Anti-Unification of Concepts in Description Logic $\mathcal{EL}$

Boris Konev
University of Liverpool
United Kingdom

Temur Kutsia
RISC, Johannes Kepler University Linz
Austria

Abstract

We study anti-unification for the description logic $\mathcal{EL}$ and introduce the notion of least general generalisation, which generalises simultaneously least common subsumer and concept matching. The idea of generalisation of two concepts is to detect maximal similarities between them, and to abstract over their differences uniformly. We demonstrate that a finite minimal complete set of generalisations for $\mathcal{EL}$ concepts always exists and establish complexity bounds for computing them. We present an anti-unification algorithm that computes generalisations with a fixed skeleton, study its properties and report on preliminary experimental evaluation.

Introduction

Description Logics as a knowledge representation formalism gained particular prominence in recent years due to widespread adoption of the web ontology language OWL as a W3C web standard (2012). Not only does the strong link between the direct model theoretic semantics of OWL 2 and the semantics of description logics provide OWL ontologies with an unambiguous meaning but it also enables one to harvest the power of logical reasoning in various ontology application scenarios, see e.g. (Baader et al. 2003; Yu 2014; Domingue, Fensel, and Hendler 2011) for more details.

Capturing expert knowledge and representing it in the form of concept descriptions and axioms is a laborious and time consuming task, which is further hindered by the fact that domain experts may disagree on basic definitions, knowledge engineers may choose to describe different concepts at different levels of granularity, different names can denote semantically equivalent concepts, concept description can be machine learned etc., which leads to the need to be able to consolidate and unify different concept descriptions into one, best suitable for a particular application.

The notion of a *least common subsumer* (lcs for short) has been introduced in (Cohen, Borgida, and Hirsh 1992) precisely to capture 'the largest set of commonalities' between concepts. An lcs of two concepts is a concept C that subsumes both of them and such that no other common subsumer of the given concepts is strictly subsumed by C. It can be seen that while such a concept indeed captures the commonalities between the given concepts, it does not highlight differences between them nor suggest a way to consolidate such differences into a new concept description. A survey of results on lcs can be found in (Baader and Küsters 2006).

The problem of identifying and consolidating differences between concepts has been addressed in the context of concept matching and concept unification. For example, Baader and Morawska (2010) give the following example of the use of unification to eliminate redundancies from ontologies: concept descriptions $\mathsf{Human} \sqcap \mathsf{Male} \sqcap \exists \mathsf{loves}.\mathsf{Sports\_car}$ and $\mathsf{Man} \sqcap \exists \mathsf{loves}.(\mathsf{Car} \sqcap \mathsf{Fast})$ intuitively refer to the same concept of a 'man loving fast cars', yet they are clearly not equivalent. Differences in the representation can be resolved by treating Man and $\mathsf{Sports\_car}$ as *concept variables* and unifying the two concept descriptions with the substitution $\{\mathsf{Man} \mapsto \mathsf{Human} \sqcap \mathsf{Male}, \mathsf{Sports\_car} \mapsto \mathsf{Car} \sqcap \mathsf{Fast}\}$. While powerful, this approach requires ontology engineers to identify which concept names should be treated as constants and which should be treated as variables, which may not always be obvious.

In this paper we propose a novel way of identifying and consolidating differences between concept descriptions based on the notion of *concept generalisation* by *anti-unification*. Speaking abstractly, a generalisation of two terms s_1 and s_2 is a term t such that s_i can be 'obtained' from t by applying some substitution τ_i to t, $i = 1, 2$. Notice that every two terms always have a generalisation $t = X$, where X is a variable, known as the *most* general generalisation. Interesting cases are *least* general generalisations (lgg for short), which retain the common parts of the input terms as much as possible, and abstract with the help of variables over the differences in the input uniformly. Anti-unification is a technique that has been successfully used to compute lggs in various theories, starting from the pioneering works by Plotkin (1970) and Reynolds (1970).

We study the anti-unification problem for concepts in the description logic $\mathcal{EL}$, which underpins the OWL 2 EL profile (Baader, Brandt, and Lutz 2005). In the DL context terms are concepts and the informal notion of 'obtaining the original concepts from the generalisation by substitutions' can be specialised in two ways: $\tau_i(t)$ is equivalent to s_i (generalisation modulo equivalence), or $\tau_i(t)$ subsumes s_i (generalisation modulo subsumption). These notions are motivated by matching modulo equivalence and subsumption, respectively (Baader and Küsters 2006).

Note that conjunction is idempotent. Interestingly, it was shown by Pottier (1989) that anti-unification in equational theories with *two* idempotent function symbols is infinitary. In contrast, we show that generalisation modulo subsumption coincides with the *lcs*, and a finite minimal complete set of generalisations modulo equivalence always exists but can be non-elementary in the size of the given concepts.

In the course of looking for lower complexity variants, we define fixed-skeleton exact generalisations, where we assume that the underlying tree structure of the generalisation (its skeleton) is fixed and contains only 'essential' nodes, and the goal it to minimise the 'variable part'. We design an algorithm that solves this problem when the homomorphisms from the skeleton to the original concepts are also given, and prove that it is terminating, sound, and complete.

One advantage of our approach is that not only does it compute generalisations, but also provides information how the variables can be instantiated to obtain the original concepts. This is given in the form of so called anti-unification triples (AUTs in short) of the form $X : [\mathsf{L}_1, \mathsf{U}_1] \triangleq [\mathsf{L}_2, \mathsf{U}_2]$. Such a triple indicates that the original concepts C_1 and C_2 differ at the nodes where X occurs and tells us that replacing X by any concept between (in the sense of subsumption) the lower bound L_i and the upper bound U_i will give the corresponding node in C_i, for $i = 1, 2$.

Knowledge engineer might find the information provided by the computed generalisation and the AUTs useful: She can clearly see (in the generalisation) where the concepts in question agree, and observe (in the AUTs) how those concepts differ. Moreover, if an upper bound in an AUT, e.g., U_1, is a concept name, it can be treated as a unification variable with a possible instantiation by U_2. For instance, in the 'man loving fast cars' example above, our algorithm returns the generalisation $X \sqcap \exists \mathsf{loves}.Y$ and two AUTs: $X : [\dots, \mathsf{Human} \sqcap \mathsf{Male}] \triangleq [\dots, \mathsf{Man}]$ and $Y : [\dots, \mathsf{SportsCar}] \triangleq [\dots, \mathsf{Car} \sqcap \mathsf{Fast}]$ (we omit the lower bounds for brevity). From here one can conclude that if Man is defined as $\mathsf{Human} \sqcap \mathsf{Male}$ and $\mathsf{SportsCar}$ is defined as $\mathsf{Car} \sqcap \mathsf{Fast}$, the original concepts become equivalent. Thus, similarly to the *lcs*, anti-unification can be used in modelling of concepts based on available concept variants, but it additionally provides insights into differences between such variants.

In our preliminary experiments we have successfully computed generalisations of 49 675 528 pairs from definitions of fully defined concepts of the $\mathcal{EL}$ variant of the GALEN ontology (Kazakov and Klinov 2015; Rector et al. 2003) revealing commonalities between concepts.

Due to space restriction, some technical proofs are deferred to the full version published as RISC Technical Report, `http://www.risc.jku.at/publications/`.

Preliminaries

Let $\mathsf{N_C}$ and $\mathsf{N_R}$ be countably infinite and disjoint sets of *concept names* and *role names*, respectively. In the description logic $\mathcal{EL}$, *concepts* C are built according to the syntax rule

$$C ::= A \mid \exists r.C \mid \textstyle\bigsqcap_{C \in \mathcal{C}} C,$$

where A ranges over $\mathsf{N_C}$, r ranges over $\mathsf{N_R}$, and the expression $\bigsqcap_{C \in \mathcal{C}} C$ is a *conjunction* over the (multi)set of concepts $\mathcal{C}$, where no $C \in \mathcal{C}$ is a conjunction in turn. When $\mathcal{C} = \{C_1, \dots, C_n\}$, for $n \geq 2$, we write $C_1 \sqcap \dots \sqcap C_n$. The conjunction over the empty set is abbreviated as $\top$.

The semantics of concepts is defined by means of *interpretations* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where the interpretation *domain* $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is a function mapping each concept name A to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and each role name r to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The function $\cdot^{\mathcal{I}}$ is inductively extended to arbitrary concepts by setting $(\bigsqcap_{C \in \mathcal{C}})^{\mathcal{I}} := \bigcap_{C \in \mathcal{C}} C^{\mathcal{I}}$, and $(\exists r.C)^{\mathcal{I}} := \{d \in \Delta^{\mathcal{I}} \mid \exists e \in C^{\mathcal{I}} : (d, e) \in r^{\mathcal{I}}\}$. For $\top$ we have $\top^{\mathcal{I}} := \Delta^{\mathcal{I}}$. Concept D *subsumes* concept C, in symbols $C \sqsubseteq D$, if for every interpretation $\mathcal{I}$ we have $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Concept C is *equivalent* to concept D, in symbols $C \equiv D$, if both $C \sqsubseteq D$ and $D \sqsubseteq C$. It is easy to see that thus defined concepts and interpretations are equivalent to the standard definition with binary conjunction (Baader, Brandt, and Lutz 2005).

A concept C is a *least common subsumer (lcs)* of concepts C_1 and C_2 if $C_i \sqsubseteq C$, for $i = 1, 2$, and for any other concept D such that $C_i \sqsubseteq D$, for $i = 1, 2$, we have $C \sqsubseteq D$. For a set of concepts $S = \{C_1, \dots, C_n\}$ we define $\mathit{lcs}(S)$ as $\mathit{lcs}(C_1, \mathit{lcs}(C_2, \dots, C_n) \dots)$. It is known that any $\mathcal{EL}$ concepts C_1, C_2 always have a unique least common subsumer, and *lcs* is commutative and associative, so $\mathit{lcs}(S)$ is correctly defined (Baader, Küsters, and Molitor 1999).

$\mathcal{EL}$ Anti-Unification

To define the anti-unification problem, we partition the set of concept names $\mathsf{N_C}$ into *concept constants* $\mathsf{N}_{\mathsf{C}a}$ and *concept variables* $\mathsf{N}_{\mathsf{C}v}$. We say that a concept is *ground* if it does not contain any concept variables. A substitution σ is a mapping from the set of concept variables to the set of $\mathcal{EL}$ concepts such that $\sigma(X) \neq X$ for finitely many $X \in \mathsf{N}_{\mathsf{C}v}$. The expression $\sigma : \{X_1 \mapsto C_1, \dots X_n \mapsto C_n\}$ denotes that $\sigma(X_i) = C_i$, for $i = 1, \dots, n$ and implicitly $\sigma(Y) = Y$ for all $Y \in \mathsf{N}_{\mathsf{C}v} \setminus \{X_1, \dots, X_n\}$. Substitutions are extended to $\mathcal{EL}$ concepts in the usual way: $\sigma(A) = A$, for $A \in \mathsf{N}_{\mathsf{C}a}$; $\sigma(\bigsqcap_{C \in \mathcal{C}} C) = \bigsqcap_{C \in \mathcal{C}} \sigma(C)$; $\sigma(\exists r.C) = \exists r.\sigma(C)$.

We say that a concept D is *more general* than a concept C *modulo equivalence*, denoted $D \preceq_{\equiv} C$ (more general *modulo subsumption*, denoted $D \preceq_{\sqsubseteq} C$) if there exists a substitution σ such that $C \equiv \sigma(D)$ (or $C \sqsubseteq \sigma(D)$, respectively). A concept G is a *generalisation* of concepts C_1 and C_2 *modulo equivalence* (a generalisation *modulo subsumption*) if it is more general than both C_1 and C_2, that is, if there exist substitutions τ_1 and τ_2 such that $C_i \equiv \tau_i(G)$ (or $C_i \sqsubseteq \tau_i(G)$, respectively), for $i = 1, 2$. Notice that G is a generalisation of C_1, C_2 modulo equivalence (or subsumption) iff there exists matchers (Baader and Küsters 2006) for the matching problems $C_i \equiv^? G$ (or $C_i \sqsubseteq^? G$, respectively), for $i = 1, 2$.

Any concepts C_1, C_2 always have some generalisation: consider $G = X$, where $X \in \mathsf{N}_{\mathsf{C}v}$ is fresh, and $\tau_i(X) = C_i$, for $i = 1, 2$. Then G is a generalisation of C_1, C_2 both modulo equivalence and subsumption. Obviously, such a generalisation is too crude as often less general generalisations exist. For example, for $C_1 = A \sqcap B$ and $C_2 = A \sqcap B'$ the generalisation $G' = A \sqcap Y$ is less general than $G = X$, as G' can be obtained from G by substituting $A \sqcap Y$ into X. This leads us to the following definition.

If $C \preceq_{\equiv} D$ and $D \preceq_{\equiv} C$ then we say that C and D are *equi-general* modulo equivalence, denoted $C \approx_{\equiv} D$ (equi-generality modulo subsumption is defined similarly). Then a generalisation G of concepts C_1 and C_2 is *least general* (abbreviated as *lgg*) if whenever a generalisation G' of C_1 and C_2 is less general than G then G' is equi-general to G.

It turns out that least general generalisations modulo subsumption coincide with the least common subsumer and so can be computed in polynomial time.

Proposition 1 *For any $\mathcal{EL}$ concepts C_1 and C_2 a least general generalisation modulo subsumption always exists and is equi-general to $lcs(C_1, C_2)$.*

Proof. First notice that $lcs(C_1, C_2)$ is a generalisation of C_1 and C_2 modulo subsumption as by definition of the least common subsumer we have $C_i \sqsubseteq \sigma_{id}(lcs(C_1, C_2))$, where σ_{id} is the identity substitution.

Let G be an arbitrary generalisation of C_1, C_2 modulo subsumption. Then $C_i \sqsubseteq \tau_i(G)$, for some τ_i and $i = 1, 2$. But then by (Baader and Küsters 2006) we have $C_i \sqsubseteq \sigma_\top(G)$, where $\sigma_\top$ is the substitution that replaces every variable in G with $\top$. By the properties of the lcs we have $lcs(C_1, C_2) \sqsubseteq \sigma_\top(G)$. So $G \preceq_{\sqsubseteq} lcs(C_1, C_2)$.

Thus $lcs(C_1, C_2)$ is a least general generalisation of C_1 and C_2 and any other least general generalisation of C_1 and C_2 is equi-general to $lcs(C_1, C_2)$. ❑

In the view of Proposition 1, from now on we only consider generalisations modulo equivalence and use $\preceq$ and $\approx$ without any indices. Unlike the modulo subsumption case, there exist incomparable *lgg*s. For example, for $C_1 = \exists r.(A \sqcap B) \sqcap \exists r(A' \sqcap B')$ and $C_2 = \exists r.(A \sqcap A') \sqcap \exists r(B \sqcap B')$ both $\exists r.(A \sqcap X) \sqcap \exists r.(B' \sqcap Y)$ and $\exists r.(B \sqcap X) \sqcap \exists r.(A' \sqcap Y)$ are (incomparable) *lgg*s.

We say that a set of generalisations $\mathcal{S}$ of concepts C_1 and C_2 is *complete* (for C_1 and C_2) if for any generalisation G' of C_1 and C_2 there exists $G \in \mathcal{S}$ such that $G \preceq G'$. The set $\mathcal{S}$ is a *minimal complete set* of generalisations of C_1 and C_2 (written $mcsg(C_1, C_2)$) if it, in addition, satisfies the minimality property: For no two distinct $G_1, G_2 \in \mathcal{S}$, $G_1 \preceq G_2$ holds. Hence, the elements of $mcsg(C_1, C_2)$ are all the *lgg*s of C_1 and C_2.

Theorem 2 *For every $\mathcal{EL}$ concepts C_1, C_2, a finite minimal complete set of generalisations exists.*

Proof. Let G be a generalisation of C_1 and C_2. It follows from Lemma 6.3.1 in (Küsters 2001) that there exist substitutions τ_1 and τ_2 such that $C_i \equiv \tau_i(G)$, for $i = 1, 2$, and for every concept variable X occurring in G its image $\tau_i(X)$ is equivalent to the conjunction of some elements of $\mathsf{sc}(C_i)$, where $\mathsf{sc}(C)$ is the set of subconcepts of a concept C defined recursively as: $\mathsf{sc}(\top) = \{\top\}$, $\mathsf{sc}(A) = \{\top, A\}$, $\mathsf{sc}(\exists r.C) = \{\top\} \cup \{\exists r.C' \mid C' \in \mathsf{sc}(C)\}$, $\mathsf{sc}(C \sqcap D) = \{C' \sqcap D' \mid C' \in \mathsf{sc}(C), D' \in \mathsf{sc}(D)\}$. Let $N = 2^{|\mathsf{sc}(C_1)|} \times 2^{|\mathsf{sc}(C_2)|}$. Notice that if G contains more than N different concept variables, then for some X and Y we have $\tau_i(X) \equiv \tau_i(Y)$, for $i = 1, 2$. Then a substitution $\sigma : X \mapsto Y$ maps G into a generalisation G' containing fewer variables than G.

Let $\mathcal{S}$ be the set of all generalisations of C_1, C_2 such that every $G \in \mathcal{S}$ contains at most N different concept variables. We can assume w.l.o.g. for every $G \in \mathcal{S}$ that G only uses variable from $\{X_1, \ldots, X_N\}$ (this can be achieved by renaming variables). It should be obvious that the role depth of every generalisation of C_1, C_2 does not exceed the maximal role depth of C_1, C_2. But then the number of different, up to equivalence, concepts in $\mathcal{S}$ is finite.

So, one can select a finite subset $\mathcal{S}' \in \mathcal{S}$ such that for every generalisation G of C_1 and C_2 there exists $G' \in \mathcal{S}'$ with $G \preceq G'$ and for no distinct $G_1, G_2 \in \mathcal{S}'$ we have $G_1 \preceq G_2$. Then for every *lgg* G of C_1 and C_2 there exists an *lgg* $G' \in \mathcal{S}'$ equi-general to G, that is, $\mathcal{S}'$ is a finite complete set of *lgg*s as required. ❑

The proof of Theorem 2 gives a non-elementary upper bound on the size of *lgg*s. As the following example shows this upper bound can be reached.

Example 3 *Let $n > 0$ be even. Consider concepts*

$$C_1 := \underbrace{\exists r \cdots \exists r}_{n}.\left(\prod_{i=1}^{n}(A_i^1 \sqcap A_i^2)\right) \sqcap \prod_{i=1}^{n}(\exists s_i^1.A_i^1 \sqcap \exists s_i^2.A_i^2)$$

and

$$C_2 := \underbrace{\exists r \cdots \exists r}_{n}.\left(\prod_{i=1}^{n}(A_i^1 \sqcap A_i^2)\right) \sqcap \prod_{i=1}^{n}(\exists s_i^1.A_i^2 \sqcap \exists s_i^2.A_i^1).$$

Let the set of concepts $\mathcal{C}_0$ be defined as

$$\left\{ \prod_{i \in S}(A_i^1 \sqcap A_i^2) \sqcap \prod_{i \notin S}(X_i^1 \sqcap X_i^2) \;\middle|\; \begin{array}{c} S \subset \{1, \ldots, n\}, \\ |S| = n/2 \end{array} \right\}.$$

We define for every $0 \leq i < n$

$$\mathcal{C}_{i+1} := \left\{ \exists r.\left(\prod_{C \in \mathcal{C}} C\right) \;\middle|\; \mathcal{C} \subseteq \mathcal{C}_i, |\mathcal{C}| = \tfrac{1}{2}|\mathcal{C}_i| \right\}.$$

One can see that

$$G := \prod_{C \in \mathcal{C}_n} C \sqcap \prod_{i=1}^{n}(\exists s_i^1.X_i^1 \sqcap \exists s_i^2.X_i^2).$$

is an lgg of C_1 and C_2. Indeed, for any substitutions τ_1, τ_2 such that $\tau_i(G) \equiv C_1$, for $i = 1, 2$, we have $\tau_1(X_i^1) = A_i^1$ and $\tau_1(X_i^2) = A_i^2$, while $\tau_2(X_i^1) = A_i^2$ and $\tau_2(X_i^1) = A_i^2$. Thus, for any substitution σ such that $G' = \sigma(G)$ is a generalisation of C_1 and C_2, there exists a substitution σ' such that $\sigma'(G') \equiv G$.

There are c^n elements in $\mathcal{C}_0$, for some $c > 1$, c^{c^n} elements in $\mathcal{C}_1$, $c^{c^{c^n}}$ elements in $\mathcal{C}_3$ etc. Thus, the size of G is non-elementary in terms of n.

Notice, however, that the concept

$$\exists r \cdots \exists r.\left(\prod_{i=1}^{n}(A_i^1 \sqcap A_i^2)\right) \sqcap \prod_{i=1}^{n}(\exists s_i^1.X_i^1 \sqcap \exists s_i^2.X_i^2)$$

is a polynomial size lgg of C_1, C_2, incomparable with G.

Fixed skeleton generalisation

Example 3 demonstrates that without restraints least general generalisation can be of size non-elementary in the size of given concepts. Moreover, the notion of an *lgg* introduced in the previous section may not always be intuitive as it is not 'monotone' regarding modifications to the given concepts.

Consider, for example, $C_1 = C_2 = \exists r.A \sqcap \exists r.B$. Then, as one would expect, $G = \exists r.A \sqcap \exists r.B$ is an *lgg* of C_1 and C_2; $X \sqcap G$ is an *lgg* of $D \sqcap C_1$ and $E \sqcap C_2$; and $\exists s.Y \sqcap G$ is an *lgg* of $\exists s.A \sqcap C_1$ and $\exists s.B \sqcap C_2$. However, for $C_1' = \exists s.A \sqcap \exists t.(D \sqcap C_1)$ and $C_2' = \exists s.B \sqcap \exists t.(E \sqcap C_2)$, concept $G' = \exists s.Y \sqcap \exists t.(X \sqcap G)$, counter to expectations, is not an *lgg* of C_1' and C_2' as $\{X \mapsto Z \sqcap \exists r.Y\}$ maps G' into $G'' = \exists s.Y \sqcap \exists t.(Z \sqcap \exists r.A \sqcap \exists r.B \sqcap \exists r.Y)$, which is a generalisation of C_1', C_2' strictly less general than G'.

More control can be gained by restricting generalisations to have a fixed tree structure or *skeleton*. Formally the skeleton $\mathsf{skel}(C)$ of a concept C is the concept obtained from C by removing all occurrences of variables.

We say that a concept G is a *generalisation* of concepts C_1, C_2 *with a fixed skeleton* G_{sk} iff G is a generalisation of C_1, C_2 and $\mathsf{skel}(G) = G_{\mathsf{sk}}$. We say that G is an *lgg* of concepts C_1, C_2 with a fixed skeleton G_{sk} if G is a generalisation of C_1, C_2 with a fixed skeleton G_{sk} and whenever a generalisation G' of C_1 and C_2 with the same skeleton G_{sk} is less general than G then G' is equi-general to G.

It can be readily checked that results of Theorem 2 transfer to the fixed skeleton case. Hence, for every $\mathcal{EL}$ concepts C_1, C_2 a finite minimal complete set of generalisations with a fixed skeleton G_{sk} always exists (possibly empty if C_1 and C_2 do not have generalisations with skeleton G_{sk}).

To develop an algorithm computing fixed skeleton *lgg*s, following (Baader and Küsters 2006), we use a structural characterisation of subsumption. We identify each $\mathcal{EL}$ concept C with a finite *description tree* $\mathfrak{T}_C$ whose nodes are labelled with sets of concept names and whose edges are labelled with role names. In detail, if C is a concept name A or $\top$, then $\mathfrak{T}_C$ has a single node $\mathbf{d}_C$ with label $l(\mathbf{d}_C) = \{A\}$ if $C = A$, or $l(\mathbf{d}_C) = \emptyset$ if $C = \top$; if $C = \exists r.D$, then $\mathfrak{T}_C$ is obtained from $\mathfrak{T}_D$ by adding a new root $\mathbf{d}_C$ and an edge from $\mathbf{d}_C$ to the root $\mathbf{d}_D$ of $\mathfrak{T}_D$ with the label $l(\mathbf{d}_C, \mathbf{d}_D) = r$ (we then call $\mathbf{d}_D$ an immediate *r-successor* of $\mathbf{d}_C$); if $C = \bigsqcap_{i=1}^{n} C_i$, for $n > 0$, then $\mathfrak{T}_C$ is obtained by identifying the roots $\mathbf{d}_{C_i}$ of all $\mathfrak{T}_{C_i}$, $1 \leq i \leq n$, into $\mathbf{d}_C$ and setting $l(\mathbf{d}_C) = \bigcup_{i=1}^{n} l(\mathbf{d}_{C_i})$. We write $\mathbf{root}(C)$ for the root node of $\mathfrak{T}_C$. Conversely, every tree $\mathfrak{T}$ of the described form gives rise to an $\mathcal{EL}$ concept $C_{\mathfrak{T}}$ in the obvious way.

A concept D subsumes a concept C iff there exists a *homomorphism* from $\mathfrak{T}_D$ to $\mathfrak{T}_C$ defined as a function φ from the nodes of $\mathfrak{T}_D$ to the nodes of $\mathfrak{T}_C$ satisfying the following properties (Baader and Küsters 2006):

1. $\varphi(\mathbf{root}(D)) = \mathbf{root}(C)$;
2. for all $\mathbf{d}_1$, $\mathbf{d}_2$ nodes of $\mathfrak{T}_D$ and $r \in \mathsf{N_R}$ such that $\mathbf{d}_2$ is an r-successor of $\mathbf{d}_1$ in $\mathfrak{T}_D$, we have that $\varphi(\mathbf{d}_2)$ is an r-successor of $\varphi(\mathbf{d}_1)$;
3. for every node $\mathbf{d}$ of $\mathfrak{T}_D$, we have $l(\mathbf{d}) \subseteq l(\varphi(\mathbf{d}))$.

We say that a function φ is a *variable ignoring* homomorphism from D to C if condition 3 above is replaced with

3'. for every node $\mathbf{d}$ of $\mathfrak{T}_D$, we have $(l(\mathbf{d}) \cap \mathsf{N_{C}}_a) \subseteq l(\varphi(\mathbf{d}))$.

Homomorphisms are extended from nodes to sets of nodes in the usual way: $\varphi(S) := \cup_{\mathbf{d} \in S}\{\varphi(\mathbf{d})\}$.

We do not always distinguish explicitly between a concept and its tree representation and between nodes and subtrees rooted at the nodes, which allows us to speak, for example, about the nodes and subtrees of an $\mathcal{EL}$ concept, apply substitutions to nodes and consider homomorphisms to be functions between concepts. We also treat a variable ignoring homomorphism from D to C as a homomorphism from $\mathsf{skel}(D)$ to C and vice versa.

While our methods can be applied to the general case, for the sake of presentation in this paper we restrict our consideration to *exact generalisations*. Intuitively, exactness requires the skeletons to contain only essential nodes that match the corresponding nodes in the input concepts entirely. We say that a concept G is an exact generalisation of concepts C_1 and C_2 if there exist substitutions τ_1 and τ_2 and variable ignoring homomorphisms φ_1 and φ_2 from G to C_1, C_2, respectively, such that for every node $\mathbf{d}$ of G we have $\tau_i(\mathbf{d}) \equiv \varphi_i(\mathbf{d})$, for $i = 1, 2$.

Since $\mathbf{root}(G)$ is mapped by φ_i into $\mathbf{root}(C_i)$, for $i = 1, 2$, we have $\tau_i(G) \equiv C_i$, so every exact generalisation is a generalisation. For example, for $C_1 = \exists r.(A \sqcap B)$ and $C_2 = \exists r.A \sqcap \exists r.B$ both $G_1 = \exists r.(A \sqcap X) \sqcap \exists r.(B \sqcap X)$ and $G_2 = \exists r.A \sqcap \exists r.(B \sqcap Y)$ are least general generalisations with the same skeleton $G_{\mathsf{sk}} = \exists r.A \sqcap \exists r.B$ and homomorphisms $\varphi_i : G_{\mathsf{sk}} \to C_i$, for $i = 1, 2$, are uniquely determined; however, G_1 is exact while G_2 is not. The notions of an exact *lgg* and of an exact (least general) generalisation with a fixed skeleton are defined in the obvious way.

Looking back at the notions of generalisation we use in this paper, one can see that we started with unrestricted generalisation and then tried to make it more specific by introducing fixed skeletons. Further, we defined a variant that we called exact generalisation, and its version with a fixed skeleton. In the definition of the latter (that has not been explicitly spelled), the existence of the corresponding homomorphisms are asserted. We now make a step further and define exact generalisations with a fixed skeleton when the homomorphisms are given.

We say that $\mathfrak{F} = (G_{\mathsf{sk}}, \varphi_1, \varphi_2)$ is a *fixed skeleton exact generalisation framework* (or simply generalisation framework for short) for concepts C_1 and C_2 if $lcs(C_1, C_2) \sqsubseteq G_{\mathsf{sk}}$ and $\varphi_i : G_{\mathsf{sk}} \to C_i$, for $i = 1, 2$, are homomorphisms. We say that that a concept G is an *exact generalisation* of concepts C_1 and C_2 *w.r.t. a generalisation framework* $\mathfrak{F}$ if $\mathsf{skel}(G) = G_{\mathsf{sk}}$, and there exist substitutions τ_1 and τ_2 (called *witness substitutions*) such that for every node $\mathbf{d}$ of G we have $\varphi_i(\mathbf{d}) \equiv \tau_i(\mathbf{d})$. An exact *lgg* w.r.t. $\mathfrak{F}$ is defined in the obvious way.

In what follows we develop a non-deterministic polynomial time algorithm that given concepts C_1, C_2 and a generalisation framework $\mathfrak{F}$ computes an exact *lgg* G w.r.t. $\mathfrak{F}$. To achieve that, we characterise substitutions τ_i that witness G being an exact *lgg* w.r.t $\mathfrak{F}$ in terms of *anti-unification triples*,

(m): Merge

$$\frac{X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X], \quad Y : [\mathsf{L}_1^Y, \mathsf{U}_1^Y] \triangleq [\mathsf{L}_2^Y, \mathsf{U}_2^Y]}{Z : [lcs(\mathsf{L}_1^X, \mathsf{L}_1^Y), \mathsf{U}_1^X \sqcap \mathsf{U}_1^Y] \triangleq [lcs(\mathsf{L}_2^X, \mathsf{L}_2^Y), \mathsf{U}_2^X \sqcap \mathsf{U}_2^Y]} \ \{X \mapsto Z, Y \mapsto Z\}$$

(sm): Split-merge

$$\frac{X : [\mathsf{L}_1^X, \mathsf{U}_1^X \sqcap \mathsf{U'}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X \sqcap \mathsf{U'}_2^X], \quad Y : [\mathsf{L}_1^Y, \mathsf{U}_1^Y] \triangleq [\mathsf{L}_2^Y, \mathsf{U}_2^Y]}{\begin{array}{c} Z : [lcs(\mathsf{L}_1^X, \mathsf{L}_1^Y), \mathsf{U}_1^X \sqcap \mathsf{U}_1^Y] \triangleq [lcs(\mathsf{L}_2^X, \mathsf{L}_2^Y), \mathsf{U}_2^X \sqcap \mathsf{U}_2^Y] \\ X' : [\mathsf{L}_1^X, \mathsf{U'}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U'}_2^X] \end{array}} \ \{X \mapsto Z \sqcap X', Y \mapsto Z\}$$

(ssm): Split-split-merge

$$\frac{X : [\mathsf{L}_1^X, \mathsf{U}_1^X \sqcap \mathsf{U'}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X \sqcap \mathsf{U'}_2^X], \quad Y : [\mathsf{L}_1^Y, \mathsf{U}_1^Y \sqcap \mathsf{U'}_1^Y] \triangleq [\mathsf{L}_2^Y, \mathsf{U}_2^Y \sqcap \mathsf{U'}_2^Y]}{\begin{array}{c} Z : [lcs(\mathsf{L}_1^X, \mathsf{L}_1^Y), \mathsf{U}_1^X \sqcap \mathsf{U}_1^Y] \triangleq [lcs(\mathsf{L}_2^X, \mathsf{L}_2^Y), \mathsf{U}_2^X \sqcap \mathsf{U}_2^Y] \\ X' : [\mathsf{L}_1^X, \mathsf{U'}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U'}_2^X], Y' : [\mathsf{L}_1^Y, \mathsf{U'}_1^Y] \triangleq [\mathsf{L}_2^Y, \mathsf{U'}_2^Y] \end{array}} \ \{X \mapsto Z \sqcap X', Y \mapsto Z \sqcap Y'\}$$

Where

(i) $lcs(\mathsf{L}_1^X, \mathsf{L}_1^Y) \sqsubseteq \mathsf{U}_1^X \sqcap \mathsf{U}_1^Y$, $lcs(\mathsf{L}_2^X, \mathsf{L}_2^Y) \sqsubseteq \mathsf{U}_2^X \sqcap \mathsf{U}_2^Y$; for no conjunct $C \neq \top$ of $\mathsf{U'}_i^X$ we have $lcs(\mathsf{L}_1^X, \mathsf{L}_1^Y) \sqsubseteq C$; for no conjunct $D \neq \top$ of $\mathsf{U'}_i^Y$ we have $lcs(\mathsf{L}_2^X, \mathsf{L}_2^Y) \sqsubseteq D$;

(ii) $X \sqcap Y$ is not equivalent to a subconcept of G;

(iii) in the **(ssm)** rule for $\mathbf{d}$, $\mathbf{e}$ nodes of G such that $X \in l(\mathbf{d})$ and $Y \in l(\mathbf{e})$ we have $\mathbf{V}_G(\mathbf{d}) \cap \mathbf{V}_G(\mathbf{e}) = \emptyset$.

Figure 1: Minimisation rules

AUTs for short, which are tuples of the form

$$X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X],$$

where X is a concept variable and L_1^X, U_1^X, L_2^X and U_2^X are $\mathcal{EL}$ concepts such that $\mathsf{L}_i^X \sqsubseteq \mathsf{U}_i^X$ for $i = 1, 2$. Intuitively, every substitution that replaces a variable X in G with a concept C 'between the lower and upper bounds for X', that is, such that $\mathsf{L}_i^X \sqsubseteq C \sqsubseteq \mathsf{U}_i^X$, for $i = 1, 2$ is a witness to G being an exact *lgg* w.r.t. $\mathfrak{F}$. We then use this characterisation to demonstrate that every exact *lgg* w.r.t. $\mathfrak{F}$ can be obtained with a substitution from the most general exact generalisation w.r.t. $\mathfrak{F}$, in which every node of G_{sk} contains a unique variable. Then a complete set of exact *lgg*s with skeleton G_{sk} for concepts C_1, C_2 can be computed by minimising the set of all exact *lgg*s of C_1, C_2 w.r.t. framework $(G_{\mathsf{sk}}, \varphi_1, \varphi_2)$, for all possible choices of homomorphisms $\varphi_i : G_{\mathsf{sk}} \to C_i$.

Let C and D be $\mathcal{EL}$ concepts, φ be a variable ignoring homomorphism from D to C and $\mathbf{d}$ be a node of D. The *difference at node* $\mathbf{d}$ *w.r.t* φ between concepts C and D is the concept

$$C \ominus_{\varphi}^{\mathbf{d}} D := \bigsqcap_{A \in l(\varphi(\mathbf{d})) \setminus l(\mathbf{d})} A \sqcap \bigsqcap_{i=1}^{n} \exists s_i.C_i,$$

where $\{\mathbf{c}_1, \ldots, \mathbf{c}_n\}$, $n \geq 0$, is the set of all immediate successors of $\varphi(\mathbf{d})$ in C such that for all $1 \leq i \leq n$,

- $s_i = l(\varphi(\mathbf{d}), \mathbf{c}_i)$,
- $\mathbf{c}_i = \mathbf{root}(C_i)$, and
- there exists no node $\mathbf{d}' \in D$ such that $l(\mathbf{d}, \mathbf{d}') = s_i$ and $\varphi(\mathbf{d}') = \mathbf{c}_i$,

For example, for $C = \exists r.(A \sqcap B) \sqcap \exists r.(A \sqcap B') \sqcap \exists s.A$, $D = \exists r.(A \sqcap Y)$, let $\mathbf{d} = \mathbf{root}(D)$, $\mathbf{c} = \mathbf{root}(C)$, φ be the variable ignoring homomorphism from D to C that maps the r-successor of $\mathbf{d}$ (denoted $\mathbf{d}_r$) into the first r-successor of $\mathbf{c}$ (denoted $\mathbf{c}_r^1$), i.e., $\varphi(\mathbf{d}) = \mathbf{c}$ and $\varphi(\mathbf{d}_r) = \mathbf{c}_r^1$. Then we have $l(\mathbf{d}) = l(\mathbf{c}) = \emptyset$, $l(\mathbf{d}_r) = \{A, Y\}$, $l(\mathbf{c}_r^1) = \{A, B\}$, and, thus, $C \ominus_{\varphi}^{\mathbf{d}} D = \exists r.(A \sqcap B') \sqcap \exists s.A$ and $C \ominus_{\varphi}^{\mathbf{d}_r} D = B$.

Let X be a concept variable, C be a concept and $\mathbf{d}$ be a node of C. We denote by $\mathbf{N}_C(X)$ the set of all nodes of C with X in their label, and by $\mathbf{V}_C(\mathbf{d})$ the set of all variables in the label of $\mathbf{d}$. Let $\mathfrak{F} = (G_{\mathsf{sk}}, \varphi_1, \varphi_2)$ be a generalisation framework for concepts C_1 and C_2 and G be a concept with $\mathsf{skel}(G) = G_{\mathsf{sk}}$. We say that a set of AUTs S is *compatible* with G, C_1, C_2 w.r.t. $\mathfrak{F}$ iff the following conditions hold:

(c1) For every variable X of G, the set S contains exactly one AUT $X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X]$.

(c2) For every AUT $X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X] \in S$, we have $\mathsf{L}_i^X \equiv lcs(\varphi_i(\mathbf{N}_G(X)))$, $i = 1, 2$;

(c3) For every node $\mathbf{d}$ of G_{sk}, we have

$$\bigsqcap_{X \in \mathbf{V}_G(\mathbf{d})} \mathsf{U}_i^X \sqsubseteq (C_i \ominus_{\varphi_i}^{\mathbf{d}} G_{\mathsf{sk}}), \ i = 1, 2.$$

When C_1, C_2 and $\mathfrak{F}$ are clear from the context, we simply talk about S being compatible with G.

The following lemma is proved by induction on the role depth of C_1 and C_2.

Lemma 4 *Let C_1, C_2 be concepts, $\mathfrak{F} = (G_{\mathsf{sk}}, \varphi_1, \varphi_2)$ be a generalisation framework, and G be a concept such that $\mathsf{skel}(G) = G_{\mathsf{sk}}$. Let S be a set of AUTs such that for every variable X of G, it contains exactly one AUT*

$X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X]$. *Assume that the substitutions τ_i, for $i = 1, 2$, are defined as follows:*

$$\tau_i := \{X \mapsto \mathsf{U}_i^X \mid X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X] \in S\}.$$

Then S is compatible with G, C_1, C_2 w.r.t. $\mathfrak{F}$ iff G is an exact generalisation of C_1, C_2 w.r.t. $\mathfrak{F}$ witnessed by τ_1, τ_2.

Notice that for any substitutions σ, σ' and concept C such that $\sigma(X) \sqsubseteq \sigma'(X)$ for every X occurring in C we have $\sigma(C) \sqsubseteq \sigma'(C)$. We use this fact to prove the following.

Corollary 5 *If τ_i are such that for every variable of G we have $\mathsf{L}_i^X \sqsubseteq \tau_i(X) \sqsubseteq \mathsf{U}_i^X$, for $i = 1, 2$, then $\tau(G) \equiv C_i$.*

Given concepts C_1 and C_2 and a generalisation framework $\mathfrak{F} = (G_{\mathsf{sk}}, \varphi_1, \varphi_2)$ we construct

- the concept $G_{\mathfrak{F}}$ by adding a fresh variable to the label of its every node, and
- the set of AUTs $S_{\mathfrak{F}}$, which for every variable X of $G_{\mathfrak{F}}$ contains the AUT $X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X]$, where $\mathsf{L}_i^X = \varphi_i(\mathbf{d})$ and $\mathsf{U}_i^X = C_i \ominus_{\varphi_i}^{\mathbf{d}} G$, where $\mathbf{d}$ is the unique node of $G_{\mathfrak{F}}$ containing variable X.

It should be obvious that $G_{\mathfrak{F}}$ is an exact generalisation of C_1, C_2 w.r.t. $\mathfrak{F}$ and $S_{\mathfrak{F}}$ is compatible with $G_{\mathfrak{F}}, C_1, C_2$ w.r.t. $\mathfrak{F}$ (in fact $G_{\mathfrak{F}}$ is a *most* general exact generalisation of C_1, C_2 w.r.t. $\mathfrak{F}$).

We say that a concept G' and a set of AUTs S' is obtained from a concept G and a set of AUTs S by an application of a *minimisation rule* α with a *side substitution* σ, given in Figure 1, in symbols $(G, S) \rightsquigarrow_{\alpha}^{\sigma} (G', S')$, if α is of the form

$$\frac{S_1}{S_2}\ \sigma,$$

and if (up to variable renaming) S can be represented as $S = S_1 \uplus S_1'$ and $S' = S_2 \uplus S_1'$, where $\uplus$ denotes the disjoint union, and $G' = \sigma(G)$. We write $(G, S) \rightsquigarrow (G', S')$ if $S \rightsquigarrow_{\alpha}^{\sigma} S'$ for some α and σ. We denote by $\rightsquigarrow^*$ the reflexive transitive closure of $\rightsquigarrow$.

Example 6 *Consider concepts*

$$C_1 = \exists r.(A \sqcap B) \sqcap \exists s.(B \sqcap C) \sqcap \exists t.(A \sqcap C) \text{ and}$$
$$C_2 = \exists r.(A' \sqcap B') \sqcap \exists s.(B' \sqcap C') \sqcap \exists t.(A' \sqcap C').$$

Let $G_{\mathsf{sk}} = \exists r.\top \sqcap \exists s.\top \sqcap \exists t.\top$. Notice that homomorphisms $\varphi_i : G_{\mathsf{sk}} \to C_i$, and hence generalisation framework $\mathfrak{F} = (G_{\mathsf{sk}}, \phi_1, \phi_2)$, are uniquely determined.

Then $G_{\mathfrak{F}} = \exists r.X \sqcap \exists s.Y \sqcap \exists t.Z \sqcap W$ and $S_{\mathfrak{F}}$ is the following set of AUTs.

$$\begin{aligned} X &: [A \sqcap B, A \sqcap B] \triangleq [A' \sqcap B', A' \sqcap B'] \\ Y &: [B \sqcap C, B \sqcap C] \triangleq [B' \sqcap C', B' \sqcap C'] \\ Z &: [A \sqcap C, A \sqcap C] \triangleq [A' \sqcap C', A' \sqcap C'] \\ W &: [C_1, \top] \triangleq [C_2, \top]. \end{aligned}$$

*Notice that $lcs(A \sqcap B, B \sqcap C)$ is B, and conjunctions $A \sqcap B$ and $B \sqcap C$ can be represented as $\mathsf{U}_1^X \sqcap \mathsf{U'}_1^X$ and $\mathsf{U}_1^Y \sqcap \mathsf{U'}_1^Y$, respectively, where $\mathsf{U}_1^X = \mathsf{U}_1^Y = B$, $\mathsf{U'}_1^X = A$ and $\mathsf{U'}_1^Y = C$. Thus, the **(ssm)** rule is applicable to X and Y with the side substitution $\{X \mapsto X_1 \sqcap V_1, Y \mapsto Y_1 \sqcap V_1\}$ and the set of AUTs*

$$\begin{aligned} X_1 &: [A \sqcap B, A] \triangleq [A' \sqcap B', A'] \\ Y_1 &: [B \sqcap C, C] \triangleq [B' \sqcap C', C'] \\ V_1 &: [B, B] \triangleq [B', B']. \end{aligned}$$

*Similarly, the **(sm)** rule is applicable to X_1 and Z with the side substitution $\{X_1 \mapsto V_2, Z \mapsto Z_1 \sqcap V_2\}$ and the set of AUTs*

$$V_2 : [A, A] \triangleq [A', A'],\ Z_1 : [A \sqcap C, C] \triangleq [A' \sqcap C', C']$$

*Finally, the **(m)** rule applies to Y_1 and Z_1 giving $\{Z_1 \mapsto V_3, Y_1 \mapsto V_3\}$ and*

$$V_3 : [C, C] \triangleq [C', C'].$$

Putting it all together an application of the substitution

$$\{X \mapsto V_2 \sqcap V_1,\ Y \mapsto V_3 \sqcap V_1,\ Z \mapsto V_3 \sqcap V_2\}$$

to G produces a generalisation

$$G' = \exists r.(V_1 \sqcap V_2) \sqcap \exists s.(V_3 \sqcap V_1) \sqcap \exists t.(V_3 \sqcap V_2) \sqcap W.$$

It can be seen that the lgg of C_1 and C_2 w.r.t. $\mathfrak{F}$

$$G = \exists r.(V_1 \sqcap V_2) \sqcap \exists s.(V_3 \sqcap V_1) \sqcap \exists t.(V_3 \sqcap V_2)$$

can be obtained from G' by eliminating variable W with a substitution $W \mapsto \top$.

In what follows we prove that every generalisation can be obtained by applying minimisation rules to $(G_{\mathfrak{F}}, S_{\mathfrak{F}})$ and then simplifying the result, that is, that our procedure is complete. We illustrate the main ideas of the completeness proof here and defer the technical details to the full version.

The outline of our approach is as follows. Given a generalisation G and a compatible set of AUTs S we construct a generalisation G' and a set of AUTs S' such that (G, S) can be obtained from (G', S') by an application of one of the minimisation rules and (G', S') is in some sense closer to $(G_{\mathfrak{F}}, S_{\mathfrak{F}})$. By inductive reasoning, we end up with a sequence of rule applications such that (G, S) is obtained starting from some (G_0, S_0) such that (G_0, S_0) cannot be obtained by any rule application. We then apply the same sequence of rules to $(G_{\mathfrak{F}}, S_{\mathfrak{F}})$ aiming to produce (G, S).

Notice that not every syntactic form of every generalisation can be obtained by applying minimisation rules to $(F_{\mathfrak{F}}, S_{\mathfrak{F}})$, as the following example demonstrates.

Example 7 *Consider $C_1 = A_1 \sqcap A_2$, $C_2 = B_1 \sqcap B_2$ and $G = X_1 \sqcap X_2$. Then $G_{\mathsf{sk}} = \top$ and the homomorphisms φ_1, φ_2 are trivial. Notice that the set of AUTs S defined as*

$$\{X_i : [A_1 \sqcap A_2, A_i] \triangleq [B_1 \sqcap B_2, B_i] \mid 1 \leq i \leq 2\}$$

is compatible with G, C_1, C_2 w.r.t. $\mathfrak{F} = (G_{\mathsf{sk}}, \varphi_1, \varphi_2)$, yet (G, S) cannot be obtained by applying minimisation rules to $G_{\mathfrak{F}} = X$ and $S_{\mathfrak{F}} = \{X : [A_1 \sqcap A_2, A_1 \sqcap A_2] \triangleq [B_1 \sqcap B_2, B_1 \sqcap B_2]\}$.

To address this issue, suppose that S is a set of AUTs compatible with the concept G w.r.t. some generalisation framework $\mathfrak{F}$ and let $\mathcal{V}$ be a set of variables of G such that every $\{X, Y\} \subseteq \mathcal{V}$ we have $\mathbf{N}_G(X) = \mathbf{N}_G(Y)$. Notice that since all variables of $\mathcal{V}$ only occur in the same nodes of G, by definition of compatibility, we have $\mathsf{L}_i^X \equiv \mathsf{L}_i^Y$, for $i = 1, 2$.

We say that G' and S' are obtained by *reducing repeated variables $\mathcal{V}$ into Z*, in symbols $(G, S)^{\mathcal{V}} \rightrightarrows_Z (G', S')$ if

- G' is obtained from G by replacing in the label of every node $\mathbf{d}$ all $X \in \mathcal{V}$ with Z, and
- S' is obtained from the set S by replacing the AUTs $X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X]$, for each $X \in \mathcal{V}$, with $Z : [\mathsf{L}_1, \bigsqcap_{X \in \mathcal{V}} \mathsf{U}_1^X] \triangleq [\mathsf{L}_2, \bigsqcap_{X \in \mathcal{V}} \mathsf{U}_2^X]$, where $\mathsf{L}_i \equiv \mathsf{L}_i^X$, for $i = 1, 2$ and $X \in \mathcal{V}$.

We write $(G, S) \rightrightarrows (G', S')$ if $(G, S)^{\mathcal{V}} \rightrightarrows_Z (G', S')$ for some $\mathcal{V}$ and Z. The relation $\rightrightarrows^*$ is the reflexive transitive closure of $\rightrightarrows$. It should be clear that G' is equi-general to G and S' is compatible with G' w.r.t. $\mathfrak{F}$.

Another problem is that backward applications of minimisation rules starting with (G, S) not always result in exactly $(G_{\mathfrak{F}}, S_{\mathfrak{F}})$.

Example 8 *Let $C_1 = \exists r.A \sqcap \exists r.B$, $C_2 = \exists r.(A \sqcap B)$ and $G_{\mathsf{sk}} = \exists r.A \sqcap \exists r.B$. The homomorphisms $\varphi_i : G_{\mathsf{sk}} \to C_i$, for $i = 1, 2$, are uniquely determined by the shape of G_{sk} and C_1, C_2.*

Consider a concept $G = \exists r.(A \sqcap X) \sqcap \exists r.(B \sqcap X)$. It is easy to see that G is an exact generalisation of C_1, C_2 w.r.t. $\mathfrak{F} = (G_{\mathsf{sk}}, \varphi_1, \varphi_2)$ and the following set of AUTs S is compatible with G w.r.t. φ_1 and φ_2:

$$S := \{X : [\top, \top] \triangleq [A \sqcap B, A \sqcap B]\}.$$

Then $(G_1, S_1) \rightsquigarrow^{\sigma}_{\textit{(m)}} (G, S)$, where $G_1 = \exists r.(A \sqcap X_1) \sqcap \exists r.(B \sqcap X_2)$ and

$$\begin{aligned} S_1 := \{&X_1 : [A, \top] \triangleq [A \sqcap B, A \sqcap B], \\ &X_2 : [B, \top] \triangleq [A \sqcap B, A \sqcap B]\}. \end{aligned}$$

Notice that S_1 differs from

$$\begin{aligned} S_{\mathfrak{F}} := \{&X_1 : [A, \top] \triangleq [A \sqcap B, B], \\ &X_2 : [B, \top] \triangleq [A \sqcap B, A]\} \end{aligned}$$

However $G_{\mathfrak{F}} = G_1$ and $(G_{\mathfrak{F}}, S_{\mathfrak{F}}) \rightsquigarrow^* (G, S)$.

For two sets of AUTs S and S', compatible with G w.r.t. $\mathfrak{F}$, S is *weaker* than S', in symbols $(G, S') \gg (G, S)$, if for every variable X that occurs in G and the corresponding AUTs $X : [\mathsf{L}_1, \mathsf{U}_1] \triangleq [\mathsf{L}_2, \mathsf{U}_2] \in S$ and $X : [\mathsf{L}'_1, \mathsf{U}'_1] \triangleq [\mathsf{L}'_2, \mathsf{U}'_2] \in S'$ we have $\mathsf{L}_i \equiv \mathsf{L}'_i$ and $\mathsf{U}'_i \sqsubseteq \mathsf{U}_i$, for $i = 1, 2$. In Example 8 above, we have $(G_{\mathfrak{F}}, S_1) \gg (G_{\mathfrak{F}}, S_{\mathfrak{F}})$.

The following lemma is proved by induction on the number of variables that have more than one occurrence in the generalisation.

Lemma 9 *Let C_1, C_2 and G be $\mathcal{EL}$ concepts such that the label of every node of G contains a variable. Let S be a set of AUTs compatible with G w.r.t. some generalisation framework $\mathfrak{F}$. Then there exist a concept G^* and sets of AUTs S^* and S' such that $(G_{\mathfrak{F}}, S_{\mathfrak{F}}) \rightsquigarrow^* (G^*, S^*)$, and $(G, S) \rightrightarrows^* (G^*, S') \gg (G^*, S^*)$.*

Lemma 9 requires every node of G to contain a variable, which does not always hold. We say that a concept G and a set of AUTs S are obtained by *variable elimination* from (G', S') if $G = \{X \mapsto \top\}(G')$ and S is obtained from S' by removing the AUT $X : [\mathsf{L}_1^X, \mathsf{U}_1^X] \triangleq [\mathsf{L}_2^X, \mathsf{U}_2^X]$ such that $\mathsf{U}_1^X \sqcap \mathsf{U}_2^X \equiv \top$. Let $\mathsf{elim}_S(G')$ denote the result of exhaustive variable elimination. Then the following theorem immediately follows from Lemma 9.

Theorem 10 (Completeness) *Let C_1, C_2 and G be $\mathcal{EL}$ concepts and $\mathfrak{F}$ a generalisation framework. If G is an exact generalisation of C_1, C_2 w.r.t. $\mathfrak{F}$ then $(G_{\mathfrak{F}}, S_{\mathfrak{F}}) \rightsquigarrow^* (G', S)$, for some set of AUTs S and a generalisation G' of C_1, C_2 such that $\mathsf{elim}_S(G')$ is less general than G.*

The following statement is an immediate consequence of the shape of the minimisation rules.

Theorem 11 (Soundness) *Let C_1, C_2 be $\mathcal{EL}$ concepts and $\mathfrak{F}$ be a generalisation framework. Let a concept G and a set of AUTs S be such that $(G_{\mathfrak{F}}, S_{\mathfrak{F}}) \rightsquigarrow^* (G, S)$. Then $\mathsf{elim}_S(G)$ is an exact generalisation of C_1, C_2 w.r.t. $\mathfrak{F}$.*

Notice that every application of the **(ssm)** rule introduces a shared variable Z in the nodes where X and Y occur; every application of the **(sm)** rule introduces $Z \sqcap X'$ into the nodes where X occurs. We use these properties to prove inductively that every computation terminates.

Theorem 12 (Termination) *Let C_1, C_2 be $\mathcal{EL}$ concepts and $\mathfrak{F}$ a generalisation framework. Then any sequence $(G_{\mathfrak{F}}, S_{\mathfrak{F}}) = (G_0, S_0)$, (G_1, S_1),..., (G_m, S_m) such that $(G_i, S_i) \rightsquigarrow (G_{i+1}, S_{i+1})$ and no minimisation rule applies to (G_m, S_m) contains polynomially many elements.*

Finally we notice that every (G_i, S_i) can be computed in (non-deterministic) polynomial time. Indeed, even though minimisation rules are formulated in such a way that a repeated computation of *lcs* is need, which can be exponential in the size of input (Küsters 2001), we only use *lcs* to check that U_i^X and U_i^Y are such that $lcs(\mathsf{L}_i^X, \mathsf{L}_i^Y) \sqsubseteq \mathsf{U}_i^X \sqcap \mathsf{U}_i^Y$, which is equivalent to $\mathsf{L}_i^X \sqsubseteq \mathsf{U}_i^X \sqcap \mathsf{U}_i^Y$ and $\mathsf{L}_i^Y \sqsubseteq \mathsf{U}_i^X \sqcap \mathsf{U}_i^Y$. The side conditions in the minimisation rules ensure that the length of every sequence of rule applications starting from $(G_{\mathfrak{F}}, S\mathfrak{F})$ is polynomial.

Complexity of fixed skeleton generalisations. We use the machinery developed to analyse exact generalisations w.r.t. generalisation frameworks to establish complexity bounds for computing fixed skeleton generalisations. It follows from the NP-completeness of matching modulo equivalence (Baader and Küsters 2006) that the computational problem of checking if G is a generalisation of C_1 and C_2 is NP-complete. For least general generalisations, we rely on the fact that if G is a fixed skeleton *lgg* of concepts C_1, C_2 with skeleton G_{sk} then there exist homomorphisms

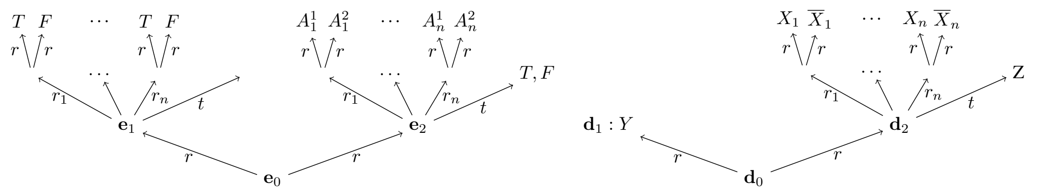

Figure 2: Concepts C'_1 (left) and G' (right) for the proof of Theorem 13.

$\varphi_i : G_{\mathsf{sk}} \to C_i$, for $i = 1, 2$, such that G is an *lgg* of C_1 and C_2 w.r.t. $(G_{\mathsf{sk}}, \varphi_1, \varphi_2)$.

Theorem 13 *Let C_1, C_2, G_{sk} and G be $\mathcal{EL}$ concepts. Then the problem of checking if G is a least general exact generalisation of C_1 and C_2 with skeleton G_{sk} is coNP-hard.*

Proof. We proceed by reduction from unsatisfiability. Given a 3CNF formula

$$\phi = (l_1^{(1)} \vee l_2^{(1)} \vee l_3^{(1)}) \wedge \cdots \wedge (l_1^{(m)} \vee l_2^{(m)} \vee l_3^{(m)}),$$

where, for $i \in \{1, 2, 3\}$ and $j \in \{1, \ldots, m\}$, every $l_i^{(j)}$ is a literal from the set $\{p_1, \ldots, p_n, \neg p_1, \ldots, \neg p_n\}$, we present concepts C'_1, C'_2 over variables $\{X_1, \ldots, X_n, \overline{X}_1, \ldots, \overline{X}_n\}$ and a ground concept G', and analyse conditions under which G' is an exact *lgg* of C'_1, C'_2 w.r.t. $(G'_{\mathsf{sk}}, \varphi_1, \varphi_2)$, for some homomorphisms $\varphi_i : G'_{\mathsf{sk}} \to C'_i$, $i = 1, 2$, $G'_{\mathsf{sk}} = \mathsf{skel}(G')$, and discuss implications of these considerations on the value that variables X_i and $\overline{X}_i$, which encode propositional interpretations, take under witness substitutions τ_i. In the second step, we extend C'_1, C'_2, G' into concepts C_1^ϕ, C_2^ϕ and G^ϕ in such a way that G^ϕ is an exact least general generalisation of C_1^ϕ and C_2^ϕ iff ϕ is unsatisfiable.

Consider concepts C'_1, C'_2 and G'_1 defined as follows:

$$\begin{aligned} C'_1 := \exists r.(&\exists r_1.(\exists r.T \sqcap \exists r.F) \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.T \sqcap \exists r.F) \sqcap \exists t.\top) \sqcap \\ \exists r.(&\exists r_1.(\exists r.A_1^1 \sqcap \exists r.A_1^2) \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.A_n^1 \sqcap \exists r.A_n^2) \sqcap \exists t.(T \sqcap F)), \end{aligned}$$

$$\begin{aligned} C'_2 := \exists r.(&\exists r_1.(\exists r.T' \sqcap \exists r.F') \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.T' \sqcap \exists r.F') \sqcap \exists t.\top) \sqcap \\ \exists r.(&\exists r_1.(\exists r.{A'}_1^1 \sqcap \exists r.{A'}_1^2) \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.{A'}_n^1 \sqcap \exists r.{A'}_n^2) \sqcap \exists t.(T' \sqcap F')). \end{aligned}$$

and

$$\begin{aligned} G' := \exists r.Y \sqcap \exists r.(&\exists r_1.(\exists r.X_1 \sqcap \exists r.\overline{X}_1) \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.X_n \sqcap \exists r.\overline{X}_n) \sqcap \exists t.Z). \end{aligned}$$

We illustrate concepts C'_1 and G' in Figure 2. Let $G'_{\mathsf{sk}} = \mathsf{skel}(G')$ and let sets of homomorphisms $\Phi_1^{\shortparallel}$ and $\Phi_1^{\times}$ from G'_{sk} to C'_1 be defined as $\Phi_1^{\shortparallel} = \{\varphi_1^{\shortparallel} \mid \varphi_1^{\shortparallel}(\mathbf{d}_1) = \mathbf{e}_1, \varphi_1^{\shortparallel}(\mathbf{d}_2) = \mathbf{e}_2\}$ and $\Phi_1^{\times} = \{\varphi_1^{\times} \mid \varphi_1^{\times}(\mathbf{d}_1) = \mathbf{e}_2, \varphi_1^{\times}(\mathbf{d}_2) = \mathbf{e}_1\}$. Notice that every homomorphism $\varphi : G'_{\mathsf{sk}} \to C'_1$ belongs to either $\Phi_1^{\shortparallel}$ or $\Phi_1^{\times}$. Sets of homomorphisms $\Phi_2^{\shortparallel}$ and $\Phi_2^{\times}$ from G'_{sk} to C'_2 are defined similarly.

It should be clear that for any $\varphi_1^{\shortparallel} \in \Phi_1^{\shortparallel}$ and $\varphi_2^{\shortparallel} \in \Phi_2^{\shortparallel}$, the concept G' is an exact generalisation of C'_1 and C'_2 w.r.t. $\mathfrak{F}^{\shortparallel} = (G'_{\mathsf{sk}}, \varphi_1^{\shortparallel}, \varphi_2^{\shortparallel})$ with witness substitutions $\sigma_1^{\shortparallel}$ and $\sigma_2^{\shortparallel}$ such that

$$\begin{aligned} \sigma_1^{\shortparallel}(Y) \equiv \exists r_1.(&\exists r.T \sqcap \exists r.F) \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.T \sqcap \exists r.F) \sqcap \exists t.\top, \\ \sigma_1^{\shortparallel}(Z) \equiv T \sqcap F \text{ and } &\{\sigma_1^{\shortparallel}(X_i), \sigma_1^{\shortparallel}(\overline{X}_i)\} \equiv \{A_i^1, A_i^2\}, \end{aligned}$$

for every i, $1 \leq i \leq n$. For $\sigma_2^{\shortparallel}$ the condition is similar. By considering cases it can be proved that G' is $lgg(C'_1, C'_2)$ w.r.t. $\mathfrak{F}^{\shortparallel}$.

Concept G' is also an exact generalisation of C'_1 and C'_2 w.r.t. $\mathfrak{F} = (G'_{\mathsf{sk}}, \varphi_1^{\times}, \varphi_2)$, for any $\varphi_1^{\times} \in \Phi_1^{\times}$ and $\varphi_2 \in \Phi_2^{\shortparallel} \cup \Phi_2^{\times}$, with witness substitutions $\sigma_1^{\times}$

$$\begin{aligned} \sigma_1^{\times}(Y) \equiv \exists r_1.(&\exists r.A_1^1 \sqcap \exists r.A_1^2) \sqcap \cdots \sqcap \\ &\exists r_n.(\exists r.A_n^1 \sqcap \exists r.A_n^2) \sqcap \exists t.(T \sqcap F), \\ \sigma_1^{\times}(Z) \equiv \top \text{ and } &\{\sigma_1^{\times}(X_i), \sigma_1^{\times}(\overline{X}_i)\} \equiv \{T, F\}, \end{aligned}$$

for every i, $1 \leq i \leq n$. It is easy to see that, since for some $i \neq j$ we have $\sigma_1^{\times}(W_i) = \sigma_1^{\times}(W_j)$ where W_i is one of $X_i, \overline{X}_i$ and W_j is one of $X_j, \overline{X}_j$, the **(m)** or **(sm)** minimisation rule always applies to the corresponding set of AUTs and so G' is not an $lgg(C'_1, C'_2)$ w.r.t. $\mathfrak{F}$. For $\mathfrak{F} = (G'_{\mathsf{sk}}, \varphi_1, \varphi_2^{\times})$ the reasoning is similar.

Thus, G' is an exact least general generalisation of C'_1 and C'_2 w.r.t. some generalisation framework $\mathfrak{F} = (G'_{\mathsf{sk}}, \varphi_1, \varphi_2)$ iff $\varphi_1 \in \Phi_1^{\shortparallel}$ and $\varphi_2 \in \Phi_2^{\shortparallel}$.

Consider now

$$\begin{aligned} C''_1 &= \exists t_F.F \sqcap \textstyle\bigsqcap_{i=1}^n (\exists t_i^1.A_i^1 \sqcap \exists t_i^2.A_i^2), \\ C''_2 &= \exists t_F.F' \sqcap \textstyle\bigsqcap_{i=1}^n (\exists t_i^1.{A'}_i^1 \sqcap \exists t_i^2.{A'}_i^2), \text{ and} \\ G'' &= \exists t_F.Y_F \sqcap \textstyle\bigsqcap_{i=1}^n (\exists t_i^1.Y_i^1 \sqcap \exists t_i^2.Y_i^2) \end{aligned}$$

G'' is an exact *lgg* of C''_1 and C''_2 with skeleton $\mathsf{skel}(G'')$ and for every witness substitutions τ_1 and τ_2 we always have

$$\begin{aligned} &\tau_1(Y_F) \equiv F;\ \tau_1(Y_i^1) \equiv A_i^1;\ \tau_1(Y_i^2) \equiv A_i^2, \\ &\tau_2(Y_F) \equiv F';\ \tau_2(Y_i^1) \equiv {A'}_i^1;\ \tau_2(Y_i^2) \equiv {A'}_i^2,\ i = 1, 2. \end{aligned}$$

Finally, for the 3CNF formula

$$\phi = (l_1^{(1)} \vee l_2^{(1)} \vee l_3^{(1)}) \wedge \cdots \wedge (l_1^{(m)} \vee l_2^{(m)} \vee l_3^{(m)})$$

define a translation function $f(p_i) = X_i$ and $f(\neg p_i) = \overline{X}_i$ and concepts G^ϕ, C_1^ϕ,and C_2^ϕ as

$$C_1^\phi := C'_1 \sqcap C''_1 \sqcap \textstyle\bigsqcap_{j=1}^m \exists s_j.(\mathbf{A} \sqcap T \sqcap F), \text{ and}$$

$$C_2^\phi := C_2' \sqcap C_2'' \sqcap \textstyle\prod_{j=1}^{m} \exists s_j.(\mathbf{A}' \sqcap T' \sqcap F'),$$
$$G^\phi := G' \sqcap G'' \sqcap \prod_{j=1}^{m} \exists s_j.\Big(\mathbf{Y} \sqcap Y_F \sqcap f(l_1^{(j)}) \sqcap f(l_2^{(j)}) \sqcap f(l_3^{(j)}) \sqcap Z\Big),$$

where $\mathbf{Y} = Y_1^1 \sqcap Y_1^2 \sqcap \cdots \sqcap Y_n^1 \sqcap Y_n^2$, $\mathbf{A} = A_1^1 \sqcap A_1^2 \sqcap \cdots \sqcap A_n^1 \sqcap A_n^2$, and $\mathbf{A}' = A_1'^1 \sqcap A_1'^2 \sqcap \cdots \sqcap A_n'^1 \sqcap A_n'^2$.

For every witness substitution $\sigma_1^{\shortparallel}$ (resp. $\sigma_2^{\shortparallel}$) we always have $(\sigma_1^{\shortparallel} \cup \tau_1)(G^\phi) \equiv C_1^\phi$ (resp. $(\sigma_2^{\shortparallel} \cup \tau_2)(G^\phi) \equiv C_2^\phi$); for $\sigma_1^{\times}$ we have $(\sigma_1^{\times} \cup \tau_1)(G^\phi) \equiv C_1^\phi$ iff $I \models \phi$, where the interpretation I is defined as $I = \{p_i \mid \sigma_1^{\times}(X_i) = T\}$. Thus, G^ϕ is an exact least general generalisation with a fixed skeleton of concepts C_1^ϕ, C_2^ϕ iff ϕ is unsatisfiable. ❏

Experimental evaluation

We have implemented our anti-unification algorithm as a *Mathematica* package. To reduce the degree of non-determinism, in the **(sm)** and **(ssm)** rules we additionally require $\mathsf{U}_1^X \sqcap \mathsf{U}_1^Y \sqcap \mathsf{U}_2^X \sqcap \mathsf{U}_2^Y \not\equiv \top$, $\mathsf{U}_1'^Y \sqcap \mathsf{U}_2'^Y \not\equiv \top$, and $\mathsf{U}_1'^Y \sqcap \mathsf{U}_2'^Y \not\equiv \top$ and use the following strategy:

- Repeat until no minimisation rule is applicable
  - Apply **(m)** exhaustively (does not generate branching)
  - Apply **(sm)** whenever applicable (causes branching)
  - Apply **(ssm)** whenever applicable (causes branching)
- If no rule is applicable, return $\mathsf{elim}_S(G)$.

While this strategy leads to incompleteness, in many practical cases it gives shorter and more meaningful generalisations. The implementation accepts a generalisation framework as input but it can construct one based on the lcs.

To evaluate our procedure, we have computed generalisations of the right hand sides of concept equalities of the form $A :\equiv C$ of the GALEN-EL ontology (Kazakov and Klinov 2015; Rector et al. 2003). There are 9968 such definitions in the ontology. We experimented with various ways of computing lcs-based skeleton, from taking just the lcs to making it some smaller size concept that subsumes the lcs but still retains the common structure of the input concepts. The statistics reported here is for the latter.

We ran our test on all 9968 selected axioms. To avoid computing trivial or very simple generalisations, we restricted our consideration to cases when generalisations were either ground or had skeleton of depth at least 2.

It took 77.5 hours on Dell Linux Workstation with Intel Xeon E5-2680 v2 CPU and 384 GB RAM to anti-unify 49 675 528 concept pairs and compute 99 529 answers satisfying our selection criteria, among which 750 were ground, 42 258 showed that the anti-unified concepts differed from each other only by concept names (we call them *renaming generalisations*), and 56 521 showed non-renaming differences. To evaluate the scalability of our implementation, we also run our tool on 1 000 randomly picked axioms, which took it 47 minutes.

Our experiments revealed interesting insights into the structure of the ontology; we illustrate some of our findings below. A generalisation G being ground indicates that the input concepts are identical, i.e., some concept descriptions repeat in the ontology: There are axioms of the form $A :\equiv G$ and $B :\equiv G$, which could have been better modelled as $A :\equiv G$ and $B :\equiv A$. Examples of repeated definitions are AbductionOfGlenoHumeralJoin, Anynomous-469, and Anynomous-488, all defined as

$$\mathsf{Abduction} \sqcap \exists \mathsf{actsSpecificallyOn}.\,\mathsf{GlenoHumeralJoint};$$

Anynomous-109 and Anynomous-331 both are defined as

$$\mathsf{Device} \sqcap \exists \mathsf{isSpecificPhysicalMeansOf}.\,(\mathsf{NonDirectInspecting} \sqcap \exists \mathsf{actsSpecificallyOn}.\mathsf{Eye});$$

etc.

The definitions of the concepts AdenomaOfColon and AdrenalPhaeochromocytoma can be obtained from one another by renaming of concept names, and their renaming generalisation is

$$\mathsf{BodyStructure} \sqcap \exists \mathsf{hasUniqueAssociatedProcess}.\,(\mathsf{BenignNeoplasticProcess} \sqcap \exists \mathsf{actsSpecificallyOn}.\,(X_1 \sqcap \exists \mathsf{isSpecificStructuralComponentOf}.\,X_2)),$$

where the differences are only in concept names as the computed AUTs show

$$X_1 : [\ldots, \mathsf{Adenocyte}] \triangleq [\ldots, \mathsf{ChromaffinCell}],$$
$$X_2 : [\ldots, \mathsf{Colon}] \triangleq [\ldots, \mathsf{AdrenalMedulla}].$$

(The lower bounds are omitted for brevity.)

An interesting case is the generalisation of Anonymous-257 and Anonymous-529, which shows that these definitions differ in two places, but in exactly the same way:

$$X_1 \sqcap \exists \mathsf{IsDivisionOf}.\,(X_1 \sqcap \exists \mathsf{isPairedOrUnpaired}.\,\mathsf{atLeastPaired}),$$

where

$$X_1 : [\ldots, \mathsf{GenericBodyStructure}] \triangleq [\ldots, \mathsf{BodyPart}].$$

We call such generalisations *nonlinear*.

An example of a nonlinear and nonrenaming generalisation has been obtained, e.g., for Anonymous-158 and Anonymous-340:

$$\mathsf{absence} \sqcap \exists \mathsf{isExistenceOf}.\,(X_1 \sqcap X_2 \sqcap \exists \mathsf{isConsequenceOf}.\,(\mathsf{GeneralisedProcess} \sqcap X_1 \sqcap X_3 \sqcap \exists \mathsf{LocativeAttribute}.\,\mathsf{OrganicSolidStructure})),$$

where the AUTs are

$$X_1 : [\ldots, \exists \mathsf{hasIntrinsicAbnormalityStatus}.\,\mathsf{normal}] \triangleq [\ldots, \mathsf{GeneralisedProcess}],$$
$$X_2 : [\ldots, \mathsf{GeneralisedSubstance}] \triangleq [\ldots, \top],$$
$$X_3 : [\ldots, \top] \triangleq [\ldots, \exists \mathsf{hasAbnormalityStatus}.\mathsf{nonNormal}].$$

From this generalisation one can see that the definitions of Anonymous-158 and Anonymous-340 are quite similar. As for the differences, the generalisation and the AUT for X_1 show that the definition of Anonymous-158 contains the concept $\exists$hasIntrinsicAbnormalityStatus. normal in two places where the definition of Anonymous-340 has GeneralisedProcess. From the AUT for X_2 we conclude that Anonymous-158 contains GeneralisedSubstance, which is not present in Anonymous-340. Similarly, the AUT for X_3 says that $\exists$hasAbnormalityStatus.nonNormal occurs in Anonymous-340 but not in Anonymous-158.

Conclusions

We have studied the anti-unification problem for the description logic $\mathcal{EL}$, proved that a finite minimal complete set of generalisations for $\mathcal{EL}$ concepts always exists and established complexity bounds for computing them. We presented an anti-unification algorithm for a fixed generalisation framework case and proved that it always terminates with a generalisation of the given concepts and that every *lgg* can be computed by an algorithm run. We evaluated its performance in a case study. Investigating the existence of tractable strategies that guarantee that the output is always an *lgg* and computing *lgg*s in presence of background knowledge (TBoxes) constitutes future work.

Acknowledgements

This work has been partially supported by the UK Engineering and Physical Sciences Research Council, under the project EP/M012646/1, and by the Austrian Science Fund (FWF), under the projects P 24087-N18 and P 28789-N32.

References

Baader, F., and Küsters, R. 2006. Nonstandard inferences in description logics: The story so far. In *Mathematical Problems from Applied Logic I*, volume 4 of *International Mathematical Series*. 1–75.

Baader, F., and Morawska, B. 2010. Unification in the description logic EL. *LMCS* 6(3).

Baader, F.; Calvanese, D.; McGuiness, D.; Nardi, D.; and Patel-Schneider, P. 2003. *The Description Logic Handbook: Theory, implementation and applications*. CUP.

Baader, F.; Brandt, S.; and Lutz, C. 2005. Pushing the $\mathcal{EL}$ envelope. In *IJCAI*, 364–369. Professional Book Center.

Baader, F.; Küsters, R.; and Molitor, R. 1999. Computing least common subsumers in description logics with existential restrictions. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 99*, 96–103. Morgan Kaufmann.

Cohen, W. W.; Borgida, A.; and Hirsh, H. 1992. Computing least common subsumers in description logics. In *Proceedings of the 10th National Conference on Artificial Intelligence.*, 754–760. AAAI Press / The MIT Press.

Domingue, J.; Fensel, D.; and Hendler, J. A., eds. 2011. *Handbook of Semantic Web Technologies*. Springer.

Kazakov, Y., and Klinov, P. 2015. Advancing ELK: not only performance matters. In *Proceedings of the 28th International Workshop on Description Logics*, volume 1350 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Küsters, R. 2001. *Non-Standard Inferences in Description Logics*, volume 2100 of *LNCS*. Springer.

Plotkin, G. D. 1970. A note on inductive generalization. *Machine Intel.* 5(1):153–163.

Pottier, L. 1989. Generalisation de termes en theorie equationnelle. Cas associatif-commutatif. Research Report RR-1056, INRIA Sophia Antipolis.

Rector, A. L.; Rogers, J.; Zanstra, P. E.; and van der Haring, E. J. 2003. OpenGALEN: Open source medical terminology and tools. In *AMIA Annual Symposium Proceedings3*. AMIA.

Reynolds, J. C. 1970. Transformational systems and the algebraic structure of atomic formulas. *Machine Intel.* 5(1):135–151.

W3C OWL Working Group. 2012. OWL 2 Web Ontology Language Document Overview. W3C Recommendation.

Yu, L. 2014. *A Developer's Guide to the Semantic Web*. Springer.