Computing Least Common Subsumers in Description Logics with Existential Restrictions*

Franz Baader, Ralf Küsters, and Ralf Molitor
LuFg Theoretische Informatik, RWTH Aachen
email: {baader,kuesters,molitor}@informatik.rwth-aachen.de

Abstract

Computing the least common subsumer (lcs) is an inference task that can be used to support the "bottom-up" construction of knowledge bases for KR systems based on description logics. Previous work on how to compute the lcs has concentrated on description logics that allow for universal value restrictions, but not for existential restrictions. The main new contribution of this paper is the treatment of description logics with existential restrictions. Our approach for computing the lcs is based on an appropriate representation of concept descriptions by certain trees, and a characterization of subsumption by homomorphisms between these trees. The lcs operation then corresponds to the product operation on trees.

1 Introduction

Knowledge representation systems based on description logics (DL) can be used to describe the knowledge of an application domain in a structured and formally well-understood way. Traditionally, the knowledge base of a DL system is built in a "top-down" fashion by first formalizing the relevant concepts of the domain (its terminology) by *concept descriptions*, i.e., expressions that are built from atomic concepts (unary predicates) and atomic roles (binary predicates) using the concept constructors provided by the DL language. In a second step, the concept descriptions are used to specify properties of objects and individuals occurring in the domain. DL systems provide their users with inference services that support both steps: classification of concepts and of individuals. Classification of concepts determines subconcept/superconcept relationships (called subsumption relationships) between the concepts of a given terminology, and thus allows one to structure the terminology in the form of a subsumption hierarchy. Classification of individuals (or objects) determines whether a given individual is always an instance of a certain concept (i.e., whether this instance relationship is implied by the descriptions of the individual and the concept).

This traditional "top-down" approach for constructing a DL knowledge base is not always adequate, however. On the one hand, it need not be clear from the outset which are the relevant concepts in a particular application. On the other hand, even if it is clear which (intuitive) concepts should be introduced, it is sometimes difficult to come up with formal definitions of these concepts within the available description language. For example, in one of our applications in chemical process engineering [Sattler, 1998; Baader and Sattler, 1996], the process engineers prefer to construct the knowledge base (which consists of descriptions of standard building blocks of process models, such as reactors) in the following "bottom-up" fashion: first, they introduce several "typical" examples of the standard building block as individuals, and then they generalize (the descriptions of) these individuals into a concept description that (i) has all the individuals as instances, and (ii) is the most specific description satisfying property (i). The task of computing a description satisfying (i) and (ii) can be split into two subtasks: computing the most specific concept of a single individual, and computing the least common subsumer of a given finite number of concepts. The *most specific concept* (msc) of an individual b is the most specific concept description C (expressible in the given DL) that has b as an instance, and the *least common subsumer* (lcs) of n concept descriptions $C_1, \ldots, C_n$ is the most specific concept description in the given DL that subsumes $C_1, \ldots, C_n$.

The present paper investigates the second subtask for the sub-language $\mathcal{ALE}$ of the DL employed in our process engineering application. This language allows both for value restrictions and existential restrictions, but not for full negation and disjunction (since the lcs operation is trivial in the presence of disjunction, and thus does not provide useful information). It can, e.g., be used to introduce the concept of a reactor with cooling jacket by the description $\mathsf{Reactor} \sqcap \exists \mathsf{connected\text{-}to}.\mathsf{Cooling\text{-}Jacket} \sqcap \forall \mathsf{functionality}.\neg \mathsf{Vaporize}$, where Vaporize is a primitive concept (i.e., not further defined). Previous work on

*This work was partially supported by the EC Working Group CCL II, the *Studienstiftung des deutschen Volkes*, and the *Deutsche Forschungsgemeinschaft* Grant No. GRK 185/3-98.

how to compute the lcs [Cohen and Hirsh, 1994; Frazier and Pitt, 1996] has concentrated on sub-languages of the DL used by the system CLASSIC [Brachman *et al.*, 1991], which allows (among other constructors) for value restrictions, but not for existential restrictions. Thus, the main new contribution of the present paper is the treatment of existential restrictions.

For didactic reasons, we will start by showing how to compute the lcs in the small language $\mathcal{EL}$, which allows for conjunction and existential restrictions only and extend our treatment in two steps to $\mathcal{FLE}$ by adding value restrictions, and then to $\mathcal{ALE}$ by further adding primitive negation. For all three languages, we proceed in the following manner. First, we introduce an appropriate data structure for representing concept descriptions (so-called description trees), and show that subsumption can be characterized by the existence of homomorphisms between description trees. From this characterization we then deduce that the lcs operation on concept descriptions corresponds to the product operation on description trees, which can easily be computed. We will also comment on the complexity of subsumption and the lcs for the languages under consideration. Because of the space limitation, we cannot give all the technical details. These details as well as complete proofs can be found in [Baader *et al.*, 1998].

2 Preliminaries

Concept descriptions are inductively defined with the help of a set of *constructors*, starting with a set N_C of *primitive concepts* and a set N_R of *primitive roles*. The constructors determine the expressive power of the DL. In this paper, we consider concept descriptions built from the constructors shown in Table 1. In the description logic $\mathcal{EL}$, concept descriptions are formed using the constructors top-concept ($\top$), conjunction ($C \sqcap D$) and existential restriction ($\exists r.C$). The description logic $\mathcal{FLE}$ additionally provides us with value restrictions ($\forall r.C$), and $\mathcal{ALE}$ allows for all the constructors shown in Table 1.

The semantics of a concept description is defined in terms of an *interpretation* $\mathcal{I} = (\Delta, \cdot^I)$. The domain Δ of $\mathcal{I}$ is a non-empty set of individuals and the interpretation function $\cdot^I$ maps each primitive concept $P \in N_C$ to a set $P^I \subseteq \Delta$ and each primitive role $r \in N_R$ to a binary relation $r^I \subseteq \Delta \times \Delta$. The extension of $\cdot^I$ to arbitrary concept descriptions is inductively defined, as shown in the third column of Table 1.

One of the most important traditional inference services provided by DL systems is computing the subsumption hierarchy. The concept description C is *subsumed* by the description D ($C \sqsubseteq D$) iff $C^I \subseteq D^I$ holds for all interpretations $\mathcal{I}$. The concept descriptions C and D are *equivalent* ($C \equiv D$) iff they subsume each other.

In this paper, we are interested in the non-standard inference task of computing the *least common subsumer* (lcs) of concept descriptions. Given $n \geq 2$ concept descriptions $C_1, \ldots, C_n$ in a description logic $\mathcal{L}$, a concept description C of $\mathcal{L}$ is an lcs of $C_1, \ldots, C_n$ (for short, $C = \mathsf{lcs}(C_1, \ldots, C_n)$) iff (i) $C_i \sqsubseteq C$ for all $1 \leq i \leq n$, and (ii) C is the least concept description with this property, i.e., if C' satisfies $C_i \sqsubseteq C'$ for all $1 \leq i \leq n$, then $C \sqsubseteq C'$.

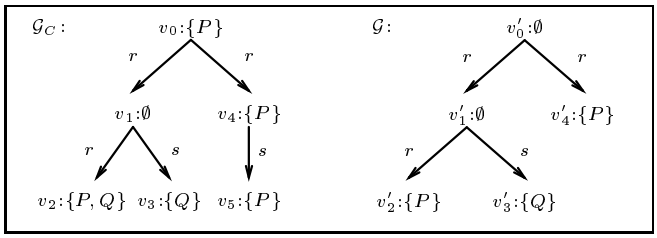

Figure 1: $\mathcal{EL}$-description trees.

Depending on the DL under consideration, the lcs of two or more descriptions need not always exist, but if it exists, then it is unique up to equivalence. In the following, we will show that, for the DLs $\mathcal{EL}$, $\mathcal{FLE}$, and $\mathcal{ALE}$, the lcs always exists and can effectively be computed. We will mostly restrict the attention to the problem of computing the lcs of two concept descriptions, since the lcs of $n > 2$ descriptions can be obtained by iterated application of the binary lcs operation.

3 Getting started – the lcs in $\mathcal{EL}$

As mentioned in the introduction, our method for computing the lcs is based on an appropriate representation of concept descriptions by trees. In the case of the small DL $\mathcal{EL}$, these trees, called *$\mathcal{EL}$-description trees*, are of the form $\mathcal{G} = (V, E, v_0, \ell)$, where $\mathcal{G}$ is a tree with root v_0 whose edges $vrw \in E$ are labeled with primitive roles $r \in N_R$, and whose nodes $v \in V$ are labeled with sets $\ell(v)$ of primitive concepts from N_C. The empty label corresponds to the top-concept.

Intuitively, such a tree is merely a graphical representation of the syntax of the concept description. More formally, every $\mathcal{EL}$-concept description C can be written (modulo equivalence) as $C \equiv P_1 \sqcap \ldots \sqcap P_n \sqcap \exists r_1.C_1 \sqcap \ldots \sqcap \exists r_m.C_m$ with $P_i \in N_C \cup \{\top\}$. This description can now be translated into an $\mathcal{EL}$-description tree $\mathcal{G}_C = (V, E, v_0, \ell)$ as follows. The set of all primitive concepts occurring in the top-level conjunction of C yields the label $\ell(v_0)$ of the root v_0, and each existential restriction $\exists r_i.C_i$ in this conjunction yields an r_i-successor that is the root of the tree corresponding to C_i. For example, the $\mathcal{EL}$-concept description

$$C := P \sqcap \exists r.(\exists r.(P \sqcap Q) \sqcap \exists s.Q) \sqcap \exists r.(P \sqcap \exists s.P)$$

yields the tree $\mathcal{G}_C$ depicted on the left-hand side of Fig. 1.

Conversely, every $\mathcal{EL}$-description tree $\mathcal{G} = (V, E, v_0, \ell)$ can be translated into an $\mathcal{EL}$-concept description $C_\mathcal{G}$. Intuitively, the primitive concepts in the label of v_0 yield the primitive concepts in the top-level conjunction of $C_\mathcal{G}$, and each r-successor v of v_0 yields an existential restriction $\exists r.C$ where C is the $\mathcal{EL}$-concept description obtained by translating the subtree of $\mathcal{G}$ with root v. For a leaf $v \in V$, the empty label is translated into the top-concept. For example, the $\mathcal{EL}$-description tree $\mathcal{G}$ in

name of constructor	Syntax	Semantics	$\mathcal{EL}$	$\mathcal{FLE}$	$\mathcal{ALE}$
primitive concept $P \in N_C$	P	$P^I \subseteq \Delta$	x	x	x
top-concept	$\top$	Δ	x	x	x
conjunction	$C \sqcap D$	$C^I \cap D^I$	x	x	x
existential restriction for $r \in N_R$	$\exists r.C$	$\{x \in \Delta \mid \exists y : (x,y) \in r^I \wedge y \in C^I\}$	x	x	x
value restriction for $r \in N_R$	$\forall r.C$	$\{x \in \Delta \mid \forall y : (x,y) \in r^I \rightarrow y \in C^I\}$		x	x
primitive negation for $P \in N_C$	$\neg P$	$\Delta \setminus P^I$			x
bottom-concept	$\bot$	$\emptyset$			x

Table 1: Syntax and semantics of concept descriptions.

Fig. 1 yields the $\mathcal{EL}$-concept description

$$C_{\mathcal{G}} = \exists r.(\exists r.P \sqcap \exists s.Q) \sqcap \exists r.P.$$

These translations preserve the semantics of concept descriptions in the sense that $C \equiv C_{\mathcal{G}_C}$.

Subsumption in $\mathcal{EL}$ can be characterized using the following notion: a *homomorphism* from an $\mathcal{EL}$-description tree $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ to an $\mathcal{EL}$-description tree $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ is a mapping $\varphi : V_H \longrightarrow V_G$ such that (1) $\varphi(w_0) = v_0$, (2) $\ell_H(v) \subseteq \ell_G(\varphi(v))$ for all $v \in V_H$, and (3) $\varphi(v) r \varphi(w) \in E_G$ for all $vrw \in E_H$.

Theorem 1 *Let C, D be $\mathcal{EL}$-concept descriptions and $\mathcal{G}_C, \mathcal{G}_D$ the corresponding $\mathcal{EL}$-description trees. Then $C \sqsubseteq D$ iff there exists a homomorphism from $\mathcal{G}_D$ to $\mathcal{G}_C$.*

In our example, the $\mathcal{EL}$-concept description $C_{\mathcal{G}}$ subsumes C, because mapping v_i' onto v_i for all $0 \leq i \leq 4$ yields a homomorphism from $\mathcal{G} = \mathcal{G}_{C_{\mathcal{G}}}$ to $\mathcal{G}_C$ (see Fig. 1).

Theorem 1 is a special case of the characterization of subsumption between simple conceptual graphs in [Chein and Mugnier, 1992], and of the characterization of containment of conjunctive queries in [Abiteboul *et al.*, 1995]. In the more general setting of simple conceptual graphs and conjunctive queries, testing for the existence of a homomorphism is an NP-complete problem. In the restricted case of $\mathcal{EL}$-description trees, however, testing for the existence of a homomorphism can be realized in polynomial time [Reyner, 1977; Baader *et al.*, 1998], which shows that subsumption between $\mathcal{EL}$-concept descriptions is a tractable problem.

Least common subsumer in $\mathcal{EL}$

The characterization of subsumption by homomorphisms allows us to characterize the lcs by the product of $\mathcal{EL}$-description trees. The *product* $\mathcal{G} \times \mathcal{H}$ of two $\mathcal{EL}$-description trees $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ and $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ is defined by induction on the depth of the trees. Let $\mathcal{G}(v)$ denote the subtree of $\mathcal{G}$ with root v. We define (v_0, w_0) to be the root of $\mathcal{G} \times \mathcal{H}$, labeled with $\ell_G(v_0) \cap \ell_H(w_0)$. For each r-successor v of v_0 in $\mathcal{G}$ and w of w_0 in $\mathcal{H}$, we obtain an r-successor (v, w) of (v_0, w_0) in $\mathcal{G} \times \mathcal{H}$ that is the root of the product of $\mathcal{G}(v)$ and $\mathcal{H}(w)$.

For example, consider the $\mathcal{EL}$-description tree $\mathcal{G}_C$ (Fig. 1) and the $\mathcal{EL}$-description tree $\mathcal{G}_D$ (Fig. 2), where $\mathcal{G}_D$ corresponds to the $\mathcal{EL}$-concept description $D := \exists r.(P \sqcap \exists r.P \sqcap \exists s.Q)$. The product $\mathcal{G}_C \times \mathcal{G}_D$ is depicted on the right-hand side of Fig. 2.

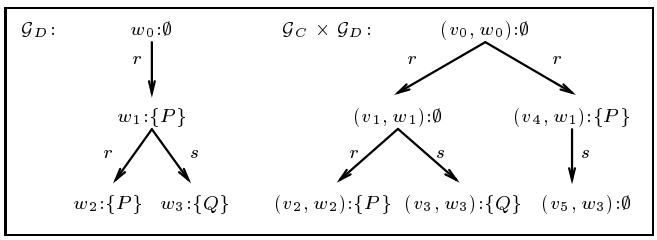

Figure 2: The product of $\mathcal{EL}$-description trees.

Theorem 2 *Let C, D be two $\mathcal{EL}$-concept descriptions and $\mathcal{G}_C$, $\mathcal{G}_D$ the corresponding $\mathcal{EL}$-description trees. Then $C_{\mathcal{G}_C \times \mathcal{G}_D}$ is the lcs of C and D.*

In our example, we thus obtain

$$\mathsf{lcs}(C, D) = \exists r.(\exists r.P \sqcap \exists s.Q) \sqcap \exists r.(P \sqcap \exists s.\top).$$

The size of the lcs of two $\mathcal{EL}$-concept descriptions C, D can be bounded by the size of $\mathcal{G}_C \times \mathcal{G}_D$, which is polynomial in the size of $\mathcal{G}_C$ and $\mathcal{G}_D$. Since the size of the description tree corresponding to a given description is linear in the size of the description, we obtain:

Proposition 3 *The size of the lcs of* two *$\mathcal{EL}$-concept descriptions C, D is polynomial in the size of C and D, and the lcs can be computed in polynomial time.*

In our process engineering application, however, we are interested in the lcs of $n > 2$ concept descriptions $C_1, \ldots, C_n$. This lcs can be obtained from the product $\mathcal{G}_{C_1} \times \cdots \times \mathcal{G}_{C_n}$ of their corresponding $\mathcal{EL}$-description trees. Therefore, the size of the lcs can be bounded by the size of this product. It has turned out [Baader *et al.*, 1998] that, even for the small DL $\mathcal{EL}$, this size cannot be polynomially bounded.

Proposition 4 *The size of the lcs of n $\mathcal{EL}$-concept descriptions $C_1, \ldots, C_n$ of size linear in n may grow exponential in n.*

4 Extending the results to $\mathcal{FLE}$

Our goal is to obtain a characterization of the lcs in $\mathcal{FLE}$ analogous to the one given in Theorem 2 for $\mathcal{EL}$. To achieve this goal, we first extend the notion of a description tree from $\mathcal{EL}$ to $\mathcal{FLE}$. In order to cope with value

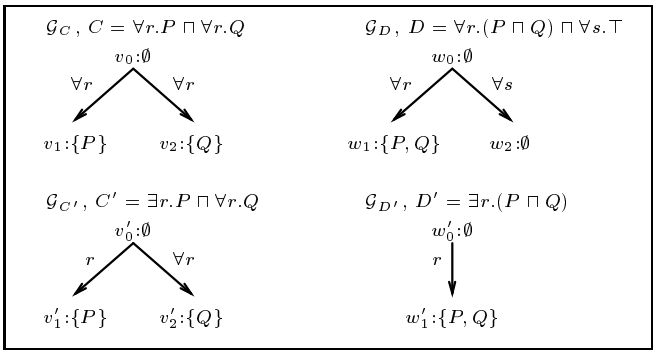

Figure 3: $\mathcal{FLE}$-description trees.

restrictions occurring in $\mathcal{FLE}$-concept descriptions, we allow for two types of edges, namely those labeled with a primitive role $r \in N_R$ (corresponding to existential restrictions of the form $\exists r.C$) and those labeled with $\forall r$ for $r \in N_R$ (corresponding to value restrictions of the form $\forall r.D$). Just as for $\mathcal{EL}$, there is a 1–1 correspondence between $\mathcal{FLE}$-concept descriptions and $\mathcal{FLE}$-description trees.

The notion of a homomorphism also extends to $\mathcal{FLE}$-description trees in a natural way. A homomorphism from an $\mathcal{FLE}$-description tree $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ into an $\mathcal{FLE}$-description tree $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ is a mapping $\varphi : V_H \longrightarrow V_G$ that satisfies the conditions (1)–(3) on homomorphisms between $\mathcal{EL}$-description trees, and additionally (4) $\varphi(v)\,\forall r\,\varphi(w) \in E_G$ for all $v\,\forall r\,w \in E_H$.

However, these straightforward extensions are not sufficient to obtain a sound and complete characterization of subsumption in $\mathcal{FLE}$ based on homomorphisms between $\mathcal{FLE}$-description trees. For example, consider the $\mathcal{FLE}$-concept descriptions and their translations into $\mathcal{FLE}$-description trees depicted in Fig. 3. It is easy to see that $C \sqsubseteq D$ and $C' \sqsubseteq D'$, but there exists neither a homomorphism from $\mathcal{G}_D$ to $\mathcal{G}_C$ nor one from $\mathcal{G}_{D'}$ to $\mathcal{G}_{C'}$.

To avoid these problems, we must normalize the $\mathcal{FLE}$-concept descriptions before translating them into $\mathcal{FLE}$-description trees. The *normal form* of an $\mathcal{FLE}$-concept description C is obtained from C by exhaustively applying the following *normalization rules*:

$$\begin{array}{rcl} \forall r.E \sqcap \forall r.F & \longrightarrow & \forall r.(E \sqcap F) \\ \forall r.E \sqcap \exists r.F & \longrightarrow & \forall r.E \sqcap \exists r.(E \sqcap F) \\ \forall r.\top & \longrightarrow & \top \\ E \sqcap \top & \longrightarrow & E \end{array}$$

Since each normalization rule preserves equivalence, the resulting normalized $\mathcal{FLE}$-concept description is equivalent to the original one. The rules should be read modulo commutativity of conjunction; e.g., $\exists r.E \sqcap \forall r.F$ is also normalized to $\exists r.(E \sqcap F) \sqcap \forall r.F$.

Now, the *$\mathcal{FLE}$-description tree $\mathcal{G}_C$ corresponding to C* is obtained from C by first normalizing C, and then translating the resulting normalized $\mathcal{FLE}$-concept description into a tree. Each $\mathcal{FLE}$-description tree $\mathcal{G} = (V, E, v_0, \ell)$ obtained this way satisfies the following properties [Baader *et al.*, 1998]:

- For each node $v \in V$ and each primitive role $r \in N_R$, v has at most one outgoing edge labeled $\forall r$.
- Let $\{vrw, v\forall rw'\} \subseteq E$, and let C denote the $\mathcal{FLE}$-concept description corresponding to the subtree of $\mathcal{G}$ with root w, and C' the one corresponding to the subtree of $\mathcal{G}$ with root w'. Then $C \sqsubseteq C'$.
- Leaves in $\mathcal{G}$ labeled with the empty set cannot be reached via an edge labeled $\forall r$ for some $r \in N_R$, i.e., $C_\mathcal{G}$ does not contain a subconcept of the form $\forall r.\top$.

The proof of soundness and completeness of the characterization of subsumption in $\mathcal{FLE}$ stated in the next theorem makes heavy use of these properties [Baader *et al.*, 1998].

Theorem 5 *Let C, D be two $\mathcal{FLE}$-concept descriptions and $\mathcal{G}_C$, $\mathcal{G}_D$ their corresponding $\mathcal{FLE}$-description trees. Then $C \sqsubseteq D$ iff there exists a homomorphism from $\mathcal{G}_D$ to $\mathcal{G}_C$.*

It should be noted that there is a close relationship between the normalization rules introduced above and some of the so-called *propagation rules* employed by tableaux-based subsumption algorithms, as e.g. introduced in [Donini *et al.*, 1992]. The main idea underlying our second normalization rule and the propagation rule treating value restrictions is to make the knowledge implicitly given by a conjunction of the form $\forall r.E \sqcap \exists r.F$ explicit by propagating E onto the existential restriction according to the equivalence $\forall r.E \sqcap \exists r.F \equiv \forall r.E \sqcap \exists r.(E \sqcap F)$. As shown in [Donini *et al.*, 1992], this propagation rule may lead to an exponential blow-up of the tableau, and the same is true for our normalization rule. More precisely, applying the normalization rules introduced above to an $\mathcal{FLE}$-concept description C may lead to a normalized concept description, and hence a corresponding $\mathcal{FLE}$-description tree $\mathcal{G}_C$, of size exponential in the size of C. This exponential blow-up cannot be avoided since (i) as for $\mathcal{EL}$, existence of a homomorphism between $\mathcal{FLE}$-description trees can be tested in polynomial time; and (ii) subsumption in $\mathcal{FLE}$ is an NP-complete problem [Donini *et al.*, 1992].

Least common subsumer in $\mathcal{FLE}$

Just as for $\mathcal{EL}$, we can now use the characterization of subsumption in $\mathcal{FLE}$ by homomorphisms to characterize the lcs of two $\mathcal{FLE}$-concept descriptions by the product of $\mathcal{FLE}$-description trees. The *product* $\mathcal{G} \times \mathcal{H}$ of two $\mathcal{FLE}$-description trees $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ and $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ is again defined by induction on the depth of the trees. As before, (v_0, w_0) is the root of $\mathcal{G} \times \mathcal{H}$, and r-successors of (v_0, w_0) are obtained in the same way as for $\mathcal{EL}$. Additionally, we now obtain a $\forall r$-successor (v, w) of (v_0, w_0) in $\mathcal{G} \times \mathcal{H}$ if v is the $\forall r$-successor of v_0 in $\mathcal{G}$ and w the one of w_0 in $\mathcal{H}$, and (v, w) is the root of the product of $\mathcal{G}(v)$ and $\mathcal{H}(w)$.

Theorem 6 *Let C, D be two $\mathcal{FLE}$-concept descriptions and $\mathcal{G}_C$, $\mathcal{G}_D$ their corresponding $\mathcal{FLE}$-description trees. Then $C_{\mathcal{G}_C \times \mathcal{G}_D}$ is the lcs of C and D.*

As mentioned above, $\mathcal{FLE}$ differs from $\mathcal{EL}$ in that the $\mathcal{FLE}$-description tree $\mathcal{G}_C$ corresponding to an $\mathcal{FLE}$-concept description C may be of size exponential in the size of C. Therefore, even for *two* $\mathcal{FLE}$-concept descriptions C, D, the size of their lcs cannot be polynomially bounded by the size of C and D [Baader *et al.*, 1998].

Proposition 7 *The size of the lcs of* two $\mathcal{FLE}$*-concept descriptions* C, D *may be exponential in the size of* C *and* D.

5 Extending the results to $\mathcal{ALE}$

In order to characterize the lcs of two $\mathcal{ALE}$-concept descriptions by the product of description trees, we must adapt the notions description tree, homomorphism, and product appropriately, taking into account the additional constructors primitive negation and bottom-concept.

First, we extend the notion of $\mathcal{FLE}$-description trees to *$\mathcal{ALE}$-description trees* by additionally allowing for negated primitive concepts $\neg P$ and the bottom-concept $\bot$ in the labels of nodes. Again, as for $\mathcal{EL}$ and $\mathcal{FLE}$, there is a 1–1 correspondence between $\mathcal{ALE}$-concept descriptions and $\mathcal{ALE}$-description trees.

Since $\mathcal{ALE}$ is an extension of $\mathcal{FLE}$, and since we are again interested in a characterization of subsumption by homomorphisms, we must normalize $\mathcal{ALE}$-concept descriptions before translating them into their corresponding $\mathcal{ALE}$-description trees. In addition to the normalization rules for $\mathcal{FLE}$, we need three more rules, which deal with the fact that $\mathcal{ALE}$-concept descriptions may contain inconsistent sub-descriptions (i.e., $\bot$ and $P \sqcap \neg P$ for $P \in N_C$):

$$\begin{array}{rcl} P \sqcap \neg P & \longrightarrow & \bot, \text{ for each } P \in N_C \\ \exists r.\bot & \longrightarrow & \bot \\ E \sqcap \bot & \longrightarrow & \bot \end{array}$$

Starting with an $\mathcal{ALE}$-concept description C, the exhaustive application of these rules, together with the rules for $\mathcal{FLE}$, yields an equivalent $\mathcal{ALE}$-concept description in normal form, which is used to construct the *$\mathcal{ALE}$-description tree $\mathcal{G}_C$ corresponding to C*.

In addition to the conditions for $\mathcal{FLE}$-description trees, the $\mathcal{ALE}$-description trees obtained this way satisfy the following condition: if the label of a node contains $\bot$, then its label is $\{\bot\}$ and it is a leaf that cannot be reached by an edge with label $r \in N_R$.

Unfortunately, the straightforward adaptation of the notion of a homomorphism from $\mathcal{FLE}$-description trees to $\mathcal{ALE}$-description trees does not yield a sound and complete characterization of subsumption in $\mathcal{ALE}$. As an example, consider the following $\mathcal{ALE}$-concept descriptions:

$$\begin{array}{rcl} C & := & (\forall r.\exists r.(P \sqcap \neg P)) \sqcap (\exists s.(P \sqcap \exists r.Q)), \\ D & := & (\forall r.(\exists r.P \sqcap \exists r.\neg P)) \sqcap (\exists s.\exists r.Q). \end{array}$$

The description D is already in normal form, and the normal form of C is $C' := \forall r.\bot \sqcap \exists s.(P \sqcap \exists r.Q)$. The corresponding $\mathcal{ALE}$-description trees $\mathcal{G}_C$ and $\mathcal{G}_D$ are depicted in Figure 4.

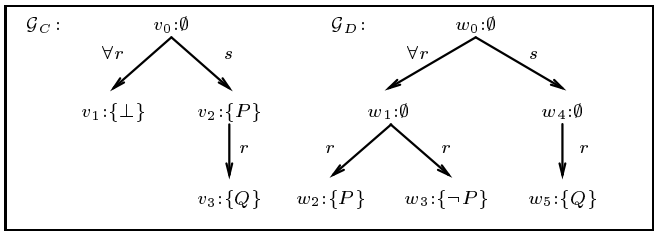

Figure 4: $\mathcal{ALE}$-description trees.

It is easy to see that there does not exist a homomorphism (in the sense of Section 4) from $\mathcal{G}_D$ into $\mathcal{G}_C$, although we have $C \sqsubseteq D$. In particular, the $\mathcal{ALE}$-concept description $\exists r.P \sqcap \exists r.\neg P$ corresponding to the subtree with root w_1 of $\mathcal{G}_D$ subsumes $\bot$, which is the concept description corresponding to the subtree with root v_1 in $\mathcal{G}_C$. Therefore, a homomorphism from $\mathcal{G}_D$ into $\mathcal{G}_C$ should be allowed to map the whole tree corresponding to $\exists r.P \sqcap \exists r.\neg P$, i.e., the nodes w_1, w_2, w_3, onto the tree corresponding to $\bot$, i.e., onto v_1.

A *homomorphism* from an $\mathcal{ALE}$-description tree $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ into an $\mathcal{ALE}$-description tree $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ is a mapping $\varphi : V_H \longrightarrow V_G$ such that (1) $\varphi(w_0) = v_0$, (2) $\ell_H(v) \subseteq \ell_G(\varphi(v))$ or $\ell_G(\varphi(v)) = \{\bot\}$ for all $v \in V_H$, (3) for all $vrw \in E_H$, either $\varphi(v)r\varphi(w) \in E_G$, or $\varphi(v) = \varphi(w)$ and $\ell_G(\varphi(v)) = \{\bot\}$, and (4) for all $v\forall rw \in E_H$, either $\varphi(v)\forall r\varphi(w) \in E_G$, or $\varphi(v) = \varphi(w)$ and $\ell_G(\varphi(v)) = \{\bot\}$.

In our example, if we map w_0 onto v_0; w_1, w_2, and w_3 onto v_1; w_4 onto v_2; and w_5 onto v_3, then the above conditions are satisfied, i.e., this mapping yields a homomorphism from $\mathcal{G}_D$ into $\mathcal{G}_C$.

With this new notion of a homomorphism between $\mathcal{ALE}$-description trees, we can again characterize subsumption in $\mathcal{ALE}$ in a sound and complete way [Baader *et al.*, 1998].[1]

Theorem 8 *Let* C, D *be two* $\mathcal{ALE}$*-concept descriptions and* $\mathcal{G}_C$, $\mathcal{G}_D$ *the corresponding* $\mathcal{ALE}$*-description trees. Then* $C \sqsubseteq D$ *iff there exists a homomorphism from* $\mathcal{G}_D$ *into* $\mathcal{G}_C$.

Least common subsumer in $\mathcal{ALE}$

The definition of the product of $\mathcal{ALE}$-description trees must be adapted to the modified notion of a homomorphism. In particular, this definition must treat leaves with label $\{\bot\}$ in a special manner. In fact, such a leaf corresponds to the bottom-concept, and since $\bot \sqsubseteq C$ for all $\mathcal{ALE}$-concept descriptions C, we have $\mathsf{lcs}(\bot, C) \equiv C$. Thus, our product operation should be defined such that $C_{\mathcal{G}_\bot \times \mathcal{G}_C} \equiv C$.

More precisely, the *product $\mathcal{G} \times \mathcal{H}$ of two $\mathcal{ALE}$-description trees* $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ and $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ is defined as follows. If $\ell_G(v_0) = \{\bot\}$ ($\ell_H(w_0) = \{\bot\}$), then we define $\mathcal{G} \times \mathcal{H}$ by replacing each node w in $\mathcal{H}$ (v in $\mathcal{G}$) by (v_0, w) ((v, w_0)). Otherwise, we

[1] Note that subsumption in $\mathcal{ALE}$ is also an NP-complete problem [Donini *et al.*, 1992].

define $\mathcal{G} \times \mathcal{H}$ by induction on the depth of the trees analogous to the definition of the product of $\mathcal{FLE}$-description trees.

In the example, $\mathcal{G}_C \times \mathcal{G}_D$ can be obtained from $\mathcal{G}_D$ by replacing w_0 by (v_0, w_0), w_i by (v_1, w_i) for $i = 1, 2, 3$, w_4 by (v_2, w_4), and w_5 by (v_3, w_5) (see Fig. 4).

Theorem 9 *Let C, D be two $\mathcal{ALE}$-concept descriptions and $\mathcal{G}_C$, $\mathcal{G}_D$ their corresponding $\mathcal{ALE}$-description trees. Then $C_{\mathcal{G}_C \times \mathcal{G}_D}$ is the lcs of C and D.*

The proof of Proposition 7 also works if we view the $\mathcal{FLE}$-concept descriptions used in this proof as special $\mathcal{ALE}$-concept descriptions. Thus, we have:

Proposition 10 *The size of the lcs of* two *$\mathcal{ALE}$-concept descriptions C, D may be exponential in the size of C, D.*

6 Conclusion and future work

We have described a method for computing the least common subsumer in the description logic $\mathcal{ALE}$. In the worst case, the result of this computation may be exponential in the size of the input descriptions. However, the examples that show this exponential behavior [Baader *et al.*, 1998] are rather artificial, and thus we believe that this complexity will not pose a problem in practice.

Our method depends on the characterization of subsumption by homomorphisms on description trees, because this allows us to construct the lcs as the product of the description trees. For sub-languages of CLASSIC, a similar method has been used to construct the lcs [Cohen and Hirsh, 1994; Frazier and Pitt, 1996], even though the characterization of subsumption (via a structural subsumption algorithm [Borgida and Patel-Schneider, 1994]) is not explicitly given in terms of homomorphisms. The main difference is that these languages do not allow for existential restrictions. The results for simple conceptual graphs (conjunctive queries) mentioned below Theorem 1 characterize subsumption (resp. containment) with the help of homomorphisms, but they do not consider the lcs, and they cannot handle value restrictions.

The language $\mathcal{ALE}$ is expressive enough to be quite useful in our process engineering application. In fact, the descriptions of standard building blocks of process models that we currently represent in our DL system can all be expressed within this language. However, in order to support the "bottom-up" approach for constructing knowledge bases outlined in the introduction, we must also be able to compute the most specific concept for individuals. Unfortunately, the msc need not always exist in $\mathcal{ALE}$. For the DL $\mathcal{ALN}$, it was shown in [Baader and Küsters, 1998] that this problem can be overcome by allowing for cyclic concept descriptions, but $\mathcal{ALN}$ does not allow for existential restrictions. Thus, we must either extend the approach of [Baader and Küsters, 1998] to $\mathcal{ALE}$, or resort to an approximation of the msc, as proposed in [Cohen and Hirsh, 1994]. In the process engineering application, we can also use the lcs operation directly to structure the existing knowledge base. In fact, it has turned out that the subsumption hierarchy obtained from the knowledge base of standard building blocks is rather flat. To obtain a deeper hierarchy (which better supports search and hence reuse of building blocks), we will try to construct intermediate levels of concepts by applying the lcs operation. Of course, this only makes sense if the lcs yields concepts that have an intuitive meaning in the application domain.

References

[Abiteboul *et al.*, 1995] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[Baader and Küsters, 1998] 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 *Proc. of KI'98*, LNCS 1504, 1998.

[Baader and Sattler, 1996] F. Baader and U. Sattler. Knowledge representation in process engineering. In *Proc. of DL'96*, 1996.

[Baader *et al.*, 1998] F. Baader, R. Küsters, and R. Molitor. Computing least common subsumers in description logics with existential restrictions. LTCS-Report 98-09. See http://www-lti.informatik.rwth-aachen.de/Forschung/Papers.html.

[Borgida and Patel-Schneider, 1994] A. Borgida and P. Patel-Schneider. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *J. of AI Research*, 1, 1994.

[Brachman *et al.*, 1991] R. J. Brachman, D. McGuinness, P. Patel-Schneider, L. Resnick, and A. Borgida. Living with CLASSIC: When and how to use a KL-ONE-like language. In *Principles of Semantic Networks*. Morgan Kaufmann, 1991.

[Chein and Mugnier, 1992] M. Chein and M. Mugnier. Conceptual graphs: Fundamental notions. *Revue d'Intelligence Artificielle*, 6(4), 1992.

[Cohen and Hirsh, 1994] W. W. Cohen and H. Hirsh. Learning the CLASSIC description logic: Theoretical and experimental results. In *Proc. of KR'94*. Morgan Kaufmann, 1994.

[Donini *et al.*, 1992] F.M. Donini, M. Lenzerini, D. Nardi, B. Hollunder, W. Nutt, and A.M. Spaccamela. The complexity of existential quantification in concept languages. *Journal of Artificial Intelligence*, 52, 1992.

[Frazier and Pitt, 1996] M. Frazier and L. Pitt. CLASSIC learning. *Machine Learning*, 25, 1996.

[Reyner, 1977] S. Reyner. An analysis of a good algorithm for the subtree problem. *SIAM Journal of Computing*, 6(4), 1977.

[Sattler, 1998] U. Sattler. *Terminological Knowledge Representation Systems in a Process Engineering Application*. PhD thesis, RWTH Aachen, 1998.