$ in line 8 and MINIMISE$(t'', t_{\text{min}}, \mathcal{O})$ returns t_{min} in line 7 so that the induction hypothesis is true. Assume that the nesting depth of function calls is greater 0. Let $n = |R^{E,\mathrm{s}(t_{min})-\mathrm{s}(\mathcal{C}),\mathcal{C},\mathcal{O}}|$. Then, by induction hypothesis, for

each run $i \leq n$ of the for-loop, the value t_{min}^{i+1} returned by $\text{MINIMISE}(t'_i, t^i_{\text{min}}, \mathcal{O})$ in line 6 is given by t^i_{min} in case $\text{COMPUTEALL}(t'_i, t^i_{\text{min}}, \mathcal{O}) = \emptyset$ and, otherwise, by a term from $\{t \mid t \in \text{COMPUTEALL}(t'_i, t^i_{\text{limit}}, \mathcal{O})$, there is no $t''' \in \text{COMPUTEALL}(t'_i, t^i_{\text{min}}, \mathcal{O})$ such that $\text{s}(t''') < \text{s}(t)\}$. By induction on the length of the entire for-loop-iteration, we can show that, after the application of each rule i, the value of t_{min}^{i+1} is the minimum of the values $\{t^1_{\text{min}}\} \cup \bigcup_{j \leq i} \text{COMPUTEALL}(t'_j, t^j_{\text{min}}, \mathcal{O})$. Thus, also in this case, the function $\text{MINIMISE}(t'', t_{\text{min}}, \mathcal{O})$ returns t_{min} in case $\text{COMPUTEALL}(t'', t_{\text{min}}, \mathcal{O}) = \emptyset$ and, otherwise, a term from $\{t \mid t \in \text{COMPUTEALL}(t'', t_{\text{min}}, \mathcal{O})$, there is no $t' \in \text{COMPUTEALL}(t'', t_{\text{min}}, \mathcal{O})$ such that $\text{s}(t') < \text{s}(t)\}$. We can conclude that t in this theorem is indeed minimal. □

Proofs for Theorem 2

Theorem 2 *Let $\mathcal{O}$ be an $\mathcal{EL}$ ontology, $E_0 \in \mathcal{E}_\mathcal{O}$ and t a term such that $\mathfrak{n}_{E_0} \rightarrow^+_{G^\sqsubseteq(\mathcal{O},E_0)} t$. Let further $k \geq 1$ 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 $\text{COMPUTERULES}(E, k, \mathcal{C}, \mathcal{O})$ computes rules of type DR2 for $R^{E,k,\mathcal{C},\mathcal{O}}$ in time exponential in the size of $\mathcal{S}^{\nabla\!\!\!/}_E$ in the worst case.*

Proof. We divide the above theorem into three claims:

1. The function $\text{COMPUTERULES}(E, k, \mathcal{C}, \mathcal{O})$ computes every rule $\mathfrak{n}_E \rightarrow \sqcap(S_n)$ of type DR2 from $R^{E,k,\mathcal{C},\mathcal{O}}$.
2. The function $\text{COMPUTERULES}(E, k, \mathcal{C}, \mathcal{O})$ computes only rules of type DR2 from $R^{E,k,\mathcal{C},\mathcal{O}}$.
3. The function $\text{COMPUTERULES}(E, k, \mathcal{C}, \mathcal{O})$ terminates within time exponential in the size of $\mathcal{S}^{\nabla\!\!\!/}_E$.

We start by showing Claim 1: The function $\text{COMPUTERULES}(E, k, \mathcal{C}, \mathcal{O})$ computes every rule $\mathfrak{n}_E \rightarrow \sqcap(S_n)$ of type DR2 from $R^{E,k,\mathcal{C},\mathcal{O}}$. We can show by induction on the length of the set $S_n \setminus S^{\text{req}}$ that, if a subset S' of S_n is an element of M^{test} in line 8, then S_n will be added to M^{res} after $|S_n \setminus S^{\text{req}}| + 1$ iterations of the while-loop. Assume that there is a rule $\mathfrak{n}_E \rightarrow \sqcap(S_n)$ in $R^{E,k,\mathcal{C},\mathcal{O}}$. Then, $S_n \subseteq \mathcal{S}^{\nabla\!\!\!/}_E$, $k \geq n$, $\mathcal{O} \models \text{con}(t) \equiv \text{con}(\mathcal{C}[\bigsqcap(S_n)/X])$ and, unless $S_n = \{E_\top\}$, S_n is irreducible wrt. $\mathcal{O}$. In that case, the test in line 3 will be successful independent from the remaining elements of $\mathcal{S}^{\nabla\!\!\!/}_E$. We compute S^{req} in line 5 and add it to M^{test}. The base case of induction is that $S = S_n$ is in M^{test}. In this case, $s \leq k$ and the set M^{test} is not empty so that the condition of the while-loop in line 8 is true. For S_n, the condition in line 12 is true since, by assumption, $\mathcal{O} \models \text{con}(t) \equiv \text{con}(\mathcal{C}[\bigsqcap(S_n)/X])$, so that S_n is added to M^{res} within 1 iteration of the while-loop. Now, we assume that $S \neq S_n$ is in M^{test} in line 8. In this case, $s \leq k$ and the set M^{test} is not empty so that the condition of the while-loop in line 8 is true.

- In case $S = \emptyset$, the condition in line 12 is false and, since there is no S' with $\langle S', S\rangle \in \texttt{SUCC}$, we skip lines 16-20 so that $\emptyset$ is added to M^{expand} in line 22. In line 24, one of the sets from M^{expand} is $\emptyset$ and, in line 25, one of the processed elements of $\mathcal{S}^{\nabla\!\!\!/}_E$ is some $E' \in S_n$. All three conditions in line 26 are true, since $\langle \emptyset, E'\rangle \notin \texttt{EXCL}$ and $E' \notin \emptyset$ and $M^{\text{res}} = \emptyset$. In line 27, $\{E'\}$ is added to M^{test}. We reach line 8 with $\{E'\}$ in M^{test}. By induction hypothesis, S_n will be added to M^{res} after $|S_n \setminus \{E'\}| + 1$ iterations of the while-loop from now on. Thus, for $S = \emptyset$ it holds that S_n is added to M^{res} after $|S_n \setminus \emptyset| + 1$ iterations of the while-loop.
- In case $S \neq \emptyset$, the condition in line 12 is false, since S_n is irreducible wrt. $\mathcal{O}$. The condition in line 17 is false for every S' with $\langle S', S\rangle \in \texttt{SUCC}$ since S_n is irreducible wrt. $\mathcal{O}$ so that the variable *reducible* is still false in line 21. Consequently, S is added to M^{expand}. In line 24, one of the sets from M^{expand} is S and, in line 25, one of the processed elements of $\mathcal{S}^{\nabla\!\!\!/}_E$ is some $E' \in S_n \setminus S$. All three conditions in line 26 are true: $\langle S, E'\rangle \notin \texttt{EXCL}$ because S_n is irreducible wrt. $\mathcal{O}$; $E' \notin S$ by assumption; M^{res} cannot contain any proper subsets of S_n because S_n is irreducible wrt. $\mathcal{O}$. Consequently, in line 27, $S \cup \{E'\}$ is added to M^{test}. We reach line 8 with $S \cup \{E'\}$ in M^{test}. By induction hypothesis, S_n will be added to M^{res} after $|S_n \setminus (S \cup \{E'\})| + 1$ iterations of the while-loop from now on. Thus, for S it holds that S_n is added to M^{res} after $|S_n \setminus S| + 1$ iterations of the while-loop.

We now show Claim 2: The function $\text{COMPUTERULES}(E, k, \mathcal{C}, \mathcal{O})$ computes only rules of type DR2 from $R^{E,k,\mathcal{C},\mathcal{O}}$. We need to show for each set $S \in M^{\text{res}}$ that: (a) $S \neq \emptyset$; (b) $|S| \leq k$; (c) $\mathcal{O} \models \text{con}(t) \equiv \text{con}(\mathcal{C}[\bigsqcap(S)/X])$; (d) unless $S = \{E_\top\}$, S is irreducible wrt. $\mathcal{O}$. Let $S \in M^{\text{res}}$.

(a). $S \neq \emptyset$: This is true, since, in line 12, we test whether $S \neq \emptyset$ immediately before including S into M^{res}.

(b). $|S| \leq k$: If we examine lines 9,10,22,23,27 of the algorithm, we can see that the value of s as well as the size of elements within the sets M^{test} and M^{expand} increase by 1. Since, immediately before the first run of the while-loop, in line 7, we set s to the size of the element S^{req} of M^{test}, in every run of the while-loop in line 13, we have $s = |S|$. Due to the condition $s \leq k$ in line 8, it holds for each set S' added to M^{res} in line 13, that $|S'| \leq k$. Consequently, $|S| \leq k$.

(c). $\mathcal{O} \models \text{con}(t) \equiv \text{con}(\mathcal{C}[\bigsqcap(S)/X])$: In line 1, we define D as $\text{con}(\mathcal{C}[\mathfrak{n}_E/X])$ and in Definition 3, we define $t = \mathcal{C}[\mathfrak{n}_E/X]$ so that $D = \text{con}(t)$. In line 12, we test whether $\mathcal{O} \models D \equiv \text{con}(\mathcal{C}[\bigsqcap(S)/X])$ immediately before adding S to M^{res}. Thus, $\mathcal{O} \models \text{con}(t) \equiv \text{con}(\mathcal{C}[\bigsqcap(S)/X])$.

(d). Unless $S = \{E_\top\}$, S is irreducible wrt. $\mathcal{O}$: The irreducibility is a consequence of the third condition in line 26 and can be shown as follows. First, note that, as a consequence of the definition of $\mathcal{E}_\mathcal{O}$, for every $E \in \mathcal{E}_\mathcal{O} \setminus \{E_\top\}$, we have $\mathcal{O} \not\models \text{con}(E) \equiv \top$. Thus, all sets $\{E\} \subseteq (\mathcal{E}_\mathcal{O} \setminus \{E_\top\})$ are irreducible wrt. $\mathcal{O}$. Assume that $S \neq \{E_\top\}$ and that S is reducible wrt. $\mathcal{O}$. By induction on the size of S, we can show that there is an irreducible, non-empty subset S'' of S such that $\mathcal{O} \models \text{con}(\bigsqcap(S)) \equiv \text{con}(\bigsqcap(S''))$. Then, $\mathcal{O} \models \text{con}(t) \equiv \text{con}(\mathcal{C}[\bigsqcap(S'')/X])$.

Since $|S''| < |S|$, we have $|S''| \leq k$ so that S'' fulfills all conditions for rules of type (DR2) in Definition 3. Thus, as a consequence of Claim 1 of this proof, S'' must be an element of M^{res} at the point when S is being created. However, in line 26, immediately before creating S, we make sure that no set $S''' \subset S$ with $S''' \in M^{\text{res}}$ exists. Thus, our assumption was wrong and S must be irreducible wrt. $\mathcal{O}$.

At last, we show Claim 3: The function COMPUTERULES$(E, k, \mathcal{C}, \mathcal{O})$ terminates within time exponential in the size of $\mathcal{S}_E^{\not\top}$. Computations in line 3 and 5 require 1 and $m = |\mathcal{S}_E^{\not\top}|$ polynomial-time reasoning calls, respectively. The while-loop is executed at most k times. In every loop run, lines 12-22 are executed $|M^{\text{test}}|$ times and lines 25-29 $|M^{\text{expand}}|$ times. The maximum size of the two sets is $\binom{m}{k}$ and $\binom{m}{k+1}$, respectively. In lines 12-22, we need maximally k polynomial-time reasoning calls and k^2 constant-time insertions into `EXCL`. In lines 25-29, we only need a linear (in the size of m) number of constant-time operations including look-ups and insertions. Thus, overall, $\binom{m}{k}$ and $\binom{m}{k+1}$ are the dominating factor within the function COMPUTERULES$(E, k, \mathcal{C}, \mathcal{O})$. Since k only has effect on the computation up to the size of $\mathcal{S}_E^{\not\top}$, we conclude that COMPUTERULES$(E, k, \mathcal{C}, \mathcal{O})$ terminates within time exponential in the size of $\mathcal{S}_E^{\not\top}$.

□