Ralf Küsters
Ralf Molitor

Structural Subsumption and Least Common Subsumers in a Description Logic with Existential and Number Restrictions

Abstract. The least common subsumer (lcs) of a set of concept descriptions is the most specific concept description that subsumes all of the concept descriptions in the given set. By computing the lcs, commonalities between concept descriptions can be made explicit. This is an important inference task useful in several applications, including, for instance, the bottom-up construction of description logic knowledge bases.

Previous work on the lcs has concentrated on description logics that either allow for number restrictions or for existential restrictions. Many applications, however, require to combine these constructors. In this work, we present an algorithm for computing the lcs in the description logic $\mathcal{ALEN}$ which comprises both constructors—number and existential restrictions—as well as concept conjunction, primitive negation, and value restrictions. To prove correctness of our lcs algorithm, we develop a structural characterization of subsumption in $\mathcal{ALEN}$.

Keywords: description logics, least common subsumer, complexity

1. Introduction

Description logics (DLs) [4] denote a family of knowledge representation formalisms that allow to represent the terminological knowledge of an application domain in a structured and well-defined way. The main feature of DLs and DL-based knowledge representation systems are inference services which allow to derive implicit knowledge from the knowledge explicitly stored in the knowledge base. Standard inferences include computing subconcept/superconcept relationships between concept descriptions (*subsumption relationships*) as well as checking whether a given individual belongs to a certain concept (*instance relationships*). These standard inferences have been studied intensively during the last 15 years or so. The boundary between decidability and undecidability as well as (for decidable DLs) the complexity of the standard inferences are now well-investigated [17, 13].

Presented by **Name of Editor**; *Received* ?

Studia Logica **0**: 1–34, 0.

Moreover, there are highly optimized implementations of reasoners for very expressive DLs [23, 19, 24, 31].

Although standard inferences help to structure a knowledge base by automatically detecting inconsistencies and building a concept hierarchy, they provide only little support for actually constructing and maintaining knowledge bases. To overcome this problem, new inferences, so-called *non-standard inferences*, have been introduced and studied (see [26] for an overview). In the present paper, we are concerned with one of the first and most important such inference, the least common subsumer (lcs). We study the lcs in the DL $\mathcal{ALEN}$, which includes the concept constructors concept conjunction, primitive negation as well as number, value, and existential restrictions.

The *least common subsumer (lcs)* of a set of concept descriptions is the most specific concept description (w.r.t. subsumption) within the given DL that subsumes all of the concept descriptions in the set. In other words, the lcs respresents and should make explicit the commonalities between concept descriptions. As an example, consider the concept MOD of a *mother with only daughters* and the concept FMC of a *father with many (at least four) children*, which can be expressed in the DL $\mathcal{ALEN}$ as follows:

$$\begin{aligned} \mathsf{MOD} &= \mathsf{Human} \sqcap \mathsf{Female} \sqcap \exists \mathsf{has\text{-}child}.\top \sqcap \forall \mathsf{has\text{-}child}.(\mathsf{Female} \sqcap \mathsf{Human}) \\ \mathsf{FMC} &= \mathsf{Human} \sqcap \neg \mathsf{Female} \sqcap \forall \mathsf{has\text{-}child}.\mathsf{Human} \sqcap \geq 4\, \mathsf{has\text{-}child} \end{aligned}$$

The lcs of MOD and FMC (in $\mathcal{ALEN}$) can be expressed as the concept description

$$\mathsf{Human} \sqcap \exists \mathsf{has\text{-}child}.\top \sqcap \forall \mathsf{has\text{-}child}.\mathsf{Human}$$

which intuitively corresponds to the concept of parents.

The lcs is motivated by various applications, including similarity-based information retrieval [32, 30] and inductive learning [18, 15, 14, 16]. The one that has mainly driven the work presented here is the bottom-up construction of knowledge bases [5, 7] in the context of chemical process engineering [36, 34, 12]. In a nutshell, in this application the knowledge engineer wishes to define a new concept description—in the process of building and refining the knowledge base bottom-up—by simply choosing more specific concept descriptions that are already stored in the knowledge base and then let the DL system compute the lcs of the chosen concept descriptions as a candidate for the new description to be defined. This candidate is then inspected by the knowledge engineer and, after possible modifications, is added to the knowledge base. For instance, assume that in the above example the concepts MOD and FMC are already stored in the knowledge base and that the knowledge engineer wants to define the concept of parents. Since the engineer knows that MOD and FMC are subconcepts of the concept parents, she can ask the system to compute the lcs of MOD and FMC, and in this case would immediately obtain the desired concept description, which she then adds to the knowledge base.

Note that in a DL that allows for disjunction, the lcs of concept descriptions is simply their disjunction. However, in this case, the lcs is not useful as it does not help to make explicit the properties that the input concepts have in common. Hence, the lcs appears to be useful only in cases where the DL does not allow for disjunction (see also the related work). In the above example, the lcs of MOD and FMC would be the disjunction $\mathsf{MOD} \sqcup \mathsf{FMC}$ of MOD and FMC in case the DL considered allows for concept disjunction. Presenting this concept description to the knowledge engineer does obviously not help her to define a new concept description.

The first lcs algorithms proposed in the literature were applicable to sublanguages of CLASSIC, more precisely, DLs that allow for number restrictions [16, 15] (see also [27]) but not for existential restrictions. Later, the lcs has been investigated for the DL $\mathcal{ALE}$ [7], which allows for existential restrictions but not for number restrictions. However, many applications, such as those in medical informatics and the envisioned Semantic Web (see Part III of [4]) ask for more expressive DLs which include both number and existential restrictions. In fact, also the mentioned application in chemical process engineering requires both constructors. In this application one wants, for example, to be able to describe a reactor with cooling jacket and exactly two inlet valves. Such a concept can be described in the DL $\mathcal{ALEN}$, which contains both exististential and number restrictions, as $\mathsf{Reactor} \sqcap \exists \mathsf{is\text{-}coupled\text{-}to.Cooling\text{-}Jacket} \sqcap (=2\ \mathsf{has\text{-}inlet}) \sqcap \forall \mathsf{has\text{-}inlet.Valve}$.

In this work, we propose an algorithm for computing the lcs of concept descriptions in the DL $\mathcal{ALEN}$. We show that the lcs of concept descriptions in $\mathcal{ALEN}$ always exists, that it may grow exponentially in the size of the input descriptions, and that it can be computed in double exponential time. Similar to previous approaches [16, 5, 7], our lcs algorithm builds on a structural characterization of subsumption.

Typically, such a characterization works in two steps. First, concept descriptions are turned into a structural normal form, which is a finite representation of all facts implicitly represented in the description. Second, the subsumer and the subsumee, given in structural normal form, are compared syntactically. A sound and complete characterization then ensures that the structural normal form indeed represents all implied facts. Now, given that the structural normal form of concept descriptions can be computed effectively, the lcs of concept descriptions can be obtained by first computing their structural normal forms and then extracting the "common facts" represented in these normal forms.

For $\mathcal{ALEN}$, however, computing the structural normal form already requires to (inductively) compute the lcs (see our running example in Section 3). Consequently, the proof of correctness of the lcs algorithm needs to be interleaved with the proof of soundness and completeness of the characterization of subsumption, making the proofs quite involved. In [29], this approach is in fact pursued. However, in an attempt to avoid these interleaving proofs, the author made unproved assumptions concerning the existence and other properties of the lcs. Moreover, the lcs algorithm presented there is incorrect in that the computed concept description not necessarily subsumes the input descriptions (see Section 3).

In this work, we devise a more relaxed notion of a structural normal form which does not involve the lcs computation and therefore allows to decouple the characterization of subsumption from computing the lcs. Instead of a single $\mathcal{ALEN}$-concept description, our normal form consists of a set of $\mathcal{ALEN}$-concept descriptions, where some of the implicit facts are not made explicit.

Related work. We already mentioned that the first works on computing the lcs, which were mainly motivated by inductive learning [18, 14], concentrated on the DL Classic [16, 15] (see also [27, 26]). Motivated by the bottom-up construction of knowledge bases, Baader et al. [7] were the first to study the lcs for a DL that contains existential restrictions. As explained, the main contribution of the present work, which is an extended version of [28], is an lcs algorithm for a DL that combines existential and number restrictions.

In all the works on lcs mentioned so far, including the present one, background knowledge bases are not considered. This is a common assumption as in many cases knowledge bases can be eliminated. In particular, this is the case for so-called acyclic knowledge bases where definitions of concepts do not contain self-references, and therefore, can be eliminated by a procedure called unfolding (see, e.g., [33]). However, for cyclic knowledge bases [33, 1], whose semantics is often defined by fixed-points, this is not possible [33]. These knowledge bases have to be explicitly taken into account both in standard inferences, such as subsumption and consistency checking, and in non-standard inferences, such as the lcs computation as studied here. The lcs w.r.t. cyclic knowlegde bases has first been investigated in [5] for the DL $\mathcal{ALN}$ and more recently in [3, 2] for $\mathcal{EL}$.

While modern DL systems, such as FaCT [22] and Racer [19], provide their users with the mentioned standard inference services for very expressive DLs and DL-based ontology languages, such as SHIQ and OWL [25], algorithms for solving non-standard inferences are still restricted to relatively unexpressive DLs. The main goal of the present work is to close this gap in case of the lcs by providing algorithms for more expressive DLs. There are two other approches to close this gap. First, one can try to approximate an expressive DL by a less expressive DL and then apply non-standard inferences to the latter DL. First results in this direction have been obtained in [11] and [10] where $\mathcal{ALC}$- and $\mathcal{ALCN}$-concept descriptions have been approximated by concept descriptions in DLs without concept disjunction. Second, one considers non-standard inferences w.r.t. a background knowledge base and distinguishes between a target language in which the non-standard inferences are performed and a typically much more expressive DL used to formulate the background knowledge base. First results for the lcs modulo background terminologies have been obtained in [8]. The two approaches also allow the user of a DL system to benefit from non-standard inferences in case the inferences do not make sense for the full DL underlying the DL system. For example, as already explained above, if the DL system allows for disjunction, then lcs computation does not make sense. By approximating concepts by concepts which do not contain concept disjunction or by restricting disjunction to occur only in the background knowledge base, the lcs operation can still be beneficial.

Outline of the paper. In Section 2, we formally introduce the DL $\mathcal{ALEN}$ and the lcs operation. Section 3 contains a running example and a discussion of the main difficulties that occur when computing the lcs. In subsequent sections, this example is used to illustrate the notions introduced. Section 4 then covers the characterization of subsumption and Section 5 the lcs algorithm, with the size of the lcs and the complexity of computing the lcs studied in Section 6. We conclude in Section 7.

2. Preliminaries

We introduce the DL $\mathcal{ALEN}$ and sublanguages thereof as well as the inference problems subsumption and least common subsumer.

Concept descriptions are defined inductively starting with a set N_C of

Constructor name	Syntax	$\mathcal{ALEN}$	$\mathcal{ALE}$	$\mathcal{EL}$	$\mathcal{ALN}$
top-concept	$\top$	x	x	x	
bottom-concept	$\bot$	x	x		x
concept name $P \in N_C$	P	x	x	x	x
primitive negation, $P \in N_C$	$\neg P$	x	x		x
conjunction	$C \sqcap D$	x	x	x	x
existential restrictions, $r \in N_R$	$\exists r.C$	x	x	x	
value restrictions, $r \in N_R$	$\forall r.C$	x	x		
number restrictions	$(\geq nr)$	x			x
$r \in N_R$, $n \in \mathbb{N}$	$(\leq nr)$	x			x

Table 1. Syntax of concept descriptions.

concept names and a set N_R of *role names*. Depending on the DL under consideration different *constructors* are used to build more complex concept descriptions.

In this work, we consider the DL $\mathcal{ALEN}$ whose concept descriptions are built from the constructors shown in Table 1. The sublanguages $\mathcal{ALE}$, $\mathcal{EL}$, and $\mathcal{ALN}$ of $\mathcal{ALEN}$ are build from subsets of these constructors as also shown in Table 1. We speak of $\mathcal{ALEN}$-,$\mathcal{ALE}$, $\mathcal{EL}$-, and $\mathcal{ALN}$-concept descriptions and more generally of $\mathcal{L}$-concept descriptions for some DL $\mathcal{L}$ in the obvious way.

Later on we will need the notion of the (role) depth of a concept description. Given an $\mathcal{ALEN}$-concept description C its depth, $\mathsf{depth}(C)$, is defined inductively as follows:

- $\mathsf{depth}(\bot) := \mathsf{depth}(\top) := \mathsf{depth}(P) := \mathsf{depth}(\neg P) := 0$;
- $\mathsf{depth}(\leq n\, r) := \mathsf{depth}(\geq n\, r) := 1$;
- $\mathsf{depth}(\exists r.C) := \mathsf{depth}(\forall r.C) := 1 + \mathsf{depth}(C)$; and
- $\mathsf{depth}(C \sqcap D) := max(\mathsf{depth}(C), \mathsf{depth}(D))$.

The *size* of a concept description is simply the number of characters used to represent this concept description as a term where concept and role names as well as parentheses (if used) have size one. Formally, we have that

- $\mathsf{size}(\bot) := \mathsf{size}(\top) := \mathsf{size}(P) := 1$;
- $\mathsf{size}(\neg P) := 2$;

Constructor	Semantics
$\top$	$\Delta_\mathcal{I}$
$\bot$	$\emptyset$
$P \in N_C$	$P^\mathcal{I} \subseteq \Delta_\mathcal{I}$
$\neg P$ with $P \in N_C$	$\Delta_\mathcal{I} \setminus P^\mathcal{I}$
$C \sqcap D$	$C^\mathcal{I} \cap D^\mathcal{I}$
$\exists r.C$	$\{x \in \Delta_\mathcal{I} \mid \exists y : (x,y) \in r^\mathcal{I} \wedge y \in C^\mathcal{I}\}$
$\forall r.C$	$\{x \in \Delta_\mathcal{I} \mid \forall y : (x,y) \in r^\mathcal{I} \rightarrow y \in C^\mathcal{I}\}$
$(\geq nr)$, $r \in N_R$, $n \in \mathbb{N}$ $(\leq nr)$, $r \in N_R$, $n \in \mathbb{N}$	$\{x \in \Delta_\mathcal{I} \mid \#\{y : (x,y) \in r^\mathcal{I}\} \geq n\}$, $\{x \in \Delta_\mathcal{I} \mid \#\{y : (x,y) \in r^\mathcal{I}\} \leq n\}$

Table 2. Semantics of concept descriptions.

- $\mathsf{size}(\leq n\, r) := \mathsf{size}(\geq n\, r) := 2 + \mathsf{size}(n)$ where $\mathsf{size}(n)$ is the number of bits needed to represent n in binary;
- $\mathsf{size}(\exists r.C) := \mathsf{size}(\forall r.C) := 3 + \mathsf{size}(C)$; and
- $\mathsf{size}(C \sqcap D) := 1 + \mathsf{size}(C) + \mathsf{size}(D)$.

The semantics of a concept description is defined in terms of an *interpretation* $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$. The domain $\Delta_\mathcal{I}$ of $\mathcal{I}$ is a non-empty set of individuals and the interpretation function $\cdot^\mathcal{I}$ maps each concept name $P \in N_C$ to a set $P^\mathcal{I} \subseteq \Delta_\mathcal{I}$ and each role name $r \in N_R$ to a binary relation $r^\mathcal{I} \subseteq \Delta_\mathcal{I} \times \Delta_\mathcal{I}$. The extension of $\cdot^\mathcal{I}$ to arbitrary concept descriptions is defined inductively as shown in Table 2.

One of the most important traditional inference problems provided by DL systems is computing the subsumption hierarchy. The concept description C is *subsumed* by the description D ($C \sqsubseteq D$) iff $C^\mathcal{I} \subseteq D^\mathcal{I}$ holds for all interpretations $\mathcal{I}$. The concept descriptions C and D are *equivalent* ($C \equiv D$) iff they subsume each other. As shown in [20], testing subsumption of $\mathcal{ALEN}$-concept descriptions is PSPACE-complete.

In this paper, we are interested in the computation of least common subsumers.

Definition 2.1. Let $\mathcal{L}$ be a DL. Given $n \geq 2$ $\mathcal{L}$-concept descriptions $C_1, \ldots, C_n$, the $\mathcal{L}$-concept description C is the *least common subsumer* (lcs) of $C_1, \ldots, C_n$ ($C = \mathsf{lcs}(C_1, \ldots, C_n)$ for short) iff (i) $C_i \sqsubseteq C$ for all $1 \leq i \leq n$, and (ii) C is the least concept description in $\mathcal{L}$ with this property, i.e., if the $\mathcal{L}$-concept description C' satisfies $C_i \sqsubseteq C'$ for all $1 \leq i \leq n$, then $C \sqsubseteq C'$.

Depending on the DL under consideration, the lcs of two or more descriptions need not always exist. For instance, in the DL CLASSIC the lcs exists if features are interpreted as partial functions but does not exist if features are considered to be total functions [27]. However, if the lcs exists, then it is obviously uniquely determined up to equivalence. The main contribution of this paper is to show that in $\mathcal{ALEN}$ the lcs always exists and that it can be computed effectively.

For the sake of simplicity, in the remainder of this paper we assume that the set of role names N_R is the singleton $\{r\}$. However, all definitions and results can easily be generalized to arbitrary sets of role names as different role names can be dealt with independently.

We say that an $\mathcal{ALEN}$-concept description C is in *normal form* if each conjunction $C_1 \sqcap \cdots \sqcap C_n$ occurring in C contains

1. at most one number restriction of the form $(\geq n\ r)$,
2. at most one number restriction of the form $(\leq n\ r)$, and
3. at most one value restriction of the form $\forall r.C$.

For instance, the concept description $C_1 = (\geq 3\, r) \sqcap \forall r.A \sqcap \exists r.B \sqcap \exists r.A \sqcap (\leq 4\, r)$ is in normal form while $C_2 = (\geq 3\, r) \sqcap (\geq 4\, r) \sqcap \forall r.A \sqcap (\leq 5\, r) \sqcap \forall r.B \sqcap \exists r.B \sqcap \exists r.A \sqcap (\leq 4\, r)$ is not in normal form.

Obviously, by exhaustively applying the following rewrite rules (modulo commutativity and associativity of $\sqcap$) every concept description can be turned into an equivalent concept description in normal form in linear time:

- $(\geq m\, r) \sqcap (\geq n\, r) \to (\geq n\, r)$ if $n \geq m$,
- $(\leq m\, r) \sqcap (\leq n\, r) \to (\leq n\, r)$ if $n \leq m$, and
- $\forall r.C \sqcap \forall r.D \to \forall r.(C \sqcap D)$.

For instance, exhaustively applying these rules to C_2 yields the concept description $(\geq 4\, r) \sqcap (\leq 4\, r) \sqcap \forall r.(A \sqcap B) \sqcap \exists r.B \sqcap \exists r.A$, which is in normal form.

3. Running Example

In order to highlight the main problems that we face in the structural characterization of subsumption and the computation of the lcs in $\mathcal{ALEN}$, we will use the following $\mathcal{ALEN}$-concept descriptions:

$$\begin{array}{rcl} C_{ex} & := & \exists r.(P \sqcap A_1) \sqcap \exists r.(P \sqcap A_2) \sqcap \exists r.(\neg P \sqcap A_1) \ \sqcap \\ & & \exists r.(Q \sqcap A_3) \sqcap \exists r.(\neg Q \sqcap A_3) \sqcap (\leq 2\, r), \text{ and} \\ D_{ex} & := & (\geq 3\, r) \sqcap \forall r.(A_1 \sqcap A_2 \sqcap A_3). \end{array}$$

The key point in the characterization of subsumption and the lcs computation is to describe the "non-trivial" concept descriptions, say C', subsuming a given concept description, say C, where "non-trivial" means that C' does not occur as a conjunct on the top-level of C. Subsequently, these concept descriptions are called *induced*. It suffices to only consider induced concept description that are minimal w.r.t. subsumption.

In what follows, we describe the concept descriptions induced by C_{ex} and D_{ex}. As we will see, given these induced concept descriptions of C_{ex} and D_{ex}, it is then easy to determine the lcs of C_{ex} and D_{ex}. As mentioned in the introduction, the problem is that to obtain the induced concept descriptions, it (at first) seems as if this requires the lcs. However, at this point we can not be sure that the lcs exists. So, we will need to find a way to characterize subsumption, and thus, represent the induced concept descriptions, without resorting to the lcs.

We start with the concept descriptions induced by C_{ex}.

Number restrictions: Because of the existential restrictions on the top-level of C_{ex}, e.g. $\exists r.(P \sqcap A_1)$ and $\exists r.(\neg P \sqcap A_1)$, we know $C_{ex} \sqsubseteq (\geq 2\, r)$, i.e., $(\geq 2\, r)$ is induced by C_{ex}. Conversely, there is no induced $\leq$-restriction, i.e., the most specific $\leq$-restriction subsuming C_{ex} is the $\leq$-restriction explicitly present on the top-level of C_{ex}.

Existential restrictions: Due to the $\leq$-restriction $(\leq 2\, r)$ on the top-level of C_{ex}, each instance of C_{ex} has at most two r-successors. Consequently, some existential restrictions have to be "merged" to a single existential restriction, where "merging" means conjoining the concept descriptions occurring in the existential restrictions. For C_{ex}, there are several ways to merge the five existential restrictions on the top-level of C_{ex} into two existential restrictions. The merging process gives rise to new (derived) concept descriptions, where the only consistent ones are:

$$\begin{array}{rcl} C^1_{ex} & := & \exists r.(P \sqcap Q \sqcap A_1 \sqcap A_2 \sqcap A_3) \ \sqcap \\ & & \exists r.(\neg P \sqcap \neg Q \sqcap A_1 \sqcap A_3) \sqcap (\leq 2\, r), \text{ and} \\ C^2_{ex} & := & \exists r.(P \sqcap \neg Q \sqcap A_1 \sqcap A_2 \sqcap A_3) \ \sqcap \\ & & \exists r.(\neg P \sqcap Q \sqcap A_1 \sqcap A_3) \sqcap (\leq 2\, r). \end{array}$$

It is clear that $C_{ex} \equiv C_{ex}^1 \sqcup C_{ex}^2$ where $\sqcup$ stands for concept disjunction with the obvious semantics. The existential restrictions $\exists r.(P \sqcap A_1 \sqcap A_2 \sqcap A_3)$, $\exists r.(\neg P \sqcap A_1 \sqcap A_3)$, $\exists r.(Q \sqcap A_1 \sqcap A_3)$, and $\exists r.(\neg Q \sqcap A_1 \sqcap A_3)$ subsume both C_{ex}^1 and C_{ex}^2. From this it can be concluded that these restrictions are induced by C_{ex}.

The induced existential restrictions can be obtained by picking one existential restriction from each of the (consistent) derived concept descriptions and applying the lcs operation to them. In our example, we have, for instance, $P \sqcap A_1 \sqcap A_2 \sqcap A_3 \equiv lcs(P \sqcap Q \sqcap A_1 \sqcap A_2 \sqcap A_3, P \sqcap \neg Q \sqcap A_1 \sqcap A_2 \sqcap A_3)$. However, as explained below, our characterization of subsumption avoids to explicitly use the lcs by employing the fact that

$$lcs(C_1, \ldots, C_n) \sqsubseteq D \text{ iff } C_i \sqsubseteq D \text{ for all } 1 \leq i \leq n. \tag{i}$$

Value restrictions: In view of $C_{ex} \equiv C_{ex}^1 \sqcup C_{ex}^2$, it not only follows that every instance of C_{ex} has exactly two r-successors but that these r-successors must satisfy the existential restrictions given in C_{ex}^1 and C_{ex}^2. In either case, all r-successors belong to $A_1 \sqcap A_3$, and thus, the value restriction $\forall r.(A_1 \sqcap A_3)$ is induced by C_{ex}.

There are two things that should be pointed out here: First, note that there is an induced value restriction only if the number of successors induced by existential restrictions coincides with the number in the $\leq$-restriction, because only in this case, we have "full information" about *all* r-successors of an instance of C. For example, if we consider the concept description

$$E_{ex} := (\leq 2\, r) \sqcap \exists r.(A_1 \sqcap A_2) \sqcap \exists r.(A_1 \sqcap A_3),$$

no value restriction is induced as the two concept descriptions, $A_1 \sqcap A_2$ and $A_1 \sqcap A_3$, in the existential restrictions are not disjoint.*

Second, if $\forall r.C'$ is the most specific value restriction induced by C, then C' corresponds to the lcs of all concept descriptions occurring in the merged existential restrictions. In the example, $A_1 \sqcap A_3 \equiv lcs(P \sqcap Q \sqcap A_1 \sqcap A_2 \sqcap A_3, \neg P \sqcap \neg Q \sqcap A_1 \sqcap A_2 \sqcap A_3, P \sqcap \neg Q \sqcap A_1 \sqcap A_2 \sqcap A_3, \neg P \sqcap Q \sqcap A_1 \sqcap A_2 \sqcap A_3)$. Again, using the equivalence (i), we avoid to compute the lcs explicitly in the characterization of subsumption.

*The structural subsumption algorithm introduced in [29], however, computes $\forall r.A_1$ as a value restriction induced by E_{ex}, and thus, is incorrect. For the same reason, the lcs-algorithm presented in [29] is incorrect: although the lcs of E_{ex} and $\forall r.A_1$ is $\top$, the algorithm returns $\forall r.A_1$. In other words, the lcs as computed by the algorithm in [29] in general does not even subsume the input descriptions.

It is easy to see that the only (minimal) concept description induced by D_{ex} is $\exists r.(A_1 \sqcap A_2 \sqcap A_3)$.

Now, given the concept descriptions induced by C_{ex} and D_{ex} it is not hard to verify that the lcs of C_{ex} and D_{ex} can be stated as

$$(\geq 2\, r) \sqcap \forall r.(A_1 \sqcap A_3) \sqcap \exists r.(A_1 \sqcap A_2 \sqcap A_3).$$

4. A Structural Characterization of Subsumption in $\mathcal{ALEN}$

In what follows, let C, D be $\mathcal{ALEN}$-concept descriptions in the normal form introduced in Section 2. The idea behind the characterization of subsumption between C and D is as follows.

Since the case $C \equiv \bot$ and the case $D \equiv \top$ trivially imply $C \sqsubseteq D$, our characterization of subsumption explicitly checks these equivalences. Otherwise, roughly speaking, each conjunct in D, i.e., each (negated) concept name, number restriction, existential restriction, and value restriction occurring on the top-level of D, is compared with the corresponding conjuncts in C. More precisely, one has to consider the conjuncts induced by C, rather than only those explicitly present in C. If all the comparisons succeeded, it follows $C \sqsubseteq D$. In the remainder of this section, the structural comparison between C and D is further explained. Theorem 4.1 states soundness and completeness of our characterization of subsumption.

We first need some notation to access the different parts of the concept descriptions. Given that C is in normal form

- $\mathsf{prim}(C)$ denotes the set of all (negated) concept names occurring on the top-level of C;
- $\mathsf{min}_r(C) := max\{k \mid C \sqsubseteq (\geq k\, r)\}$ (Note that $\mathsf{min}_r(C)$ is always finite);
- $\mathsf{max}_r(C) := min\{k \mid C \sqsubseteq (\leq k\, r)\}$; if there does not exist k with $C \sqsubseteq (\leq k\, r)$, then $\mathsf{max}_r(C) := \infty$;
- $\mathsf{val}_r(C) := C'$ if there exists a value restriction of the form $\forall r.C'$ on the top-level of C, and $\mathsf{val}_r(C) := \top$ otherwise;
- $\mathsf{exr}_r(C) := \{C' \mid \text{there exists } \exists r.C' \text{ on the top-level of } C\}$.

Although the values $\mathsf{min}_r(C)$ and $\mathsf{max}_r(C)$ need not be explicitly present in number restrictions of C, they can be computed in polynomial time in the size of C using an oracle for testing subsumption of $\mathcal{ALEN}$-concept descriptions. (Recall that subsumption testing $\mathcal{ALEN}$-concept descriptions is

PSPACE-complete [20]). For $\mathsf{min}_r(C)$, if there exists a $\geq$-restriction $(\geq m\, r)$ on the top-level of C, then $m \leq \mathsf{min}_r(C) \leq max\{m, |\mathsf{exr}_r(C)|\}$. Thus, $\mathsf{min}_r(C)$ is the maximum k with $C \sqsubseteq (\geq k\, r)$ and $m \leq k \leq max\{m, |\mathsf{exr}_r(C)|\}$. Otherwise, if C does not have a $\geq$-restriction as top-level conjunct, it suffices to check the k's between 0 and $|\mathsf{exr}_r(C)|$. In any case, only a linear number (linear in the size of C) of k's have to be checked. For $\mathsf{max}_r(C)$, it holds that if $\mathsf{val}_r(C) \equiv \bot$, then $\mathsf{max}_r(C) = 0$. Otherwise, if there exists a $\leq$-restriction $(\leq m\, r)$ on the top-level of C, then $\mathsf{max}_r(C) = m$, else $\mathsf{max}_r(C) = \infty$. In our example, we get $\mathsf{min}_r(C_{ex}) = 2$, $\mathsf{max}_r(C_{ex}) = 2$, $\mathsf{min}_r(D_{ex}) = 3$, and $\mathsf{max}_r(D_{ex}) = \infty$.

The structural comparison between the different parts of C and D can now be stated as follows (assuming $C \not\equiv \bot$ and $D \not\equiv \top$):

(Negated) concept names and number restrictions (see Theorem 4.1, 1.–3.): In order for $C \sqsubseteq D$ to hold, it is obvious that the following conditions need to be satisfied: $\mathsf{prim}(D) \subseteq \mathsf{prim}(C)$, $\mathsf{max}_r(C) \leq \mathsf{max}_r(D)$, $\mathsf{min}_r(C) \geq \mathsf{min}_r(D)$. Otherwise, it is easy to construct a counter-model for $C \sqsubseteq D$.

Existential restrictions (see Theorem 4.1, 4.): Given $D' \in \mathsf{exr}_r(D)$, then for $C \sqsubseteq D$ to hold one at first might expect that there needs to exist a $C' \in \mathsf{exr}_r(C)$ with $\mathsf{val}_r(C) \sqcap C' \sqsubseteq D'$. Although this works for$\mathcal{ALE}$-concept descriptions, it fails for $\mathcal{ALEN}$, as illustrated by our running example where $C_{ex} \sqsubseteq \exists r.(P \sqcap A_1 \sqcap A_2 \sqcap A_3)$ even though this existential restriction does not occur in C_{ex}. The reason is that in $\mathcal{ALEN}$ $\leq$-restrictions may require to merge existential restrictions, yielding new (implicit) existential restrictions. To deal with this phenomenon, sets of derived concept descriptions are considered with certain existential restrictions merged (see C^1_{ex} and C^2_{ex} in our running example).

To formulate the merging of existential restrictions of a concept description C we consider what we call existential partitions. Recall that a partition of a set S is a set $\{S_1, \ldots, S_n\}$ of non-empty subsets $S_i \subseteq S$ of S such that the S_i are pairwise disjoint and $S = S_1 \cup \cdots \cup S_n$. If S is an empty set, then the only partition of S is the empty set $\emptyset$. Now, an *existential partition* π of a concept description C is a partition of the set $\mathsf{exr}_r(C) = \{C_1, \ldots, C_m\}$ of existential restrictions occurring on the top-level of C such that the following to conditions are true:

1. The cardinality of π is $n = min\{\mathsf{max}_r(C), |\mathsf{exr}_r(C)|\}$.

2. $\bigsqcap_{C' \in S} C' \sqcap \mathsf{val}_r(C) \not\equiv \bot$ for all $S \in \pi$.

Given $\mathsf{exr}_r(C) = \{C_1, \ldots, C_m\}$, π yields an $\mathcal{ALEN}$-concept description C^π obtained from C by substituting all existential restrictions on the top-level of C with

$$\bigsqcap_{S \in \pi} \exists r. \bigsqcap_{C' \in S} C'.$$

The set of all existential partitions of C is denoted by $\Gamma_r(C)$. Note that $\Gamma_r(C) = \{\emptyset\}$ if $\mathsf{exr}_r(C) = \emptyset$.

In our running example, we have $\mathsf{exr}_r(C_{ex}) = \{C_{ex,1}, \ldots, C_{ex,5}\}$ where $C_{ex,1} = P \sqcap A_1$, $C_{ex,2} = P \sqcap A_2$, etc. Then, $\Gamma_r(C_{ex})$ consists of the following two existential partitions:

$$\begin{aligned} \pi_1 &= \{\{C_{ex,1}, C_{ex,2}, C_{ex,4}\}, \{C_{ex,3}, C_{ex,5}\}\}; \\ \pi_2 &= \{\{C_{ex,1}, C_{ex,2}, C_{ex,5}\}, \{C_{ex,3}, C_{ex,4}\}\}, \end{aligned}$$

and it is $C_{ex}^{\pi_i} = C_{ex}^i$, $i = 1, 2$ (see Section 3).

For the characterization of subsumption, we will use the following notation:

$$\mathsf{exr}_r(C)^\pi := \{\bigsqcap_{C' \in S} C' \mid S \in \pi\}.$$

Now, in case $\mathsf{exr}_r(C) \neq \emptyset$, $C \sqsubseteq D$ implies that for each $D' \in \mathsf{exr}_r(D)$ the following holds: for each $\pi \in \Gamma_r(C)$, there exists $C'_\pi \in \mathsf{exr}_r(C)^\pi$ such that $C'_\pi \sqcap \mathsf{val}_r(C) \sqsubseteq D'$. This is what is stated in Theorem 4.1, 4. Note that, using the equivalence (i) and provided that the lcs of $\mathcal{ALEN}$-concept descriptions always exists, this condition could be rewritten as: there exists a set $M := \{C'_\pi \in \mathsf{exr}_r(C)^\pi \mid \pi \in \Gamma_r(C)\}$ with $lcs(\{C' \sqcap \mathsf{val}_r(C) \mid C' \in M\}) \sqsubseteq D'$. However, since the existence of the lcs has not been proved at this point, the lcs cannot be used in the structural characterization of subsumption, unless the characterization and the lcs computation are interleaved.

If $\mathsf{exr}_r(C) = \emptyset$, then for all $D' \in \mathsf{exr}_r(D)$ it must hold, $\min_r(C) \geq 1$ and $\mathsf{val}_r(C) \sqsubseteq D'$.

In our running example, $C_{ex} \sqsubseteq \exists r.(P \sqcap A_1 \sqcap A_2 \sqcap A_3)$ illustrates the case where $\mathsf{exr}_r(C) \neq \emptyset$ and $D_{ex} \sqsubseteq \exists r.(A_1 \sqcap A_2 \sqcap A_3)$ illustrates the case that $\mathsf{exr}_r(C) = \emptyset$.

Value restrictions (see Theorem 4.1, 5.): Value restrictions can only be induced for two reasons. First, if $\mathsf{max}_r(C) = 0$, then $C \sqsubseteq \forall r.\bot$, and thus, $C \sqsubseteq \forall r.C'$ for all concept descriptions C'.

Second, the merging of existential restrictions may induce value restrictions. In contrast to induced existential restrictions, however, one further needs to take into account $\geq$-restrictions induced by "incompatible" existential restrictions. The biggest number in such $\geq$-restrictions is defined as follows:

$$\kappa_r(C) := \mathsf{min}_r(\forall r.\mathsf{val}_r(C) \sqcap \textstyle\bigsqcap_{C' \in \mathsf{exr}_r(C)} \exists r.C').$$

Note that $\kappa_r(C) = 0$ if $\mathsf{exr}_r(C) = \emptyset$. In our example, $\kappa_r(C_{ex}) = 2$, $\kappa_r(D_{ex}) = 0$, and $\kappa_r(E_{ex}) = 1$.

Now, only if $\kappa_r(C) = \mathsf{max}_r(C)$, value restrictions can be induced, since only then we "know" all the r-successors of instances of C. In our example, $\kappa_r(C_{ex}) = \mathsf{max}_r(C_{ex}) = 2$, which accounts for $C_{ex} \sqsubseteq \forall r.(A_1 \sqcap A_3)$. Conversely, $E_{ex} \not\sqsubseteq \forall r.A_1$ since $1 = \kappa_r(E_{ex}) < \mathsf{max}_r(E_{ex}) = 2$.

To formally state the comparison between value restrictions of C and D, we use the following notation:

$$\mathsf{exr}_r(C)^* := \bigcup_{\pi \in \Gamma_r(C)} \mathsf{exr}_r(C)^\pi.$$

One can show that $C \sqsubseteq D$ implies $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(D)$ provided that $\kappa_r(C) < \mathsf{max}_r(C)$. In case $0 < \kappa(C) = \mathsf{max}_r(C)$, however, it suffices if the value restriction on the top-level of D satisfies $\mathsf{val}_r(C) \sqcap C' \sqsubseteq \mathsf{val}_r(D)$ for all $C' \in \mathsf{exr}_r(C)^*$. Again, using the equivalence (i) and provided that the lcs of $\mathcal{ALEN}$-concept descriptions always exists (which, as we will show, is in fact the case but we have not proved this at this point), this condition can be restated as $\mathsf{val}_r(C) \sqcap lcs(\mathsf{exr}_r(C)^*) \sqsubseteq D'$. For reasons already mentioned, we have not employed this variant since it would require to interleave the structural characterization of subsumption and the proof of the existence of the lcs and its computation.

We are now ready to state the characterization of subsumption in $\mathcal{ALEN}$.

THEOREM 4.1. *Let C, D be two $\mathcal{ALEN}$-concept descriptions with $\mathsf{exr}_r(C) = \{C_1, \ldots, C_m\}$. Then $C \sqsubseteq D$ iff $C \equiv \bot$, $D \equiv \top$, or the following holds:*

1. $\mathsf{prim}(D) \subseteq \mathsf{prim}(C)$;
2. $\mathsf{max}_r(C) \leq \mathsf{max}_r(D)$;
3. $\mathsf{min}_r(C) \geq \mathsf{min}_r(D)$;
4. *for all $D' \in \mathsf{exr}_r(D)$ it holds that*
   (a) $\mathsf{exr}_r(C) = \emptyset$, $\mathsf{min}_r(C) \geq 1$, *and* $\mathsf{val}_r(C) \sqsubseteq D'$; *or*
   (b) $\mathsf{exr}_r(C) \neq \emptyset$ *and for each* $\pi \in \Gamma_r(C)$, *there exists* $C' \in \mathsf{exr}_r(C)^\pi$ *such that* $C' \sqcap \mathsf{val}_r(C) \sqsubseteq D'$; *and*

5. *if* $\mathsf{val}_r(D) \not\equiv \top$*, then*
   (a) $\mathsf{max}_r(C) = 0$*; or*
   (b) $\kappa_r(C) < \mathsf{max}_r(C)$ *and* $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(D)$*; or*
   (c) $0 < \kappa_r(C) = \mathsf{max}_r(C)$ *and* $\mathsf{val}_r(C) \sqcap C' \sqsubseteq \mathsf{val}_r(D)$ *for all* $C' \in \mathsf{exr}_r(C)^*$.

In the remainder of this section, we prove the above theorem. For the proof of soundness, i.e., the if-direction of Theorem 4.1, we need the following lemma.

LEMMA 4.2. *Let C be an $\mathcal{ALEN}$-concept description with $\mathsf{exr}_r(C) = \{C_1, \ldots, C_m\} \neq \emptyset$. Then, for each model $\mathcal{I}$ of C and each $x \in C^{\mathcal{I}}$, there exists $\pi \in \Gamma_r(C)$ such that for each $C' \in \mathsf{exr}_r(C)^{\pi}$, there exists $y \in \Delta_{\mathcal{I}}$ with $(x, y) \in r^{\mathcal{I}}$ and $y \in C'^{\mathcal{I}}$.*

Proof: By induction on $n := |\mathsf{exr}_r(C)| - \mathsf{max}_r(C)$.

Let $n \leq 0$**:** Then $\Gamma_r(C) = \{\pi\}$ where π is the partition $\{\{C'\}|\mathcal{C}' \in \mathsf{exr}_r(C)\}$. In particular, $\mathsf{exr}_r(C)^{\pi} = \mathsf{exr}_r(C)$. Since $x \in C^{\mathcal{I}}$, for each $C' \in \mathsf{exr}_r(C)$ there exists an r-successor y of x in $\mathcal{I}$ with $y \in C'^{\mathcal{I}}$.

Let $n > 0$**:** Since $x \in C^{\mathcal{I}}$, there exist at most $\mathsf{max}_r(C)$ r-successors of x in $\mathcal{I}$. Since $|\mathsf{exr}_r(C)| > \mathsf{max}_r(C)$, there exist $C_{i_1}, C_{i_2} \in \mathsf{exr}_r(C)$, $i_1 \neq i_2$, such that there exists an r-successor y of x in $\mathcal{I}$ with $y \in C_{i_1}^{\mathcal{I}}$ and $y \in C_{i_2}^{\mathcal{I}}$. W.l.o.g. let $i_1 = m-1$ and $i_2 = m$.

Now, let D be the $\mathcal{ALEN}$-concept description obtained from C by conjoining the existential restriction $\exists r.C_{m-1}$ and $\exists r.C_m$ on the top-level of C, i.e., replacing them by $\exists r.(C_{m-1} \sqcap C_m)$. Again w.l.o.g., let $\mathsf{exr}_r(D) = \{D_1, \ldots, D_{m-1}\}$ with $D_{m-1} = C_{m-1} \sqcap C_m$ and $D_i = C_i$ for all $1 \leq i \leq m-2$. Since $|\mathsf{exr}_r(D)| - \mathsf{max}_r(D) < n$, we get by induction that there exists a $\pi' \in \Gamma_r(D)$ such that (*): for each $D' \in \mathsf{exr}_r(D)^{\pi'}$, there exists $y \in \Delta_{\mathcal{I}}$ with $(x, y) \in r^{\mathcal{I}}$ and $y \in D'^{\mathcal{I}}$.

We extend π' to an existential partition $\pi \in \Gamma_r(C)$ by adding C_m to the set in π' containing $C_{m-1} \sqcap C_m$ and replacing $C_{m-1} \sqcap C_m$ by C_{m-1}. It is easy to see that in fact $\pi \in \Gamma_r(C)$ and $\mathsf{exr}_r(C)^{\pi} = \mathsf{exr}_r(C)^{\pi'}$. Now the claim follows by (*). ■

Proof of the if-direction of Theorem 4.1: If $C \equiv \bot$ or $D \equiv \top$, then $C \sqsubseteq D$. Assume $C \not\equiv \bot$ and $D \not\equiv \top$, and (1)–(5) are satisfied. In order to show $C \sqsubseteq D$, we show that $x \in C^{\mathcal{I}}$ implies $x \in D^{\mathcal{I}}$ for every model $\mathcal{I}$ and

$x \in \Delta_\mathcal{I}$. Obviously, it is sufficient to show $x \in D'^\mathcal{I}$ for each conjunct D' occurring on the top-level of D.

Let $D' \in \mathsf{prim}(D)$: Since $\mathsf{prim}(D) \subseteq \mathsf{prim}(C)$ and $x \in C^\mathcal{I}$, we obtain $x \in D'^\mathcal{I}$.

Let $D' = (\leq n\, r)$: By condition (2), it is $\mathsf{max}_r(C) \leq \mathsf{max}_r(D) \leq n$. Since $x \in C^\mathcal{I}$, it follows $x \in (\leq \mathsf{max}_r(C)\, r)^\mathcal{I} \subseteq (\leq n\, r)^\mathcal{I}$.

Let $D' = (\geq n\, r)$: By condition (3), it is $\mathsf{min}_r(C) \geq \mathsf{min}_r(D) \geq n$. Since $x \in C^\mathcal{I}$, it follows $x \in (\geq \mathsf{min}_r(C)\, r)^\mathcal{I} \subseteq (\geq n\, r)^\mathcal{I}$.

Let $D' = \exists r.D_1$: Assume $\mathsf{exr}_r(C) = \emptyset$. Then 4., a) implies $\mathsf{min}_r(C) \geq 1$ and $\mathsf{val}_r(C) \sqsubseteq D_1$. Hence, there exists an r-successor y of x in $\mathcal{I}$ with $y \in \mathsf{val}_r(C)^\mathcal{I} \subseteq D_1^\mathcal{I}$. Thus, $x \in (\exists r.D_1)^\mathcal{I}$.

Assume $\mathsf{exr}_r(C) \neq \emptyset$. By Lemma 4.2, there exists $\pi' \in \Gamma_r(C)$ such that for each $C' \in \mathsf{exr}_r(C)^{\pi'}$ there exists an r-successor y of x in $\mathcal{I}$ with $y \in C'^\mathcal{I}$. By condition 4., b), there exists $C' \in \mathsf{exr}_r(C)^{\pi'}$ with $C' \sqcap \mathsf{val}_r(C) \sqsubseteq D_1$. Thus, there is an r-successor y' of x in $\mathcal{I}$ with $y' \in (C' \sqcap \mathsf{val}_r(C))^\mathcal{I} \subseteq D_1^\mathcal{I}$. Hence, $x \in (\exists r.D_1)^\mathcal{I}$.

Let $D' = \forall r.D_1$: If $\mathsf{max}_r(C) = 0$, then there is no r-successor of x in $\mathcal{I}$, and hence, $x \in (\forall r.D_1)^\mathcal{I}$.

Assume $\kappa_r(C) < \mathsf{max}_r(C)$. Let y be an arbitrary r-successor of $x \in \mathcal{I}$. Then $x \in C^\mathcal{I}$ implies $y \in \mathsf{val}_r(C)^\mathcal{I}$ and since $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(D)$ by 5., b), we have $y \in \mathsf{val}_r(D)^\mathcal{I}$. Thus, $x \in (\forall r.D_1)^\mathcal{I}$.

Finally, assume $0 < \kappa_r(C) = \mathsf{max}_r(C)$. Let y be an arbitrary r-successor of x in $\mathcal{I}$. Define $\mathcal{C}(y) := \{C' \in \mathsf{exr}_r(C)^* \mid y \in C'^\mathcal{I}\}$. Since $0 < \kappa_r(C) = \mathsf{max}_r(C)$, we know that there exist exactly $\kappa_r(C)$ r-successors of x in $\mathcal{I}$, and since there exist $\kappa_r(C)$ disjoint existential restrictions on the top-level of C, we get $\mathcal{C}(y) \neq \emptyset$, say $C' \in \mathcal{C}(y)$. Then, $y \in C'^\mathcal{I}$. Obviously $y \in \mathsf{val}_r(C)^\mathcal{I}$. Now, $\mathsf{val}_r(C) \sqcap C' \sqsubseteq \mathsf{val}_r(D)$ implies $y \in \mathsf{val}_r(D)^\mathcal{I}$ and hence, $x \in (\forall r.D_1)^\mathcal{I}$. ■

For the proof of completeness of the characterization of subsumption we have to construct certain models of $\mathcal{ALEN}$-concept descriptions. The following lemma ensures the existence of these models. It easily follows from the definition of $\mathsf{min}_r(C)$ and the (finite) tree model property of $\mathcal{ALEN}$-concept descriptions [21].

LEMMA 4.3. *Let C be an $\mathcal{ALEN}$-concept description, $C \not\equiv \bot$. Then there exists a tree model $\mathcal{I}$ with root x_0 of C such that $x_0 \in C^\mathcal{I}$ and x_0 has exactly $\mathsf{min}_r(C)$ different r-successors in $\mathcal{I}$.*

Proof of the only-if-direction of Theorem 4.1: If $C \equiv \bot$ or $D \equiv \top$, nothing has to be shown. Assume $C \sqsubseteq D$, $C \not\equiv \bot$, and $D \not\equiv \top$. Then we have to show that C, D satisfy the statements (1)–(5) in Theorem 4.1.

(1) Assume that there exists $Q \in \mathsf{prim}(D) \setminus \mathsf{prim}(C)$. Let $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ be a tree model with root x_0 of C with $x_0 \in C^\mathcal{I}$ (see Lemma 4.3).

If $Q \in N_C$, then define $\mathcal{J} := (\Delta_\mathcal{J}, \cdot^\mathcal{J})$ with

- $P^\mathcal{J} := \begin{cases} P^\mathcal{I} \setminus \{x_0\}, & P = Q; \\ P^\mathcal{I}, & P \neq Q; \end{cases}$
- $r^\mathcal{J} := r^\mathcal{I}$.

If $Q = \neg P'$ for some $P' \in N_C$, then define $\mathcal{J} := (\Delta_\mathcal{J}, \cdot^\mathcal{J})$ with

- $P^\mathcal{J} := \begin{cases} P^\mathcal{I} \cup \{x_0\}, & P = P'; \\ P^\mathcal{I}, & P \neq P'; \end{cases}$
- $r^\mathcal{J} := r^\mathcal{I}$.

In either case, $\mathcal{J}$ is a model of C with $x_0 \in C^\mathcal{J}$, but $x_0 \notin D^\mathcal{J}$ in contradiction to $C \sqsubseteq D$.

Thus, $\mathsf{prim}(D) \subseteq \mathsf{prim}(C)$.

(2) Assume $\mathsf{max}_r(C) > \mathsf{max}_r(D)$. In particular, $\mathsf{max}_r(D)$ is some nonnegative integer. By the definition of $\mathsf{max}_r(C)$ there exists a model $\mathcal{I}$ and $x_0 \in \Delta_\mathcal{I}$ with $x_0 \in C^\mathcal{I}$ such that x_0 has $\mathsf{max}_r(D) + 1$ r-successors; otherwise we would have $\mathsf{max}_r(C) \leq \mathsf{max}(D)$. Obviously, $x_0 \notin D^\mathcal{I}$. Thus, by contradiction $\mathsf{max}_r(C) \leq \mathsf{max}_r(D)$.

(3) Assume $\mathsf{min}_r(C) < \mathsf{min}_r(D)$. By Lemma 4.3, there exists a (tree) model $\mathcal{I}$ with $x_0 \in C^\mathcal{I}$ having exactly $\mathsf{min}_r(C)$ r-successors. Obviously, $x_0 \notin D^\mathcal{I}$.

(4) Let $D' \in \mathsf{exr}_r(D)$ and assume $\mathsf{exr}_r(C) = \emptyset$. If $\mathsf{min}_r(C) = 0$, then there exists a tree model $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ with root x_0 of C with $x_0 \in C^\mathcal{I}$ and x_0 has 0 r-successors in $\mathcal{I}$, i.e., $r^\mathcal{I} = \emptyset$. Obviously, this implies $x_0 \notin D^\mathcal{I}$ in contradiction to $C \sqsubseteq D$. Hence, $\mathsf{min}_r(C) \geq 1$. Let $\mathcal{I} = (\Delta_\mathcal{I}, \cdot^\mathcal{I})$ be a tree model of C with root x_0 and $x_0 \in C^\mathcal{I}$. Assume $\mathsf{val}_r(C) \not\sqsubseteq D'$, i.e., there exists a tree model $\mathcal{I}' = (\Delta_{\mathcal{I}'}, \cdot^{\mathcal{I}'})$ with root y_0 such that $y_0 \in \mathsf{val}_r(C)^{\mathcal{I}'}$ and $y_0 \notin D'^{\mathcal{I}'}$. Let $\mathcal{I}_i$, $1 \leq i \leq \mathsf{min}_r(C)$, be pairwise disjoint copies of $\mathcal{I}'$ with roots y_i, which are also pairwise disjoint with $\mathcal{I}$. Define $\mathcal{J} := (\Delta_\mathcal{J}, \cdot^\mathcal{J})$ as follows:

- $\Delta_{\mathcal{J}} := \Delta_{\mathcal{I}} \cup \bigcup_{1 \leq i \leq \mathsf{min}_r(C)} \Delta_{\mathcal{I}_i}$;
- $P^{\mathcal{J}} := P^{\mathcal{I}} \cup \bigcup_{1 \leq i \leq \mathsf{min}_r(C)} P^{\mathcal{I}_i}$ for all $P \in N_C$;
- $r^{\mathcal{J}} := \bigcup_{1 \leq i \leq \mathsf{min}_r(C)} r^{\mathcal{I}_i} \cup \{(x_0, y_i) \mid 1 \leq i \leq \mathsf{min}_r(C)\}$.

It is easy to see that $\mathcal{I}$ is a model of C with $x_0 \in C^{\mathcal{I}}$, but since there does not exist an r-successor y of x_0 with $y \in D'^{\mathcal{J}}$, it is $x_0 \notin D^{\mathcal{I}}$ in contradiction to $C \sqsubseteq D$. Thus, $\mathsf{val}_r(C) \sqsubseteq D'$.

Now assume $\mathsf{exr}_r(C) \neq \emptyset$. We have to show that for each $\pi \in \Gamma_r(C)$, there exists $C' \in \mathsf{exr}_r(C)^\pi$ with $C' \sqcap \mathsf{val}_r(C) \sqsubseteq D'$. Assume that there exists $\pi \in \Gamma_r(C)$ such that $C' \sqcap \mathsf{val}_r(C) \not\sqsubseteq D'$ for all $C' \in \mathsf{exr}_r(C)^\pi$. Let C_π be the $\mathcal{ALEN}$-concept description obtained from C by removing all existential restrictions from the top-level of C and conjoining all existential restrictions from $\mathsf{exr}_r(C)^\pi$. We define a tree model $\mathcal{I}$ with root x_0 such that $x_0 \in C_\pi^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ and $x_0 \notin D^{\mathcal{I}}$ as follows:

- Define $\mathcal{I}_0 := (\{x_0\}, \cdot^{\mathcal{I}_0})$ with
  - $P^{\mathcal{I}_0} := \begin{cases} \{x_0\}, & P \in \mathsf{prim}(C); \\ \emptyset, & P \notin \mathsf{prim}(C) \end{cases}$ for all $P \in N_C$;
  - $r^{\mathcal{I}_0} := \emptyset$.
- For each $C' \in \mathsf{exr}_r(C)^\pi$, let $\mathcal{I}(C') := (\Delta_{\mathcal{I}(C')}, \cdot^{\mathcal{I}(C')})$ be a tree model with root $y(C')$ such that $y(C') \in (C' \sqcap \mathsf{val}_r(C))^{\mathcal{I}(C')}$ and $y(C') \notin D'^{\mathcal{I}(C')}$.
- If $\mathsf{min}_r(C) > |\mathsf{exr}_r(C)^\pi|$, then let $\mathcal{I}_1, \ldots, \mathcal{I}_k$, $k := \mathsf{min}_r(C) - |\mathsf{exr}_r(C)^\pi|$, be copies of $\mathcal{I}(C')$, as defined above, for some C' with root y_i, $1 \leq i \leq k$.

W.l.o.g. let all these interpretations be pairwise disjoint. Now, define $\mathcal{I} = (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ by

- $\Delta_{\mathcal{I}} := \{x_0\} \cup \bigcup_{1 \leq i \leq k} \Delta_{\mathcal{I}_i} \cup \bigcup_{C' \in \mathsf{exr}_r(C)^\pi} \Delta_{\mathcal{I}(C')}$;
- $P^{\mathcal{I}} := P^{\mathcal{I}_0} \cup \bigcup_{1 \leq i \leq k} P^{\mathcal{I}_i} \cup \bigcup_{C' \in \mathsf{exr}_r(C')^\pi} P^{\mathcal{I}(C')}$ for all $P \in N_C$;
- $r^{\mathcal{I}} := \{(x_0, y(C')) \mid C' \in \mathsf{exr}_r(C)^\pi\} \cup \{(x_0, y_i) \mid 1 \leq i \leq k\} \cup \bigcup_{1 \leq i \leq k} r^{\mathcal{I}_i} \cup \bigcup_{C' \in \mathsf{exr}_r(C)^\pi} r^{\mathcal{I}(C')}$.

By construction, $\mathcal{I}$ is a model of C_π and hence of C with $x_0 \in C^{\mathcal{I}}$. But since there does not exist an r-successor y of x_0 with $y \in D'^{\mathcal{I}}$, it follows $x_0 \notin D^{\mathcal{I}}$ in contradiction to $C \sqsubseteq D$. Thus, for each $\pi \in \Gamma_r(C)$ there exists $C' \in \mathsf{exr}_r(C)^\pi$ with $C' \sqcap \mathsf{val}_r(C) \sqsubseteq D'$.

(5) Assume $\mathsf{val}_r(D) \not\equiv \top$ and $\mathsf{max}_r(C) > 0$ (otherwise, nothing has to be shown). We have to distinguish two cases:

1. Assume $\kappa_r(C) < \mathsf{max}_r(C)$. We have to show $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(D)$. Proof by contraposition: assuming $\mathsf{val}_r(C) \not\sqsubseteq \mathsf{val}_r(D)$, we define a tree model $\mathcal{I}$ with root x_0 of C such that $x_0 \in C^{\mathcal{I}}$ and $x_0 \notin D^{\mathcal{I}}$.

In order to define $\mathcal{I}$, we distinguish two cases:

a) there exists a $\geq$-restriction of the form $(\geq m\, r)$ on the top-level of C with $m > \kappa_r(C)$, and

b) there does not exist such a number restriction (hence, $\mathsf{max}_r(C) = \infty$).

Ad (a): Consider the $\mathcal{ALEN}$-concept description C' obtained from C by removing $(\geq m\, r)$ from the top-level of C. Obviously, it is $\kappa_r(C) = \kappa_r(C')$. Let $\mathcal{I}'$ be a tree model of C' with root x_0 such that $x_0 \in C'^{\mathcal{I}}$ and x_0 has exactly $\kappa_r(C')$ r-successors in $\mathcal{I}'$ (see Lemma 4.3). By assumption, there exists a model $\mathcal{I}''$ of $\mathsf{val}_r(C)$ with root y_0 such that $y_0 \in \mathsf{val}_r(C)^{\mathcal{I}''}$ and $y_0 \notin \mathsf{val}_r(D)^{\mathcal{I}''}$. Let $\mathcal{I}_i$, $1 \leq i \leq m - \kappa_r(C)$, be disjoint copies of $\mathcal{I}''$ with roots y_i, which are w.l.o.g. also pairwise disjoint with $\mathcal{I}'$. Note that by assumption $m - \kappa_r(C) \geq 1$. Define $\mathcal{I} := (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ with

- $\Delta_{\mathcal{I}} := \Delta'_{\mathcal{I}} \cup \bigcup_{1 \leq i \leq m-\kappa_r(C)} \Delta_{\mathcal{I}_i}$;
- $P^{\mathcal{I}} := P^{\mathcal{I}'} \cup \bigcup_{1 \leq i \leq m-\kappa_r(C)} P^{\mathcal{I}_i}$ for all $P \in N_C$;
- $r^{\mathcal{I}} := r^{\mathcal{I}'} \cup \bigcup_{1 \leq i \leq m-\kappa_r(C)} r^{\mathcal{I}_i} \cup \{(x_0, y_i) \mid 1 \leq i \leq m - \kappa_r(C)\}$.

It is easy to see that $\mathcal{I}$ is a model of C with $x_0 \in C^{\mathcal{I}}$, but since x_0 has at least the r-successor y_1 with $y_1 \notin \mathsf{val}_r(D)^{\mathcal{I}}$, we obtain $x_0 \notin D^{\mathcal{I}}$ in contradiction to $C \sqsubseteq D$. Thus, $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(D)$.

Ad (b): Let $\mathcal{I}'$ be a tree model of C' with root x_0 such that $x_0 \in C^{\mathcal{I}}$ and x_0 has exactly $\kappa_r(C)$ r-successors in $\mathcal{I}'$ (see Lemma 4.3). By assumption, there exists a model $\mathcal{I}''$ of $\mathsf{val}_r(C)$ with root y_0 such that $y_0 \in \mathsf{val}_r(C)^{\mathcal{I}''}$ and $y_0 \notin \mathsf{val}_r(D)^{\mathcal{I}''}$. Define $\mathcal{I} := (\Delta_{\mathcal{I}}, \cdot^{\mathcal{I}})$ with

- $\Delta_{\mathcal{I}} := \Delta'_{\mathcal{I}} \cup \Delta_{\mathcal{I}''}$;
- $P^{\mathcal{I}} := P^{\mathcal{I}'} \cup P^{\mathcal{I}''}$ for all $P \in N_C$;
- $r^{\mathcal{I}} := r^{\mathcal{I}'} \cup r^{\mathcal{I}''} \cup \{(x_0, y_0)\}$.

Since by assumption, $\kappa_r(C) < \mathsf{max}_r(C)$, $\mathcal{I}$ is a model of C with $x_0 \in C^{\mathcal{I}}$. But since x_0 has at least the r-successor y_0 with $y_0 \notin \mathsf{val}_r(D)^{\mathcal{I}}$, we obtain $x_0 \notin D^{\mathcal{I}}$ in contradiction to $C \sqsubseteq D$. Thus, $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(D)$.

2. Assume $\kappa_r(C) = \mathsf{max}_r(C)$. Assume that there exists $C' \in \mathsf{exr}_r(C)^*$ with $\mathsf{val}_r(C) \sqcap C' \not\sqsubseteq \mathsf{val}_r(D)$. Then there exists $\pi' \in \Gamma_r(C)$ such that $C' \in \mathsf{exr}_r(C)^{\pi'}$. Note that $|\pi'| = \mathsf{max}_r(C)$. Let $C_{\pi'}$ be the $\mathcal{ALEN}$-concept description obtained from C by removing all existential restrictions from the top-level of C and conjoining all existential restrictions according to $\mathsf{exr}_r(C)^{\pi'}$. Obviously, $C_{\pi'} \sqsubseteq C$. Since, using $C \not\equiv \bot$, from the definition of existential partitions it immediately follows that $C_{\pi'} \not\equiv \bot$, there exists a tree model $\mathcal{I}$ of $C_{\pi'}$ with root $x_0 \in C_{\pi'}^{\mathcal{I}}$. Moreover, since $\mathsf{max}_r(C) = \kappa_r(C)$, for every $C'' \in \mathsf{exr}_r(C)^{\pi'}$, there exists exactly one r-successor y of x_0 in $\mathcal{I}$ with $y \in C''^{\mathcal{I}}$; x_0 has no other r-successors. In particular, there exists an r-successor y of x_0 such that $y \in (\mathsf{val}_r(C) \sqcap C')^{\mathcal{I}}$. By assumption, there exists a tree model $\mathcal{I}' = (\Delta_{\mathcal{I}'}, \cdot^{\mathcal{I}'})$ with root y_0 such that $y_0 \in (\mathsf{val}_r(C) \sqcap C')^{\mathcal{I}'}$ and $y_0 \notin \mathsf{val}_r(D)^{\mathcal{I}'}$. Define $\mathcal{J} = (\Delta_{\mathcal{J}}, \cdot^{\mathcal{J}})$ with

- $\Delta_{\mathcal{J}} := \Delta_{\mathcal{I}} \cup \Delta_{\mathcal{I}'}$;
- $P^{\mathcal{J}} := P^{\mathcal{I}} \cup P^{\mathcal{I}'}$ for all $P \in N_C$;
- $r^{\mathcal{J}} := (r^{\mathcal{I}} \setminus \{(x_0, y)\}) \cup r^{\mathcal{I}'} \cup \{(x_0, y_0)\}$.

Then $\mathcal{J}$ is a model of C with $x_0 \in C^{\mathcal{J}}$. But since the r-successor y_0 of x_0 in $\mathcal{J}$ is not an instance of $\mathsf{val}_r(D)^{\mathcal{J}}$, it follows $x_0 \notin D^{\mathcal{J}}$ in contradiction to $C \sqsubseteq D$. Thus, $\mathsf{val}_r(C) \sqcap C' \sqsubseteq \mathsf{val}_r(D)$ for all $C' \in \mathsf{exr}_r(C)^*$. ■

We note that our characterization of subsumption is not intended to yield a (practical) subsumption algorithm. It rather serves as the basis for the lcs algorithm presented in the next section.

5. Computing the LCS in $\mathcal{ALEN}$

We now present a recursive algorithm for computing the lcs of two $\mathcal{ALEN}$-concept descriptions C and D. It is depicted in Figure 1 and called c-lcs, with further explanation given below. Although, the algorithm is presented as a binary operation (working on C and D), it can be generalized to be applicable to finite sets of concept description in the obvious way:

$$\mathsf{c\text{-}lcs}(C_1, \ldots, C_n) := \mathsf{c\text{-}lcs}(C_1, \mathsf{c\text{-}lcs}(C_2, \cdots, \mathsf{c\text{-}lcs}(C_{n-1}, C_n) \cdots)).$$

In fact, within the algorithm this is used.

Before we prove that c-lcs in fact computes the lcs of $\mathcal{ALEN}$-concept descriptions, we illustrate the algorithm in case that $C \not\sqsubseteq D$ and $D \not\sqsubseteq C$ (since the special cases $C \sqsubseteq D$, $D \sqsubseteq C$ are trivial), using our running example introduced in Section 3. The conjuncts occurring on the top-level of $\mathsf{c\text{-}lcs}(C, D)$ can, as before, be divided into three parts, namely (1) (negated) concept names and number restrictions, (2) the existential restrictions, and (3) the value restriction. These conjuncts are defined in such a way that

(a) the conditions 1.–5. in Theorem 4.1 for $C \sqsubseteq \mathsf{c\text{-}lcs}(C, D)$ and $D \sqsubseteq \mathsf{c\text{-}lcs}(C, D)$ are satisfied, and

(b) $\mathsf{c\text{-}lcs}(C, D)$ is the least concept description (w.r.t. $\sqsubseteq$) satisfying (a).

For the conditions 1.–3., this is quite obvious, since, for $E \in \{C, D\}$, we obtain

- $\mathsf{prim}(\mathsf{c\text{-}lcs}(C, D)) = \mathsf{prim}(C) \cap \mathsf{prim}(D) \subseteq \mathsf{prim}(E)$,
- $\mathsf{min}_r(\mathsf{c\text{-}lcs}(C, D)) = min\{\mathsf{min}_r(C), \mathsf{min}_r(D)\} \leq \mathsf{min}_r(E)$, and
- $\mathsf{max}_r(\mathsf{c\text{-}lcs}(C, D)) = max\{\mathsf{max}_r(C), \mathsf{max}_r(D)\} \geq \mathsf{max}_r(E)$.

In our running example, we have

- $\mathsf{prim}(\mathsf{c\text{-}lcs}(C_{ex}, D_{ex})) = \emptyset$,
- $\mathsf{min}_r(\mathsf{c\text{-}lcs}(C_{ex}, D_{ex})) = 2$, and
- $\mathsf{max}_r(\mathsf{c\text{-}lcs}(C_{ex}, D_{ex})) = \infty$.

Things are more involved for existential and value restrictions. Let us first consider the definition of $\mathsf{exr}_r(\mathsf{c\text{-}lcs}(C, D))$, i.e., the existential restrictions obtained from the sets $\mathcal{C}_r(C)$ and $\mathcal{C}_r(D)$ (see Figure 1). Roughly speaking, if $\Gamma_r(E) \neq \{\emptyset\}$, then $\mathcal{C}_r(E)$ contains all (minimal) concept descriptions occurring in an existential restriction induced by E for $E \in \{C, D\}$. Each such concept description is obtained as the recursively computed lcs of a set of concept descriptions consisting of one concept description from $\mathsf{exr}_r(E)^\pi$ (conjoined with $\mathsf{val}_r(E)$) for each $\pi \in \Gamma_r(E)$. In our running example, each pair of concept descriptions occurring in the existential restrictions on the top-level of C^1_{ex} and C^2_{ex}, respectively, yields such an lcs, and thus $\mathcal{C}_r(C_{ex}) = \{P \sqcap A_1 \sqcap A_2 \sqcap A_3, Q \sqcap A_1 \sqcap A_3, \neg Q \sqcap A_1 \sqcap A_3, \neg P \sqcap A_1 \sqcap A_3\}$ (see Section 3).

If $\mathsf{min}_r(E) = 1$ and $\Gamma_r(E) = \emptyset$, we set $\mathcal{C}_r(E) = \{\mathsf{val}_r(E)\}$ since then $\mathsf{exr}_r(E) = \emptyset$, and $\exists r.\mathsf{val}_r(E)$ is the unique minimal existential restriction induced by E. This case is illustrated by D_{ex} in our running example, where $\mathsf{min}_r(D_{ex}) = 3$, $\mathsf{exr}_r(D_{ex}) = \emptyset$, and $\mathcal{C}_r(D_{ex}) = \{A_1 \sqcap A_2 \sqcap A_3\}$.

Finally, if $\mathsf{min}_r(E) = 0$, i.e., there exists no existential restriction subsuming E, then no existential restriction can occur on the top-level of a common subsumer of C and D. Therefore, we set $\mathcal{C}_r(E) := \emptyset$.

Given $\mathcal{C}_r(C)$ and $\mathcal{C}_r(D)$, the lcs of each pair $C' \in \mathcal{C}_r(C)$ and $D' \in \mathcal{C}_r(D)$ gives rise to an existential restriction on the top-level of the lcs of C and D. This is because $\exists r.C'$ and $\exists r.D'$ are exisistential restrictions induced by C and D, respectively, and hence, the lcs of C' and D' is induced by the lcs of C and D. In our example, we obtain $\mathsf{exr}_r(\mathsf{c\text{-}lcs}(C_{ex}, D_{ex})) = \{A_1 \sqcap A_2 \sqcap A_3, A_1 \sqcap A_3\}$, giving rise to the existential restrictions $\exists r.(A_1 \sqcap A_2 \sqcap A_3)$ and $\exists r.(A_1 \sqcap A_3)$ (where the latter one can be omitted).

It remains to comment on $\forall r.\mathsf{c\text{-}lcs}(C_r^*, D_r^*)$. Intuitively, $\forall r.E_r^*$ is the most specific value restriction subsuming E for $E \in \{C, D\}$. (Thus, $\forall r.\mathsf{c\text{-}lcs}(C_r^*, D_r^*)$ is the most specific value restriction subsuming both C and D.) If E has no induced value restriction, E_r^* coincides with $\mathsf{val}_r(E)$. This is the case for D_{ex}, where $(D_{ex})_r^* = \mathsf{val}_r(D_{ex}) = A_1 \sqcap A_2 \sqcap A_3$. If $\mathsf{max}_r(E) = 0$, the fact that $E \sqsubseteq \forall r.\bot$ is made explicit by defining $E_r^* := \bot$. Finally, if $0 < \kappa_r(E) = \mathsf{max}_r(E)$, then the induced value restriction is again made explicit by recursively computing the lcs of the merged existential restrictions (each conjoined with $\mathsf{val}_r(E)$). In our running example, this case is illustrated by C_{ex}: We have that $\mathsf{val}_r(C_{ex}) = \top$, but since $\kappa_r(C_{ex}) = \mathsf{max}_r(C_{ex}) = 2$, we obtain $(C_{ex})_r^* = A_1 \sqcap A_3$, which is the recursively computed lcs of $\mathsf{exr}_r(C_{ex})^* = \mathsf{exr}_r(C_{ex}^1) \cup \mathsf{exr}_r(C_{ex}^2)$ (see Section 3).

The following lemma is easy to prove by induction on the role depth of the concept descriptions.

LEMMA 5.1. *The algorithm* c-lcs *depicted in Figure 1 always terminates and satisfies that* $\mathsf{depth}(\mathsf{c\text{-}lcs}(C, D)) \leq \max(\mathsf{depth}(C), \mathsf{depth}(D))$.

The following theorem states correctness of the lcs algorithm depicted in Figure 1. Its proof is by induction on the maximum role depth of the input concept descriptions and makes heavy use of the structural characterization of subsumption given in Theorem 4.1.

THEOREM 5.2. *Let* C, D *be* $\mathcal{ALEN}$*-concept descriptions. Then, the lcs of* C *and* D *exists and is computed by the algorithm* c-lcs, *i.e.,* $\mathsf{c\text{-}lcs}(C, D) \equiv \mathsf{lcs}(C, D)$.

PROOF. Since by Lemma 5.1, $\mathsf{c\text{-}lcs}$ always terminates, the existence of the lcs follows from $\mathsf{c\text{-}lcs}(C,D) \equiv \mathsf{lcs}(C,D)$. To show $\mathsf{c\text{-}lcs}(C,D) \equiv \mathsf{lcs}(C,D)$ it suffices to show that

i) $C \sqsubseteq \mathsf{c\text{-}lcs}(C,D)$ and $D \sqsubseteq \mathsf{c\text{-}lcs}(C,D)$, and

ii) for all E with $C, D \sqsubseteq E$, it follows $\mathsf{c\text{-}lcs}(C,D) \sqsubseteq E$.

The proof is by induction on $max\{\mathsf{depth}(C), \mathsf{depth}(D)\}$. Let L:=$\mathsf{c\text{-}lcs}(C,D)$.

i): We show $C \sqsubseteq \mathsf{c\text{-}lcs}(C,D)$. The case for D is analogous. If $C \sqsubseteq D$, $C \equiv \bot$ or $L \equiv \top$, nothing has to be shown. Assume $C \not\equiv \bot$ and $L \not\equiv \top$. We show that C and L satisfy the conditions (1)–(5) in Theorem 4.1.

By definition of L,

(1) $\mathsf{prim}(L) \subseteq \mathsf{prim}(C)$,

(2) $\mathsf{min}_r(C) \geq \mathsf{min}_r(L)$, and

(3) $\mathsf{max}_r(C) \leq \mathsf{max}_r(L)$.

(5) Assume $\mathsf{val}_r(L) \not\equiv \top$. If $\mathsf{max}_r(C) = 0$, nothing has to be shown. If $\kappa_r(C) < \mathsf{max}_r(C)$, then $C_r^* = \mathsf{val}_r(C)$. By induction, $\mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(L)$. Assume $0 < \kappa_r(C) = \mathsf{max}_r(C)$. Then, $C_r^* = \mathsf{c\text{-}lcs}(\{\mathsf{val}_r(C) \sqcap C' \mid C' \in \mathsf{exr}_r(C)^*\})$. By induction, $\mathsf{val}_r(C) \sqcap C' \sqsubseteq C_r^*$ for all $C' \in \mathsf{exr}_r(C)^*$, and $C_r^* \sqsubseteq \mathsf{val}_r(L) = \mathsf{c\text{-}lcs}(C_r^*.D_r^*)$. This implies $\mathsf{val}_r(C) \sqcap C' \sqsubseteq \mathsf{val}_r(L)$ for all $C' \in \mathsf{exr}_r(C)^*$.

(4) Let $L' \in \mathsf{exr}_r(L)$. This implies $\mathcal{C}_r(C) \neq \emptyset$ and $\mathcal{C}_r(D) \neq \emptyset$.

Assume $\mathsf{exr}_r(C) = \emptyset$. Then $\mathcal{C}_r(C) \neq \emptyset$ implies $\mathcal{C}_r(C) = \{\mathsf{val}_r(C)\}$, and hence, $\mathsf{min}_r(C) \geq 1$. By induction, $\mathsf{val}_r(C) \sqsubseteq L'$.

Assume $\mathsf{exr}_r(C) \neq \emptyset$. Let L' be the result of $\mathsf{c\text{-}lcs}$ applied to $\mathsf{c\text{-}lcs}(\{C_1 \sqcap \mathsf{val}_r(C), \ldots, C_k \sqcap \mathsf{val}_r(C)\}) \in \mathcal{C}_r(C)$ and some $D \in \mathcal{C}_r(D)$. Then, for each $\pi \in \Gamma_r(C)$, there exists $i \in \{1, \ldots, k\}$ such that $C_i \in \mathsf{exr}_r(C)^\pi$. Now, by the induction hypthesis, we obtain $C_i \sqcap \mathsf{val}_r(C) \sqsubseteq L'$.

Thus, we have shown $C \sqsubseteq L$.

ii): Let E be an $\mathcal{ALEN}$-concept description with $C, D \sqsubseteq E$. If $C \sqsubseteq D$ or $D \sqsubseteq C$, we get $\mathsf{c\text{-}lcs}(C,D) \sqsubseteq E$. Assume $C \not\sqsubseteq D$ and $D \not\sqsubseteq C$. In particular, this implies $C \not\equiv \bot$ and $D \not\equiv \bot$, and hence, $L \not\equiv \bot$.

If $E \equiv \top$, nothing has to be shown. Assume $E \not\equiv \top$. We show that L and E satisfy the conditions (1)–(5) in Theorem 4.1. By assumption, $C \not\equiv \bot$,

$D \not\equiv \bot$, and $E \not\equiv \top$. Thus, by Theorem 4.1, $C \sqsubseteq E$ ($D \sqsubseteq E$) implies that C (D) and E satisfy the conditions (1)–(5) in Theorem 4.1. In the following, we will use this fact without referring to Theorem 4.1 each time.

(1) Since $\mathsf{prim}(E) \subseteq \mathsf{prim}(C)$ and $\mathsf{prim}(E) \subseteq \mathsf{prim}(D)$, we can conclude that $\mathsf{prim}(E) \subseteq \mathsf{prim}(C) \cap \mathsf{prim}(D) = \mathsf{prim}(L)$.

(2) Since $\mathsf{max}_r(C) \leq \mathsf{max}_r(E)$ and $\mathsf{max}_r(D) \leq \mathsf{max}_r(E)$, it follows that $\mathsf{max}_r(L) = min\{\mathsf{max}_r(C), \mathsf{max}_r(D)\} \leq \mathsf{max}_r(E)$.

(3) Since $\mathsf{min}_r(C) \geq \mathsf{min}_r(E)$ and $\mathsf{min}_r(D) \geq \mathsf{min}_r(E)$, we have $\mathsf{min}_r(L) = max\{\mathsf{min}_r(C), \mathsf{min}_r(D)\} \geq \mathsf{min}_r(E)$.

(5) Assume $\mathsf{val}_r(E) \not\equiv \top$. If $\mathsf{max}_r(L) = 0$, nothing has to be shown. According to condition (5) we have to distinguish two cases:

   a) Assume $\kappa_r(L) < \mathsf{max}_r(L)$. We have to show $\mathsf{val}_r(L) \sqsubseteq \mathsf{val}_r(E)$. First note that $\mathsf{depth}(C_r^*) < \mathsf{depth}(C)$ and $\mathsf{depth}(D_r^*) < \mathsf{depth}(D)$ by Lemma 5.1. Now, provided that $C_r^* \sqsubseteq \mathsf{val}_r(E)$ and $D_r^* \sqsubseteq \mathsf{val}_r(E)$, induction yields that $\mathsf{val}_r(L) = \mathsf{c\text{-}lcs}(C_r^*, D_r^*) \sqsubseteq \mathsf{val}_r(E)$.

      It remains to show $C_r^* \sqsubseteq \mathsf{val}_r(E)$. The proof for D_r^* is analogous.

      Assume $\mathsf{max}_r(C) = 0$. Then $C_r^* = \bot$, and hence, $C_r^* \sqsubseteq \mathsf{val}_r(E)$.

      Assume $\kappa_r(C) < \mathsf{max}_r(C)$. Then, $C_r^* = \mathsf{val}_r(C) \sqsubseteq \mathsf{val}_r(E)$, and we obtain $C_r^* \sqsubseteq \mathsf{val}_r(E)$.

      Finally, assume $0 < \kappa_r(C) = \mathsf{max}_r(C)$. Then, $\mathsf{val}_r(C) \sqcap C' \sqsubseteq \mathsf{val}_r(E)$ for all $C' \in \mathsf{exr}_r(C)^*$. Since $\mathsf{depth}(\mathsf{val}_r(C) \sqcap C') < \mathsf{depth}(C)$, by induction and Lemma 5.1 we have $C_r^* = \mathsf{c\text{-}lcs}(\{\mathsf{val}_r(C) \sqcap C' \mid C' \in \mathsf{exr}_r(C)^*\}) \sqsubseteq \mathsf{val}_r(E)$.

   b) Assume $0 < \kappa_r(L) = \mathsf{max}_r(L)$. We have to show $\mathsf{val}_r(L) \sqcap L' \sqsubseteq \mathsf{val}_r(E)$ for all $L' \in \mathsf{exr}_r(L)^*$. As in the previous case we can show $C_r^* \sqsubseteq \mathsf{val}_r(E)$ and $D_r^* \sqsubseteq \mathsf{val}_r(E)$. Now, by induction we can again conclude that $\mathsf{val}_r(L) = \mathsf{c\text{-}lcs}(C_r^*, D_r^*) \sqsubseteq \mathsf{val}_r(E)$. In particular, $\mathsf{val}_r(L) \sqcap L' \sqsubseteq \mathsf{val}_r(E)$.

(4) Let $E' \in \mathsf{exr}_r(E)$. Assume $\mathsf{exr}_r(L) = \emptyset$. This implies $\mathcal{C}_r(C) = \emptyset$ or $\mathcal{C}_r(D) = \emptyset$. W.l.o.g. let $\mathcal{C}_r(C) = \emptyset$. This would imply $\mathsf{min}_r(C) = 0$ and $\mathsf{exr}_r(C) = \emptyset$ in contradiction to Theorem 4.1 applied to C and E. Thus, it is $\mathsf{exr}_r(L) \neq \emptyset$ and we have to show that, for each $\pi \in \Gamma_r(L)$, there exists $L' \in \mathsf{exr}_r(L)^\pi$ with $L' \sqcap \mathsf{val}_r(L) \sqsubseteq E'$.

   Since $C \sqsubseteq E$, it holds

(*a*) $\mathsf{exr}_r(C) = \emptyset$, $\mathsf{min}_r(C) \geq 1$, and $\mathsf{val}_r(C) \sqsubseteq E'$; or

(*b*) for each $\pi \in \Gamma_r(C)$, there exists $C' \in \mathsf{exr}_r(C)^\pi$ with $C' \sqcap \mathsf{val}_r(C) \sqsubseteq E'$;

and since $D \sqsubseteq E$, it holds

(*a′*) $\mathsf{exr}_r(D) = \emptyset$, $\mathsf{min}_r(D) \geq 1$, and $\mathsf{val}_r(D) \sqsubseteq E'$; or

(*b′*) for each $\pi' \in \Gamma_r(D)$, there exists $D' \in \mathsf{exr}_r(D)^{\pi'}$ with $D' \sqcap \mathsf{val}_r(D) \sqsubseteq E'$.

We have to consider the possible combinations of cases from $\{(a), (b)\}$ and $\{(a'), (b')\}$.

Assume (*a*) **and** (*a′*)**:** Then we have that $\Gamma_r(C) = \Gamma_r(D) = \emptyset$, and thus, $\mathcal{C}_r(C) = \{\mathsf{val}_r(C)\}$, $\mathcal{C}_r(D) = \{\mathsf{val}_r(D)\}$, and $\mathsf{exr}_r(L) = \{\mathsf{c\text{-}lcs}(\mathsf{val}_r(C),\ \mathsf{val}_r(D))\}$. Consequently, $C_r^* = \mathsf{val}_r(C)$ and $D_r^* = \mathsf{val}_r(D)$. That is, $\mathsf{val}_r(L) = \mathsf{c\text{-}lcs}(\mathsf{val}_r(C), \mathsf{val}_r(D))$. By induction, we get that $\mathsf{val}_r(L) \sqsubseteq E'$, and hence, for all $\pi \in \Gamma_r(L)$, we can choose an arbitrary $L' \in \mathsf{exr}_r(L)^\pi$ and get $L' \sqcap \mathsf{val}_r(L) \sqsubseteq E'$.

Assume (*a*) **and** (*b′*)**:** Then there exists a tuple $(D_1, \ldots, D_\ell)$ in the set $\times_{\pi' \in \Gamma_r(D)} \mathsf{exr}_r(D)^{\pi'}$ such that $D_j \sqcap \mathsf{val}_r(D) \sqsubseteq E'$ for all $1 \leq j \leq \ell$, and

$$L_0 := \mathsf{c\text{-}lcs}(\mathsf{val}_r(C), \mathsf{c\text{-}lcs}(\{D_j \sqcap \mathsf{val}_r(D) \mid 1 \leq j \leq \ell\})) \in \mathsf{exr}_r(L).$$

By induction, $L_0 \sqsubseteq E'$.
Now, for each $\pi \in \Gamma_r(L)$, there exists $L' \in \mathsf{exr}_r(L)^\pi$ such that $L' \sqsubseteq L_0 \sqsubseteq E'$; in particular, $L' \sqcap \mathsf{val}_r(L) \sqsubseteq E'$.

Assume (*b*) **and** (*a′*)**:** Symmetric to the previous case.

Assume (*b*) **and** (*b′*)**:** As in the previous two cases, there exists a tuple $(D_1, \ldots, D_\ell) \in \times_{\pi' \in \Gamma_r(D)} \mathsf{exr}_r(D)^{\pi'}$ such that $D_j \sqcap \mathsf{val}_r(D) \sqsubseteq E'$ for all $1 \leq j \leq \ell$. In addition, there exists $(C_1, \ldots, C_k) \in \times_{\pi \in \Gamma_r(C)} \mathsf{exr}_r(C)^\pi$ such that $C_i \sqcap \mathsf{val}_r(C) \sqsubseteq E'$ for all $1 \leq i \leq k$. Then, with

$$\begin{array}{lll} L_0 & := & \mathsf{c\text{-}lcs}(\quad \mathsf{c\text{-}lcs}(\{C_i \sqcap \mathsf{val}_r(C) \mid 1 \leq i \leq k\}), \\ & & \qquad\qquad \mathsf{c\text{-}lcs}(\{D_j \sqcap \mathsf{val}_r(D) \mid 1 \leq j \leq \ell\}) \\ & &) \end{array}$$

we have that $L_0 \in \mathsf{exr}_r(L)$, and by induction, $L_0 \sqsubseteq E'$. As above, we get that, for each $\pi \in \Gamma_r(L)$, there exists $L' \in \mathsf{exr}_r(L)^\pi$ such that $L' \sqsubseteq L_0$, and hence, $L' \sqcap \mathsf{val}_r(L) \sqsubseteq E'$.

This completes the proof of Theorem 5.2. ■

6. Complexity of Computing the LCS

In this section, we study the complexity of computing the lcs in $\mathcal{ALEN}$. We first consider the size of the lcs and then investigate the complexity of the algorithm presented in the previous section.

6.1. The Size of the LCS

For sublanguages of $\mathcal{ALEN}$, the following is known about the size of the lcs.

THEOREM 6.1. *1. For $\mathcal{ALN}$ and sublanguages thereof, the size of the lcs of a sequence of concept descriptions is polynomially bounded in the size of the input descriptions. (This immediately follows from results shown in [6]).*

2. For $\mathcal{EL}$, the size of the lcs of two concept descriptions is polynomially bounded in the size of the input descriptions. It may grow exponentially for a sequence of concept descriptions [7].

3. For $\mathcal{ALE}$, the size of the lcs of a pair or a sequence of concept descriptions may grow exponentially in the size of the input descriptions [7]. This is also true for $\mathcal{FLE}$, the sublanguage of $\mathcal{ALE}$ without negation.

All above mentioned bounds are tight.

We note that for the above DLs if the size of the lcs can be bounded by a polynomial, then it can also be computed in polynomial time. In the other cases, exponential time algorithms exist [6, 7].

Let us recall the example for $\mathcal{ALE}/\mathcal{FLE}$-concept descriptions showing the exponential blow-up of the lcs first presented in [7]. Let $D_0 = \top$ and $D_i = \exists r.(P \sqcap Q \sqcap D_{i-1})$, and let $C_0 = \top$ and $C_i = \exists r.P \sqcap \exists r.Q \sqcap \forall r.C_{i-1}$. Then it is easy to see that the lcs of C_n and D_n in $\mathcal{ALE}$ ($\mathcal{FLE}$) is equivalent to E_n where $E_0 = \top$ and $E_n = \exists r.(P \sqcap E_{n-1}) \sqcap \exists r.(Q \sqcap E_{n-1})$. Represented as a tree, E_n is the full binary tree of depth n where the edges are labeled with r and the nodes are labeled with P and Q, respectively. Thus, the size of E_n is exponential in n and it is easy to verify that there does not exist a more concise representation of the lcs of C_n and D_n. As shown in [9], there are examples of $\mathcal{ALE}$-concept descriptions for which the lcs grows exponentially even if a background terminology may be used to represent the lcs.

Using the lcs algorithm presented in Section 5, one verifies, analogously to $\mathcal{ALE}$, that the lcs in $\mathcal{ALEN}$ of C_n and D_n is E_n and that there does not exist a more compact representation of E_n in $\mathcal{ALEN}$. As a result we obtain:

COROLLARY 6.2. *The size of the lcs in $\mathcal{ALEN}$ of two or more $\mathcal{ALEN}$-concept descriptions may grow exponentially.*

It is an open problem whether this is an upper bound.

6.2. The Complexity of the LCS Algorithm

In order to obtain a complexity upper bound for computing the lcs in $\mathcal{ALEN}$, we first slightly modify the algorithm presented in Section 5, to facilitate the analysis, and then study its complexity.

The modified lcs algorithm: When computing $\mathsf{c\text{-}lcs}(C', D')$ (see the definition of $\mathsf{c\text{-}lcs}(C, D)$ in Figure 1), C' and D' are the result of applying $\mathsf{c\text{-}lcs}$ to sets, say S_C and S_D, of the form $\{C_1 \sqcap \mathsf{val}_r(C), \ldots, C_{n(C)} \sqcap \mathsf{val}_r(C)\}$ and $\{D_1 \sqcap \mathsf{val}_r(D), \ldots, D_{n(D)} \sqcap \mathsf{val}_r(D)\}$ as specified in $\mathcal{C}_r(C)$ and $\mathcal{C}_r(D)$, respectively. However, using the fact that $lcs(lcs(C, D), E) \equiv lcs(C, D, E)$, we can omit the application of $\mathsf{c\text{-}lcs}$ to S_C and S_D, respectively, and instead of computing $\mathsf{c\text{-}lcs}(C', D')$ compute $\mathsf{c\text{-}lcs}(S_C \cup S_D)$. In the same way, the concept descriptions C_r^* and D_r^* can be replaced by sets of concept descriptions. Throughout the remainder of this section, whenever we refer to $\mathsf{c\text{-}lcs}$, we mean the algorithm modified as just explained. It is clear that the modified algorithm still computes the lcs.

Complexity of $\mathsf{c\text{-}lcs}$: We will show that $\mathsf{c\text{-}lcs}$ runs in double exponential time in the size of the input concept descriptions.

We first investigate the size of the sets $\mathsf{c\text{-}lcs}$ is applied to during the recursive computation.

Let E be an $\mathcal{ALEN}$-concept description. It is clear that

- $|\Gamma_r(E)|$ can be bounded exponentially in the size of E,
- $|\mathsf{exr}_r(E)^\pi|$ as well as the size of the concept descriptions in $\mathsf{exr}_r(E)^\pi$ are linearly bounded in the size of E,
- $|\mathsf{exr}_r(E)^*|$ can be bounded exponentially in the size of E, and
- $|\mathcal{C}_r(E)|$ has an upper bound double exponential in the size of E. (This follows from the exponential upper bound for $n(E)$ and the linear upper bound for $|\mathsf{exr}_r(E)^\pi|$.)

Now, let us take a look at $\mathsf{c\text{-}lcs}(C, D)$ when recursively expanding its definition. The expansion of the recursive calls of $\mathsf{c\text{-}lcs}(C, D)$ can be viewed as a labeled tree where every node is labeled by the conjunction of (negated) concept names and number restrictions obtained by evaluating c-lcs at this node, and the edges of the tree are labeled with $\exists r$ and $\forall r$. (Note that every node has exactly one edge labeled $\forall r$, but there may be several edges labeled $\exists r$.) Clearly, the depth of the tree is bounded by $m = max\{\mathsf{depth}(C), \mathsf{depth}(D)\}$.

It is obvious that in every node of this computation tree, c-lcs is only applied to a set of conjunctions of subdescriptions (modulo commutativity and associativity of concept conjunction) of C and D of the form $\mathsf{val}_r(E)$, $E \sqcap \mathsf{val}_r(E)$, or $\perp$ where E itself is a conjunction of subdescriptions of C and D, respectively. Since the number of subdescriptions of an $\mathcal{ALEN}$-concept description is linear in the size of the concept description, the size of a conjunction of different such subdescriptions can be bounded polynomially in the size of the given concept description.

Let us consider the cardinality of sets, i.e., the number of arguments c-lcs is applied to in each node. We first concentrate on the invocation of c-lcs for computing existential restrictions. It suffices to only consider the case where $\Gamma_r(E) \neq \emptyset$, since this produces the largest number of concept descriptions. We know that the cardinality of the sets S_C and S_D (see the definition of the modified algorithm above) is bounded exponentially in the size of C and D. Thus, on the first level, for computing $\mathsf{c\text{-}lcs}(S_C \cup S_D)$, c-lcs is only applied to an exponential number of concept descriptions, say $C_1, \dots, C_k$ where k is exponentially bounded in the size of C and D. However, on the second level, an exponential number of sets $S_{C_1}, \dots, S_{C_k}$ need to be considered where every set S_{C_i} may consist of an exponential number of concept descriptions (more precisely, conjunctions of subdescriptions of C and D, as explained above). Hence, the number of arguments of (the n-ary version of) c-lcs on the second level is the product of two functions exponential in the size of C and D, and thus is still exponential. Analogously, on the third level we obtain a product of three exponential functions, which altogether is still exponential in the size of C and D. Iterating this argument, and since the depth of the computation tree is bounded by m (the maximum depth of C and D), the number of arguments to which c-lcs is applied is exponential in the size of C and D on every level of the computation tree. A similar, even simpler argument, works for the invocations of c-lcs for value restrictions.

Summing up, so far we know that every invocation of c-lcs is applied to a set of concept descriptions of cardinality bounded exponentially in the size of C and D. Moreover, every concept description in such a set is a conjunction of subdescriptions of C, D, or $\perp$. Thus, for every node of the computation tree of c-lcs, the (negated) concept names as well as the number restrictions can be computed in exponential time.

It remains to count the number of nodes of this tree. For this purpose, we will first count the number of outgoing edges of every node. Clearly, it suffices to only consider $\exists$-edges, i.e., edges labeled with $\exists r$ since every node of the computation tree has at most one outgoing $\forall$-edge.

As shown before, in every node of the computation tree the number of arguments of c-lcs can be bounded exponentially and every such argument is a conjunction of subdescriptions of C, D, or $\bot$. Consequently, for any argument E of c-lcs on any level in the computation tree, the cardinality of $\mathcal{C}_r(E)$, which in the modified version of c-lcs is a set of sets of concept descriptions, is bounded double exponentially in the size of C and D. Thus, the number of outgoing $\exists$-edges of every node is an exponential product of double exponential functions, and thus, is double exponential.

Having shown that the outdegree of every node in the computation tree is bounded double exponentially in the size of C and D, it follows that the overall number of nodes in the computation tree is bounded double exponentially, too (as the depth of the tree is bounded by m).

Altogether, this proves the following upper bound for computing the lcs in $\mathcal{ALEN}$.

COROLLARY 6.3. *The lcs of (a set of) $\mathcal{ALEN}$-concept descriptions can be computed in double exponential time.*

It is an open problem whether there also exists an exponential time algorithm—as in the case of $\mathcal{ALE}$ [7]—or whether the double exponential blow-up is unavoidable.

7. Conclusion and Outlook

We have presented an algorithm for computing the lcs in $\mathcal{ALEN}$. Its proof of correctness is based on the structural characterization of subsumption introduced in Section 4. In this characterization, we avoided to explicitly use the lcs in order to be able to decouple the characterization of subsumption and the lcs computation. Interleaving these two tasks caused previous work to be incorrect. We have shown that the size of the lcs of two $\mathcal{ALEN}$-concept descriptions can grow exponentially in the size of the input descriptions. We have also established a double exponential upper bound for the size of the lcs and the time needed to compute the lcs. It remains to be investigated whether these bounds are tight.

The lcs algorithm presented here has been implemented and integrated into the graphical ontology editor OILED [35]. In spite of the high worst-case complexity, the implementation behaves quite well in practice and there is still room for optimizations [35].

We expect that the characterization of subsumption in $\mathcal{ALEN}$ presented in this work can serve as the basis for solving other non-standard inferences besides the lcs. In fact, this characterization has already been employed in [10] to compute approximations of $\mathcal{ALCN}$-concept descriptions. It should also be applicable to matching concept descriptions in $\mathcal{ALEN}$.

Another open problem is to extend the results presented here to more expressive DLs, such as DLs with qualified number restrictions. The characterization of subsumption in $\mathcal{ALEN}$ presented here appears to be a good starting point for this purpose.

References

[1] F. Baader. Using Automata Theory for Characterizing the Semantics of Terminological Cycles. *Annals of Mathematics and Artificial Intelligence*, 18(2–4):175–219, 1996.

[2] F. Baader. Computing the least common subsumer in the description logic $\mathcal{EL}$ w.r.t. terminological cycles with descriptive semantics. In *Proceedings of the 11th International Conference on Conceptual Structures (ICCS 2003)*, volume 2746 of *Lecture Notes in Artificial Intelligence*, pages 117–130. Springer-Verlag, 2003.

[3] F. Baader. Least common subsumers and most specific concepts in a description logic with existential restrictions and terminological cycles. In G. Gottlob and T. Walsh, editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003)*, pages 319–324. Morgan Kaufmann, 2003.

[4] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P.F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[5] F. Baader and R. Küsters. Computing the Least Common Subsumer and the Most Specific Concept in the Presence of Cyclic $\mathcal{ALN}$-Concept Descriptions. In O. Herzog and A. Günter, editors, *Proceedings of the 22nd Annual German Conference on Artificial Intelligence, KI-98*, volume 1504 of *Lecture Notes in Computer Science*, pages 129–140, Bremen, Germany, 1998. Springer–Verlag.

[6] F. Baader, R. Küsters, A. Borgida, and D. McGuinness. Matching in Description Logics. *Journal of Logic and Computation*, 9(3):411–447, 1999.

[7] F. Baader, R. Küsters, and R. Molitor. Computing Least Common Subsumers in Description Logics with Existential Restrictions. In T. Dean, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 96–101, Stockholm, Sweden, 1999. Morgan Kaufmann Publishers.

[8] F. Baader, B. Sertkaya, and A.-Y. Turhan. Computing the least common subsumer w.r.t. a background terminology. In José Júlio Alferes and João Alexandre Leite, editors, *Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA 2004)*, volume 3229 of *Lecture Notes in Computer Science*, pages 400–412. Springer-Verlag, 2004.

[9] F. Baader and A.-Y. Turhan. On the Problem of Computing Small Representations of Least Common Subsumers. In *Proceedings of the 25th German Conference on Artificial Intelligence (KI 2002)*, Lecture Notes in Artificial Intelligence, pages 99–113, Aachen, Germany, 2002. Springer–Verlag.

[10] S. Brandt, R. Küsters, and A.-Y. Turhan. Approximating $\mathcal{ALEN}$-Concept Descriptions. In *Proceedings of the International Workshop on Description Logics 2002 (DL 2002)*, 2002. Available from http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/.

[11] S. Brandt, R. Küsters, and A.-Y. Turhan. Approximation and Difference in Description Logics. In D. Fensel, F. Guinchiglia, D. McGuiness, and M. A. Williams, editors, *Proceedings of the Eight International Conference on Knowledge Representation and Reasoning (KR2002)*, pages 203–214. Morgan Kaufmann Publishers, 2002.

[12] S. Brandt and A.-Y. Turhan. Using Non-standard Inferences in Description Logics — what does it buy me? In *Proceedings of the KI-2001 Workshop on Applications of Description Logics (KIDLWS'01)*, number 44 in CEUR-WS, Vienna, Austria, September 2001. RWTH Aachen. Proceedings online available from http://SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-44/.

[13] D. Calvanese and G. De Giacomo. Expressive Description Logics. In *[4]*, pages 178–218. 2003.

[14] W. W. Cohen and H. Hirsh. Learnability of description logics with equality constraints. *Machine Learning*, 17(2/3):169–199, 1994.

[15] W. W. Cohen and H. Hirsh. Learning the CLASSIC Description Logic: Theoretical and Experimental Results. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*, pages 121–133, Bonn, Germany, 1994. Morgan Kaufmann Publishers.

[16] W.W. Cohen, A. Borgida, and H. Hirsh. Computing Least Common Subsumers in Description Logics. In William Swartout, editor, *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 754–760, San Jose, CA, July 1992. MIT Press.

[17] F.M. Donini. Complexity of Reasoning. In *[4]*, pages 96–136. 2003.

[18] M. Frazier and L. Pitt. Classic Learning. *Machine Learning Journal*, 25:151–193, 1996.

[19] V. Haarslev and R. Möller. RACER system description. In R. Gor , A. Leitsch, and T. Nipkow, editors, *First International Joint Conference on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Computer Science*, pages 701–706. Springer, 2001.

[20] E. Hemaspaandra. The complexity of poor man's logic. In J. Gerbrandy, M. Marx, M. de Rijke, and Y. Venema, editors, *Essays Dedicated to Johan van Benthem on the Occasion of his 50th birthday*. Amsterdam University Press, 1999.

[21] B. Hollunder and F. Baader. Qualifying Number Restrictions in Concept Languages. In J.F. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, KR-91*, pages 335–346, Boston (USA), 1991. Morgan Kaufmann Publishers.

[22] I. Horrocks. The FaCT system. In *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (Tableaux'98)*, volume 1397 of *Lecture Notes in Artificial Intelligence*, pages 307–312, Berlin, 1998. Springer-Verlag.

[23] I. Horrocks. Using an Expressive Description Logic: FaCT or Fiction? In A.G. Cohn, L. Schubert, and S.C. Shapiro, editors, *Proceedings of*

the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), pages 636–647, Trento, Italy, 1998. Morgan Kaufmann Publishers.

[24] I. Horrocks. Implementation and optimization techniques. In *[4]*, pages 306–346. 2003.

[25] I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[26] R. Küsters. *Non-Standard Inferences in Description Logics*, volume 2100 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 2001. Ph.D. thesis.

[27] R. Küsters and A. Borgida. What's in an Attribute? Consequences for the Least Common Subsumer. *Journal of Artificial Intelligence Research*, 14:167–203, 2001.

[28] R. Küsters and R. Molitor. Computing Least Common Subsumers in $\mathcal{ALEN}$. In B. Nebel, editor, *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pages 219–224, Seattle, USA, 2001. Morgan Kaufmann Publishers.

[29] T. Mantay. Computing Least Common Subsumers in Expressive Description Logics. In N.Y. Foo, editor, *Advanced Topics in Artificial Intelligence, 12th Australian Joint Conference on Artificial Intelligence (AI'99)*, volume 1747 of *Lecture Notes in Computer Science*, pages 218–230, Sydney,Australia, 1999. Springer-Verlag.

[30] T. Mantay, R. Möller, and A. Kaplunova. Computing Probabilistic Least Common Subsumers in Description Logics. In W. Burgard, T. Christaller, and A.B. Cremers, editors, *Advances in Artificial Intelligence, 23rd Annual German Conference on Artificial Intelligence (KI-99)*, number 1701 in Lecture Notes in Computer Science, pages 89–100, Bonn, Germany, 1999. Springer–Verlag.

[31] R. Möller and V. Haarslev. Description Logic Systems. In *[4]*, pages 282–305. 2003.

[32] R. Möller, V. Haarslev, and B. Neumann. Semantics-Based Information Retrieval. In *Proceedings of the International Conference on Information Technology and Knowledge Systems (IT&KNOWS-98)*, pages 48–61, Vienna, Budapest, 1998.

[33] B. Nebel. Terminological cycles: Semantics and computational properties. In J. Sowa, editor, *Formal Aspects of Semantic Networks*, pages 331–361. Morgan Kaufmann Publishers, San Mateo, 1991.

[34] U. Sattler. *Terminological Knowledge Representation Systems in a Process Engineering Application.* PhD thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany, 1998.

[35] A.-Y. Turhan and C. Kissig. SONIC—non-standard inferences go OILED. In D. Basin and M. Rusinowitch, editors, *Proceedings of the 2nd International Joint Conference on Automated Reasoning (IJCAR 2004)*, volume 3097 of *Lecture Notes in Artificial Intelligence*, pages 321–325. Springer, 2004.

[36] L. von Wedel and W. Marquardt. ROME: A Repository to Support the Integration of Models over the Lifecycle of Model-based Engineering Processes. In *Proceedings of the 10th European Symposium on Computer Aided Process Engineering (ESCAPE-10)*, 2000.

RALF KÜSTERS
Christian-Albrechts-Universität zu Kiel
Institut für Informatik
24098 Kiel, Germany
`kuesters@ti.informatik.uni-kiel.de`

RALF MOLITOR
Swiss Life
P.O. Box
General-Guisan-Quai 40
CH-8022 Zürich, Switzerland

Input: two $\mathcal{ALEN}$-concept descriptions C and D
(analogously for a sequence of concept descriptions)
Output:

If $C \sqsubseteq D$, then $\mathsf{c\text{-}lcs}(C,D) := D$, and if $D \sqsubseteq C$, then $\mathsf{c\text{-}lcs}(C,D) := C$.
Otherwise, let $\Gamma_r(C) = \{\pi_1^C, \ldots, \pi_{n(C)}^C\}$ and $\Gamma_r(D) = \{\pi_1^D, \ldots, \pi_{n(D)}^D\}$, and define for $E \in \{C, D\}$

$$\mathcal{C}_r(E) \quad := \quad \begin{cases} \mathcal{S}_r(E), & \Gamma_r(E) \neq \{\emptyset\}, \\ \{\mathsf{val}_r(E)\}, & \Gamma_r(E) = \{\emptyset\} \wedge \mathsf{min}_r(E) \geq 1, \\ \emptyset, & \text{otherwise (i.e., } \mathsf{min}_r(E) = 0) \end{cases}$$

with

$$\mathcal{S}_r(E){:=}\{\mathsf{c\text{-}lcs}(\{E_1 \sqcap \mathsf{val}_r(E), \ldots, E_{n(E)} \sqcap \mathsf{val}_r(E)\}) \mid E_i \in \mathsf{exr}_r(E)^{\pi_i^E}, 1{\leq}i{\leq}n(E)\}$$

and let

$$E_r^* \quad := \quad \begin{cases} \mathsf{val}_r(E), & 0 \leq \kappa_r(E) < \mathsf{max}_r(E), \\ \bot, & \mathsf{max}_r(E) = 0, \\ \mathsf{c\text{-}lcs}(\{\mathsf{val}_r(E) \sqcap E' \mid E' \in \mathsf{exr}_r(E)^*\}), & \text{otherwise.} \\ & \text{(i.e., } 0 < \kappa_r(E) = \mathsf{max}_r(E)) \end{cases}$$

Define

$$\begin{aligned} \mathsf{c\text{-}lcs}(C,D){:=}&\textstyle\bigsqcap_{Q \in \mathsf{prim}(C) \cap \mathsf{prim}(D)} Q \sqcap \\ &(\leq max\{\mathsf{max}_r(C), \mathsf{max}_r(D)\}\ r) \sqcap (\geq min\{\mathsf{min}_r(C), \mathsf{min}_r(D)\}\ r) \sqcap \\ &\textstyle\bigsqcap_{C' \in \mathcal{C}_r(C), D' \in \mathcal{C}_r(D)} \exists r.\mathsf{c\text{-}lcs}(C', D') \sqcap \\ &\forall r.\mathsf{c\text{-}lcs}(C_r^*, D_r^*), \end{aligned}$$

where

- $(\leq max\{\mathsf{max}_r(C), \mathsf{max}_r(D)\}\ r)$ is omitted if $\mathsf{max}_r(C) = \infty$ or $\mathsf{max}_r(D) = \infty$, and
- $\bigsqcap_{C' \in \mathcal{C}_r(C), D' \in \mathcal{C}_r(D)} \exists r.\mathsf{c\text{-}lcs}(C', D') := \top$ if $\mathcal{C}_r(C) = \emptyset$ or $\mathcal{C}_r(D) = \emptyset$.

Figure 1. The recursive computation of the lcs in $\mathcal{ALEN}$.