TECHNISCHE
UNIVERSITÄT
DRESDEN

Dresden University of Technology
Institute for Theoretical Computer Science
Chair for Automata Theory

LTCS–Report

A Graph-Theoretic Generalization of the Least Common Subsumer and the Most Specific Concept in the Description Logic $\mathcal{EL}$

Franz Baader

LTCS-Report 04-02

Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
http://lat.inf.tu-dresden.de

Hans-Grundig-Str. 25
01062 Dresden
Germany

A Graph-Theoretic Generalization of the Least Common Subsumer and the Most Specific Concept in the Description Logic $\mathcal{EL}$

Franz Baader*
Theoretical Computer Science
Dresden University of Technology
D-01062 Dresden, Germany
e-mail: baader@inf.tu-dresden.de

Abstract

In two previous papers we have investigates the problem of computing the least common subsumer (lcs) and the most specific concept (msc) for the description logic $\mathcal{EL}$ in the presence of terminological cycles that are interpreted with descriptive semantics, which is the usual first-order semantics for description logics. In this setting, neither the lcs nor the msc needs to exist. We were able to characterize the cases in which the lcs/msc exists, but it was not clear whether this characterization yields decidability of the existence problem.

In the present paper, we develop a common graph-theoretic generalization of these characterizations, and show that the resulting property is indeed decidable, thus yielding decidability of the existence of the lcs and the msc. This is achieved by expressing the property in monadic second-order logic on infinite trees. We also show that, if it exists, then the lcs/msc can be computed in polynomial time.

1 Introduction

Early description logic (DL) systems allowed the use of value restrictions ($\forall r.C$), but not of existential restrictions ($\exists r.C$). Thus, one could express that all children are male using the value restriction $\forall\mathsf{child.Male}$, but not that someone has a son

*Partially supported by the DFG under grant BA 1122/4-3 and by National ICT Australia Limited, Canberra Research Lab.

using the existential restriction $\exists \mathsf{child}.\mathsf{Male}$. The main reason was that, when clarifying the logical status of property arcs in semantic networks and slots in frames, the decision was taken that arcs/slots should be read as value restrictions (see, e.g., [11]). Once one considers more expressive DLs allowing for full negation, existential restrictions come in as the dual of value restrictions [14]. Thus, for historical reasons, DLs that allow for existential, but not for value restrictions, were until recently mostly unexplored.

The recent interest in such DLs has at least two reasons. First, there are indeed applications where DLs without value restrictions appear to be sufficient. For example, SNOMED, the Systematized Nomenclature of Medicine [16, 15] employs the DL $\mathcal{EL}$, which allows for conjunctions, existential restrictions, and the top concept. Second, non-standard inferences in DLs [10], like computing the least common subsumer, often make sense only for DLs that do not allow for full negation. Thus, the decision of whether to use DLs with value restrictions or with existential restrictions becomes again relevant.

Non-standard inferences were introduced to support building and maintaining large DL knowledge bases. For example, computing the most specific concept (msc) of an individual and the least common subsumer (lcs) of concepts can be used in the bottom-up construction of description logic knowledge bases. Instead of defining the relevant concepts of an application domain from scratch, this methodology allows the user to give typical examples of individuals belonging to the concept to be defined. These individuals are then generalized to a concept by first computing the most specific concept of each individual (i.e., the least concept description in the available description language that has this individual as an instance), and then computing the least common subsumer of these concepts (i.e., the least concept description in the available description language that subsumes all these concepts). The knowledge engineer can then use the computed concept as a starting point for the concept definition.

The most specific concept of a given individual need not exist in languages allowing for existential restrictions or number restrictions. For the DL $\mathcal{ALN}$ (which allows for conjunctions, value restrictions, and number restrictions), it was shown in [6] that the most specific concept always exists if one adds cyclic concept definitions with greatest fixpoint semantics. If one wants to use this approach for the bottom-up construction of knowledge bases, then one must also be able to solve the standard inferences (the subsumption and the instance problem) and to compute the least common subsumer and the most specific concept in the presence of cyclic concept definitions. Thus, in order to adapt the approach employed in [6] also to the DL $\mathcal{EL}$, the impact on both standard and non-standard inferences of cyclic definitions in this DL had to be investigated first.

This investigation was carried out in a series of papers [5, 4, 2, 3] that gives an almost complete picture of the computational properties of the above mentioned standard and non-standard inferences (the subsumption and the instance problem as well as the problem of computing the lcs and the msc) in $\mathcal{EL}$ with cyclic concept definitions. Cyclic definitions in $\mathcal{EL}$ can either be interpreted with greatest fixpoint (gfp) semantics or with descriptive semantics, which is the usual first-order semantics for DLs.[1]

Regarding standard inferences, the subsumption and the instance problem turned out to be polynomial for both types of semantics. This is in strong contrast to the case of DLs with value restrictions, where even for the small DL $\mathcal{FL}_0$ (which allows for conjunctions and value restrictions only), adding cyclic terminologies increases the complexity of the subsumption problem from polynomial (for concept descriptions) to PSPACE [1, 9].

Regarding non-standard inferences it turned out that gfp-semantics is very well-behaved. With respect to this semantics the binary[2] lcs and the msc always exist and can be computed in polynomial time. For descriptive semantics, things are not as rosy. In [2] it was shown that, in general, the lcs need not exist. The paper then introduces possible candidates P_k ($k \geq 0$) for the lcs, and shows that the lcs exists iff one of these candidates is the lcs. It then gives a decidable sufficient characterization for the existence of the lcs. However, the question of how to decide the existence of the lcs in the general case remained open. In [3], analogous results were shown for the msc. In particular, the question of how to decide the existence of the msc also remained open.

In the present paper, we show that these open problems are both instances of a common graph-theoretic problem. Then we show that this graph-theoretic problem is decidable by reducing it to the problem of deciding satisfiability in monadic second-order logic on infinite trees [13]. Finally, we show that, if the lcs (msc) exists, then it can be computed in polynomial time.

In the next section, we introduce $\mathcal{EL}$ and define the subsumption and the instance problem as well as the lcs and the msc. In Section 3, we introduce the graph-theoretic problem that we want to solve in this paper, and then relate it to the problem of computing the lcs and the msc in $\mathcal{EL}$. Section 4 gives the reduction of this problem to monadic second-order logic, and Section 5 shows that the lcs (msc) can be computed in polynomial time whenever it exists.

2 Cyclic terminologies, least common subsumers, and most specific concepts

Concept descriptions are inductively defined with the help of a set of *constructors*, starting with a set N_C of *concept names* and a set N_R of *role names*. The constructors determine the expressive power of the DL. In this report, we restrict

[1] The results in [5] show that using least fixpoint semantics does not make sense in $\mathcal{EL}$.

[2] The n-ary lcs may grow exponentially even in $\mathcal{EL}$ without cyclic terminologies [7].

name of constructor	Syntax	Semantics
concept name $A \in N_C$	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
role name $r \in N_R$	r	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
top-concept	$\top$	$\Delta^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y : (x, y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
concept definition	$A \equiv D$	$A^{\mathcal{I}} = D^{\mathcal{I}}$
individual name $a \in N_I$	a	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
concept assertion	$A(a)$	$a^{\mathcal{I}} \in \mathcal{A}^{\mathcal{I}}$
role assertion	$r(a, b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

Table 1: Syntax and semantics of $\mathcal{EL}$-concept descriptions, TBox definitions, and ABox assertions.

the attention to the DL $\mathcal{EL}$, whose concept descriptions are formed using the constructors top-concept ($\top$), conjunction ($C \sqcap D$), and existential restriction ($\exists r.C$). The semantics of $\mathcal{EL}$-concept descriptions 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 $A \in N_C$ to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ and each role $r \in N_R$ to a binary relation $r^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$. The extension of $\cdot^{\mathcal{I}}$ to arbitrary concept descriptions is inductively defined, as shown in the third column of Table 1.

A *terminology* (or *TBox* for short) is a finite set of concept definitions of the form $A \equiv D$, where A is a concept name and D a concept description. In addition, we require that TBoxes do not contain *multiple definitions*, i.e., there cannot be two distinct concept descriptions D_1 and D_2 such that both $A \equiv D_1$ and $A \equiv D_2$ belongs to the TBox. Concept names occurring on the left-hand side of a definition are called *defined concepts*. All other concept names occurring in the TBox are called *primitive concepts*. Note that we allow for cyclic dependencies between the defined concepts, i.e., the definition of A may refer (directly or indirectly) to A itself. An interpretation $\mathcal{I}$ is a model of the TBox $\mathcal{T}$ iff it satisfies all its concept definitions, i.e., $A^{\mathcal{I}} = D^{\mathcal{I}}$ for all definitions $A \equiv D$ in $\mathcal{T}$.

An *ABox* is a finite set of assertions of the form $A(a)$ and $r(a, b)$, where A is a concept name, r is a role name, and a, b are individual names from a set N_I. Interpretations of ABoxes must additionally map each individual name $a \in N_I$ to an element $a^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$. An interpretation $\mathcal{I}$ is a model of the ABox $\mathcal{A}$ iff it satisfies all its assertions, i.e., $a^{\mathcal{I}} \in A^{\mathcal{I}}$ for all concept assertions $A(a)$ in $\mathcal{A}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$ for all role assertions $r(a, b)$ in $\mathcal{A}$. The interpretation $\mathcal{I}$ is a model of the ABox $\mathcal{A}$ together with the TBox $\mathcal{T}$ iff it is a model of both $\mathcal{T}$ and $\mathcal{A}$.

The semantics of (possibly cyclic) $\mathcal{EL}$-TBoxes we have defined above is called *descriptive semantic* by Nebel [12]. For some applications, it is more appropriate to interpret cyclic concept definitions with the help of an appropriate fixpoint semantics. However, in this paper we restrict our attention to descriptive semantics (see [5, 4] for definitions and results concerning cyclic terminologies in $\mathcal{EL}$ with fixpoint semantics).

We are now ready to define the subsumption and the instance problem w.r.t. descriptive semantics.

Definition 1 Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox and $\mathcal{A}$ an $\mathcal{EL}$-ABox, let C, D be concept descriptions (possibly containing defined concepts of $\mathcal{T}$), and a an individual name occurring in $\mathcal{A}$. Then,

- C is subsumed by D w.r.t. descriptive semantics ($C \sqsubseteq_{\mathcal{T}} D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$.
- a is an instance of C w.r.t. descriptive semantics ($\mathcal{A} \models_{\mathcal{T}} C(a)$) iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$ together with $\mathcal{A}$.

On the level of concept descriptions, the least common subsumer of two concept descriptions C, D is the least concept description E that subsumes both C and D. An extensions of this definition to the level of (possibly cyclic) TBoxes is not completely trivial. In fact, assume that A_1, A_2 are concepts defined in the TBox $\mathcal{T}$. It should be obvious that taking as the lcs of A_1, A_2 the least defined concept B in $\mathcal{T}$ such that $A_1 \sqsubseteq_{\mathcal{T}} B$ and $A_2 \sqsubseteq_{\mathcal{T}} B$ is too weak since the lcs would then strongly depend on what other defined concepts are already present in $\mathcal{T}$.

Consequently, to obtain the lcs we must allow the original TBox to be extended by new definitions. We say that the TBox $\mathcal{T}_2$ is a *conservative extension* of the TBox $\mathcal{T}_1$ iff $\mathcal{T}_1 \subseteq \mathcal{T}_2$ and $\mathcal{T}_1$ and $\mathcal{T}_2$ have the same primitive concepts and roles. Thus, $\mathcal{T}_2$ may contain new definitions $A \equiv D$, but then D does not introduce new primitive concepts and roles (i.e., all of them already occur in $\mathcal{T}_1$), and A is a new concept name (i.e., A does not occur in $\mathcal{T}_1$). The name "conservative extension" is justified by the fact that the new definitions in $\mathcal{T}_2$ do not influence the subsumption relationships between defined concepts in $\mathcal{T}_1$ (see [4]).

Definition 2 Let $\mathcal{T}_1$ be an $\mathcal{EL}$-TBox containing the defined concepts A, B, and let $\mathcal{T}_2$ be a conservative extension of $\mathcal{T}_1$ containing the new defined concept E. Then E in $\mathcal{T}_2$ is a *least common subsumer of A, B in $\mathcal{T}_1$ w.r.t. descriptive semantics* (lcs) iff the following two conditions are satisfied:

1. $A \sqsubseteq_{\mathcal{T}_2} E$ and $B \sqsubseteq_{\mathcal{T}_2} E$.
2. If $\mathcal{T}_3$ is a conservative extension of $\mathcal{T}_2$ and F a defined concept in $\mathcal{T}_3$ such that $A \sqsubseteq_{\mathcal{T}_3} F$ and $B \sqsubseteq_{\mathcal{T}_3} F$, then $E \sqsubseteq_{\mathcal{T}_3} F$.

The notion "most specific concept" can be extended in a similar way from concept descriptions to concepts defined in a TBox.

Definition 3 Let $\mathcal{T}_1$ be an $\mathcal{EL}$-TBox and $\mathcal{A}$ an $\mathcal{EL}$-ABox containing the individual name a, and let $\mathcal{T}_2$ be a conservative extension of $\mathcal{T}_1$ containing the defined concept E. Then E in $\mathcal{T}_2$ is a *most specific concept of a in $\mathcal{A}$ and $\mathcal{T}_1$ w.r.t. descriptive semantics* (msc) iff the following two conditions are satisfied:

1. $\mathcal{A} \models_{\mathcal{T}_2} E(a)$.
2. If $\mathcal{T}_3$ is a conservative extension of $\mathcal{T}_2$ and F a defined concept in $\mathcal{T}_3$ such that $\mathcal{A} \models_{\mathcal{T}_3} F(a)$, then $E \sqsubseteq_{\mathcal{T}_3} F$.

3 A graph-theoretic characterization of the lcs and the msc in $\mathcal{EL}$

In this section, we define the relevant graph-theoretic notions, and relate them to the subsumption and the instance problem as well as the problem of computing the lcs and the msc in $\mathcal{EL}$.

3.1 Graphs and synchronized simulations

For the purpose of this paper, a *graph* is of the form (V, E, L), where V is a finite set of nodes, $E \subseteq V \times N_e \times V$ is a set of edges labeled by elements of the finite set N_e, and L is a labelling function that assigns to every node $v \in V$ a subset $L(v)$ of the finite set N_n.

Simulations are binary relations on the nodes of a graph that respect node labels and edges in the sense defined below.

Definition 4 Let $\mathcal{G} = (V, E, L)$ be a graph. The binary relation $Z \subseteq V \times V$ is a *simulation* on $\mathcal{G}$ iff

(S1) $(v_1, v_2) \in Z$ implies $L(v_1) \subseteq L(v_2)$; and

(S2) if $(v_1, v_2) \in Z$ and $(v_1, r, v_1') \in E$, then there exists a node $v_2' \in V$ such that $(v_1', v_2') \in Z$ and $(v_2, r, v_2') \in E$.

It is easy to see that the set of all simulations on a graph $\mathcal{G}$ is closed under arbitrary unions, and thus there always exists a greatest simulation on $\mathcal{G}$. It is well-known that this greatest simulation can be computed in polynomial time [8].

$$\begin{tikzcd}
u \arrow[r, phantom, "="] & u_0 \arrow[r, "r_1"] \arrow[d, "Z"'] & u_1 \arrow[r, "r_2"] \arrow[d, "Z"'] & u_2 \arrow[r, "r_3"] \arrow[d, "Z"'] & u_3 \arrow[r, "r_4"] \arrow[d, "Z"'] & \cdots \\
v \arrow[r, phantom, "="] & v_0 \arrow[r, "r_1"] & v_1 \arrow[r, "r_2"] & v_2 \arrow[r, "r_3"] & v_3 \arrow[r, "r_4"] & \cdots
\end{tikzcd}$$

Figure 1: An infinite (u, v)-simulation chain.

$$\begin{tikzcd}
u \arrow[r, phantom, "="] & u_0 \arrow[r, "r_1"] \arrow[d, "Z"'] & u_1 \arrow[r, "r_2"] \arrow[d, "Z"'] & \cdots \arrow[r, "r_{n-1}"] & u_{n-1} \arrow[r, "r_n"] \arrow[d, "Z"'] & u_n \\
v \arrow[r, phantom, "="] & v_0 \arrow[r, "r_1"] & v_1 \arrow[r, "r_2"] & \cdots \arrow[r, "r_{n-1}"] & v_{n-1} &
\end{tikzcd}$$

Figure 2: A partial (u, v)-simulation chain.

Consequently, given two nodes u, v of $\mathcal{G}$, we can decide in time polynomial in the size of $\mathcal{G}$ whether there is a simulation Z such that $(u, v) \in Z$.

Here, we are not interested in arbitrary simulations containing a given pair of nodes,[3] but in ones that are synchronized in the sense defined below. If $(u, v) \in Z$, then any infinite path p_1 starting with u can be simulated by an infinite path p_2 starting with v. We call the pair p_1, p_2 a *(u, v)-simulation chain* (see Figure 1). Given an infinite path p_1 starting with u, we construct a simulating path p_2 step by step. The main point is, however, that the decision which node v_n to take in step n should depend only on the partial simulation chain already constructed, and *not* on the parts of the path p_1 not yet considered.

Definition 5 Let $\mathcal{G}$ be a graph, Z a simulation on $\mathcal{G}$, and u, v nodes of $\mathcal{G}$.

(1) A *partial (u, v)-simulation chain* is of the form depicted in Figure 2. A *selection function* S for u, v and Z assigns to each partial (u, v)-simulation chain of this form a node u_n such that (v_{n-1}, r_n, v_n) is an edge in $\mathcal{G}$ and $(u_n, v_n) \in Z$.

(2) Given an infinite path $u = u_0 \xrightarrow{r_1} u_1 \xrightarrow{r_2} u_2 \xrightarrow{r_3} u_3 \xrightarrow{r_4} \cdots$ and a node v such that $(u, v) \in Z$, one can use the selection function S to construct a Z-simulating path. In this case we say that the resulting (u, v)-simulation chain is *S-selected.*

(3) The simulation Z is called *(u, v)-synchronized* iff there exists a selection function S for Z such that the following holds: for every infinite S-selected (u, v)-simulation chain of the form depicted in Figure 1 there exists an $i \geq 0$ such that $u_i = v_i$.

We call a selection function *nice* iff $u_i = v_i$ in an S-selected (u, v)-simulation chain of the form depicted in Figure 1 implies $u_j = v_j$ for all $j \geq i$. It is easy to see that we can without loss of generality assume that all selection functions

[3] This would be sufficient for gfp-semantics.

are nice, i.e., if Z is (u, v)-*synchronized* then there is a *nice* selection function satisfying property (3) of Definition 5.

Before we continue defining the graph-theoretic notions necessary to characterize the existence of the lcs and the msc in $\mathcal{EL}$, we recall the connection between synchronized simulations and the subsumption and the instance problem proved in [5, 3].

3.2 The subsumption and the instance problem

It was shown in [5] that $\mathcal{EL}$-TBoxes and ABoxes can be represented as so-called description graphs. Before we can translate $\mathcal{EL}$-TBoxes into description graphs, we must normalize the TBoxes. In the following, let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, N_{def} the defined concepts of $\mathcal{T}$, N_{prim} the primitive concepts of $\mathcal{T}$, and N_{role} the roles of $\mathcal{T}$.

We say that the $\mathcal{EL}$-TBox $\mathcal{T}$ is *normalized* iff $A \equiv D \in \mathcal{T}$ implies that D is of the form

$$P_1 \sqcap \ldots \sqcap P_m \sqcap \exists r_1.B_1 \sqcap \ldots \sqcap \exists r_\ell.B_\ell,$$

for $m, \ell \geq 0$, $P_1, \ldots, P_m \in N_{prim}$, $r_1, \ldots, r_\ell \in N_{role}$, and $B_1, \ldots, B_\ell \in N_{def}$. If $m = \ell = 0$, then $D = \top$.

As shown in [5], one can (without loss of generality) restrict the attention to normalized TBox. In the following, we thus assume that all TBoxes are normalized. Normalized $\mathcal{EL}$-TBoxes can be viewed as graphs whose nodes are the defined concepts, which are labeled by sets of primitive concepts, and whose edges are given by the existential restrictions.

Definition 6 An $\mathcal{EL}$*-description graph* is a graph $\mathcal{G} = (V, E, L)$ where the edges are labeled with role names and the nodes are labeled with sets of primitive concepts. The TBox $\mathcal{T}$ can be translated into the following $\mathcal{EL}$-description graph $\mathcal{G}_\mathcal{T} = (N_{def}, E_\mathcal{T}, L_\mathcal{T})$:

- the nodes of $\mathcal{G}_\mathcal{T}$ are the defined concepts of $\mathcal{T}$;
- if A is a defined concept and

  $$A \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \exists r_1.B_1 \sqcap \ldots \sqcap \exists r_\ell.B_\ell$$

  its definition in $\mathcal{T}$, then

  - $L_\mathcal{T}(A) = \{P_1, \ldots, P_m\}$, and
  - A is the source of the edges $(A, r_1, B_1), \ldots, (A, r_\ell, B_\ell) \in E_\mathcal{T}$.

We are now ready to state the characterization of subsumption w.r.t. descriptive semantics given in [5].

Theorem 7 *Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, and A, B defined concepts in $\mathcal{T}$. Then the following are equivalent:*

1. $A \sqsubseteq_{\mathcal{T}} B$.
2. *There is a (B, A)-synchronized simulation Z on $\mathcal{G}_{\mathcal{T}}$ such that $(B, A) \in Z$.*

In [5] it is shown that the existence of such a synchronized simulation can be decided in polynomial time, and thus the subsumption problem w.r.t. descriptive semantics in $\mathcal{EL}$ can also be decided in polynomial time.

In order to characterize the instance problem, we assume that $\mathcal{T}$ is a normalized $\mathcal{EL}$-TBox and $\mathcal{A}$ an $\mathcal{EL}$-ABox. In the following, we assume that $\mathcal{T}$ is fixed and that all instance problems for $\mathcal{A}$ are considered w.r.t. this TBox. In this setting, $\mathcal{A}$ can be translated into an $\mathcal{EL}$-description graph $\mathcal{G}_{\mathcal{A}}$ by viewing $\mathcal{A}$ as a description graph and extending it appropriately by the description graph $\mathcal{G}_{\mathcal{T}}$ associated with $\mathcal{T}$.

Definition 8 Let $\mathcal{T}$ be a normalized $\mathcal{EL}$-TBox, $\mathcal{A}$ an $\mathcal{EL}$-ABox, and $\mathcal{G}_{\mathcal{T}} = (V, E, L)$ be the $\mathcal{EL}$-description graph associated with $\mathcal{T}$. The $\mathcal{EL}$-description graph $\mathcal{G}_{\mathcal{A}} = (V_{\mathcal{A}}, E_{\mathcal{A}}, L_{\mathcal{A}})$ associated with $\mathcal{A}$ and $\mathcal{T}$ is defined as follows:

- the nodes of $\mathcal{G}_{\mathcal{A}}$ are the individual names occurring in $\mathcal{A}$ together with the defined concepts of $\mathcal{T}$, i.e.,
$$V_{\mathcal{A}} := V \cup \{a \mid a \text{ is an individual name occurring in } \mathcal{A}\};$$
- the edges of $\mathcal{G}_{\mathcal{A}}$ are the edges of $\mathcal{G}$, the role assertions of $\mathcal{A}$, and additional edges linking the ABox individuals with defined concepts:
$$\begin{aligned} E_{\mathcal{A}} \;:=\; & E \cup \{(a, r, b) \mid r(a, b) \in \mathcal{A}\} \;\cup \\ & \{(a, r, B) \mid A(a) \in \mathcal{A} \text{ and } (A, r, B) \in E\}; \end{aligned}$$
- if $u \in V_{\mathcal{A}}$ is a defined concept, then it inherits its label from $\mathcal{G}_{\mathcal{T}}$, i.e.,
$$L_{\mathcal{A}}(u) := L(u) \quad \text{if } u \in V;$$
otherwise, u is an ABox individual, and then its label is derived from the concept assertions for u in $\mathcal{A}$. In the following, let P denote primitive and A denote defined concepts.
$$L_{\mathcal{A}}(u) := \{P \mid P(u) \in \mathcal{A}\} \cup \bigcup_{A(u) \in \mathcal{A}} L(A) \quad \text{if } u \in V_{\mathcal{A}} \setminus V.$$

We are now ready to recall the characterization of the instance problem in $\mathcal{EL}$ w.r.t. descriptive semantics given in [3].

Theorem 9 *Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, $\mathcal{A}$ an $\mathcal{EL}$-ABox, A a defined concept in $\mathcal{T}$ and a an individual name occurring in $\mathcal{A}$. Then the following are equivalent:*

1. *$\mathcal{A} \models_{\mathcal{T}} A(a)$.*
2. *There is a simulation Z on $\mathcal{G}_{\mathcal{A}}$ such that*
    - *$(A, a) \in Z$, and*
    - *Z is (B, u)-synchronized for all defined concepts B in $\mathcal{T}$ and nodes u of $\mathcal{G}_{\mathcal{A}}$ such that $(B, u) \in Z$.*

3.3 The main problem

To define the main graph-theoretic problem addressed in this paper, we define a type of graphs that looks like the $\mathcal{EL}$-description graphs $\mathcal{G}_{\mathcal{A}}$ obtained from an $\mathcal{EL}$-TBox and ABox.

Definition 10 The graph $\mathcal{G} = (V, E, L)$ is called *two-level graph* iff V can be partitioned into disjoint sets $V = V_1 \cup V_2$ such that $(v, r, v') \in E$ implies $v \in V_1$ or $v' \in V_2$. To make this partition explicit, we write two-level graphs as $\mathcal{G} = (V_1 \cup V_2, E, L)$.

Intuitively, a two-level graph $\mathcal{G} = (V_1 \cup V_2, E, L)$ consists of a subgraph $\mathcal{G}_1$ on V_1, a subgraph $\mathcal{G}_2$ on V_2, and possibly additional edges from nodes of $\mathcal{G}_1$ to nodes of $\mathcal{G}_2$. Obviously, the graph $\mathcal{G}_{\mathcal{A}}$ obtained from an $\mathcal{EL}$-TBox $\mathcal{T}$ and an $\mathcal{EL}$-ABox $\mathcal{A}$ is a two-level graph, where V_1 is the set of individual names occurring in $\mathcal{A}$ and V_2 is the set of concepts defined in $\mathcal{T}$.

In order to motivate the next definition, in which the subgraph $\mathcal{G}_1$ of the two-level graph $\mathcal{G} = (V_1 \cup V_2, E, L)$ is unraveled up to a certain depth, we sketch how the msc of an individual a can be obtained from $\mathcal{G}_{\mathcal{A}}$. The main idea underlying the characterization of the msc in $\mathcal{EL}$ w.r.t. descriptive semantics given in [3] is the following. We can view $\mathcal{G}_{\mathcal{A}}$ as the $\mathcal{EL}$-description graph of an $\mathcal{EL}$-TBox $\mathcal{T}_2$. It is easy to see that $\mathcal{T}_2$ is a conservative extension of $\mathcal{T}$. By the definition of $\mathcal{G}_{\mathcal{A}}$, the defined concepts of $\mathcal{T}_2$ are the defined concepts of $\mathcal{T}$ together with the individual names occurring in $\mathcal{A}$. To avoid confusion we denote the defined concept in $\mathcal{T}_2$ corresponding to the individual name b in $\mathcal{A}$ by C_b. In [4] it is shown that, w.r.t. gfp-semantics, the defined concept C_a in $\mathcal{T}_2$ is the most specific concept of a in $\mathcal{A}$ and $\mathcal{T}$. W.r.t. descriptive semantics, this is only true if $\mathcal{A}$ does not contain a cycle that is reachable from a. Otherwise, it easily follows from Theorem 9 that

a cannot be an instance of C_a. To avoid this problem, acyclic versions $\mathcal{G}_{\mathcal{A}}^{(k)}$ of $\mathcal{G}_{\mathcal{A}}$ (where cycles in $\mathcal{A}$ are unraveled into paths up to depth k starting with a) are introduced in [3]. When viewed as the $\mathcal{EL}$-description graph of an $\mathcal{EL}$-TBox, the graph $\mathcal{G}_{\mathcal{A}}^{(k)}$ contains a defined concept that corresponds to the individual a. Let us call this concept P_k. In [3] it is shown that the msc of a exists iff there is a k such that P_k is the msc. In addition, it is shown that P_k is the msc of a iff P_k is subsumed by all P_ℓ for $\ell \geq k$.

Definition 11 Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph and $u \in V_1$. The *k-unraveling of $\mathcal{G}$ w.r.t. u* is the two-level graph $\mathcal{G}_u^{(k)} := (V_1^{(k)} \cup V_2, E^{(k)}, L^{(k)})$, where

$$V_1^{(k)} := \{u_0^{(k)}\} \cup \{v_i^{(k)} \mid v \in V_1 \text{ and } 1 \leq i \leq k\};$$

$$\begin{aligned}
E^{(k)} &:= \{(v,r,w) \mid (v,r,w) \in E \text{ and } v,w \in V_2\} \ \cup \\
&\quad\ \{(v_i^{(k)}, r, w_{i+1}^{(k)}) \mid (v,r,w) \in E \text{ and } v_i^{(k)}, w_{i+1}^{(k)} \in V_1^{(k)}\} \cup \\
&\quad\ \{(v_i^{(k)}, r, w) \mid (v,r,w) \in E \text{ and } v_i^{(k)} \in V_1^{(k)}, w \in V_2\}; \\
L^{(k)}(v) &:= L(v) \quad \text{if } v \in V_2, \\
L^{(k)}(v_i^{(k)}) &:= L(v) \quad \text{if } v_i^{(k)} \in V_1^{(k)}.
\end{aligned}$$

Obviously, the k-unraveling of $\mathcal{G}$ w.r.t. u consists of an acyclic subgraph on $V_1^{(k)}$ (where any path starting with $u_0^{(k)}$ has length at most k), an arbitrary subgraph on V_2 (which coincides with the original subgraph of $\mathcal{G}$ on V_2), and additional edges from the acyclic graph into V_2 (which are induced by corresponding edges in $\mathcal{G}$).

Given two different such unravelings $\mathcal{G}_u^{(k)} = (V_1^{(k)} \cup V_2, E^{(k)}, L^{(k)})$ and $\mathcal{G}_u^{(\ell)} = (V_1^{(\ell)} \cup V_2, E^{(\ell)}, L^{(\ell)})$ of $\mathcal{G}(V_1 \cup V_2, E, L)$, their union $\mathcal{G}_u^{(k)} \cup \mathcal{G}_u^{(\ell)}$ is defined in the obvious way by building the union of the node sets, the edge sets, and the labeling functions.[4]

Definition 12 Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph, $u \in V_1$, and $k \neq \ell$. We say that $\mathcal{G}_u^{(\ell)}$ *subsumes* $\mathcal{G}_u^{(k)}$ ($\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(\ell)}$) iff there is a $(u_0^{(\ell)}, u_0^{(k)})$-synchronized simulation Z on $\mathcal{G}_u^{(k)} \cup \mathcal{G}_u^{(\ell)}$ such that $(u_0^{(\ell)}, u_0^{(k)}) \in Z$.

It is easy to see that $\ell > k$ implies $\mathcal{G}_u^{(\ell)} \sqsubseteq \mathcal{G}_u^{(k)}$ (see also Lemma 3 in [3]). We are interested in finding an index k such that the subsumption relationship also holds in the other direction.

[4] Note that the two labeling functions agree on V_2, which is the set of nodes shared by $\mathcal{G}_u^{(k)}$ and $\mathcal{G}_u^{(\ell)}$.

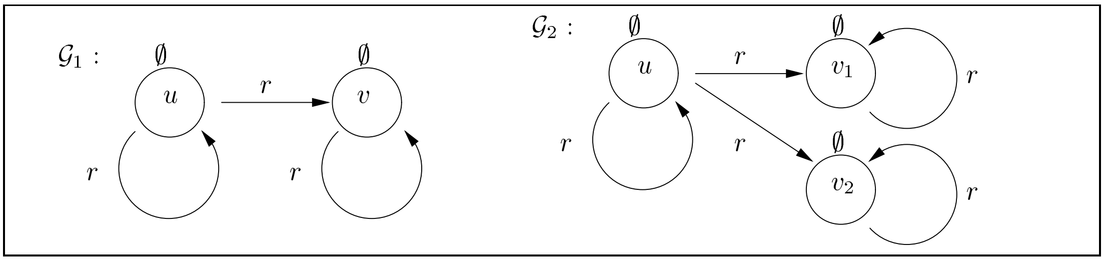

Figure 3: Two two-level graphs, one bounded and one unbounded.

Definition 13 Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph and $u \in V_1$. We say that $\mathcal{G}$ *is of bounded cycle depth w.r.t.* u iff there is a $k \geq 0$ such that $\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(\ell)}$ holds for all $\ell > k$. In this case, the minimal such k is called the *cycle depth* of $\mathcal{G}$ w.r.t. u.

The main decision problem considered in this paper is the following:

Given: A two-level graph $\mathcal{G} = (V_1 \cup V_2, E, L)$ and a node $u \in V_1$.

Question: Is $\mathcal{G}$ of bounded cycle depth w.r.t. u?

Before showing the connection of this problem to the problem of deciding the existence of the lcs and the msc in $\mathcal{EL}$ w.r.t. descriptive semantics, let us consider three examples.

First, consider the two-level graph $\mathcal{G}_1$ on the left-hand side of Figure 3 (where $V_1 := \{u\}$ and $V_2 := \{v\}$). This graph is of bounded cycle depth w.r.t. u. In fact, already $k = 0$ satisfies Definition 13 since any infinite path starting with $u_0^{(\ell)}$ will eventually lead to v, and thus can be simulated by the path $u_0^{(0)} \xrightarrow{r} v \xrightarrow{r} v \xrightarrow{r} \cdots$.

Second, consider the two-level graph $\mathcal{G}_2$ on the right-hand side of Figure 3 (where $V_1 := \{u\}$ and $V_2 := \{v_1, v_2\}$). Though this graph looks quite similar to $\mathcal{G}_1$, it is not of bounded cycle depth. In fact, $\mathcal{G}_{2,u}^{(k)} \not\sqsubseteq \mathcal{G}_{2,u}^{(k+1)}$ for all $k \geq 0$. To see this, consider the path p_1

$$u_0^{(k+1)} \xrightarrow{r} \cdots \xrightarrow{r} u_k^{(k+1)} \xrightarrow{r} u_{k+1}^{(k+1)}$$

of length $k+1$ in $\mathcal{G}_{2,u}^{(k+1)}$. If this path is simulated by a path p_2 of length $k+1$ in $\mathcal{G}_{2,u}^{(k)}$, then the last node of p_2 is either v_1 or v_2. Assume without loss of generality that it is v_1. If we continue the path p_1 by an infinite loop through v_2, then this infinite path p_1' can only be simulated in $\mathcal{G}_{2,u}^{(k)}$ by continuing to go through the node v_1. Thus, no synchronization occurs.

Third, the two-level graph $\mathcal{G}_3$ depicted in Figure 4 (where $V_1 = \{u_1, u_2\}$ and $V_2 = \{v\}$) is not of bounded cycle depth w.r.t. u_1, but shows a somewhat surprising phenomenon. Here we have $\mathcal{G}_{3,u_1}^{(k)} \sqsubseteq \mathcal{G}_{3,u_1}^{(k+1)}$ for all odd numbers k, but $\mathcal{G}_{3,u_1}^{(k)} \not\sqsubseteq$

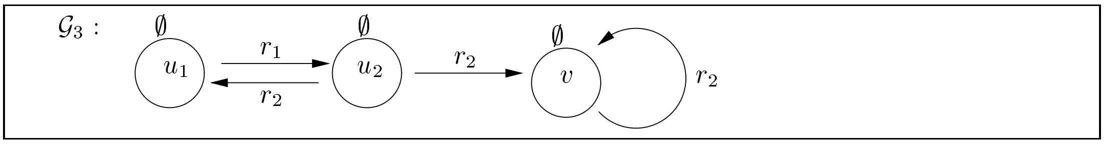

Figure 4: Another unbounded two-level graph.

$\mathcal{G}^{(k+1)}_{3,u_1}$ if k is even. First, assume that k is odd. Then there are no infinite paths in $\mathcal{G}^{(k+1)}_{3,u_1}$ that use the node $u^{(k+1)}_{1,k+1}$ since this node does not have a successor node. As an easy consequence, every infinite path in $\mathcal{G}^{(k+1)}_{3,u_1}$ can be simulated by "the same" path in $\mathcal{G}^{(k)}_{3,u_1}$. In addition, the finite path to $u^{(k+1)}_{1,k+1}$ can be simulated by a path $\mathcal{G}^{(k)}_{3,u_1}$ that ends with v. Consequently, $\mathcal{G}^{(k)}_{3,u_1} \sqsubseteq \mathcal{G}^{(k+1)}_{3,u_1}$ for odd k. In contrast, if k is even, then $u^{(k+1)}_{1,k}$ has a successor node in $\mathcal{G}^{(k+1)}_{3,u_1}$ (namely $u^{(k+1)}_{2,k+1}$) reached by an edge with label r_1. Any node reachable from $u^{(k)}_{1,0}$ in $\mathcal{G}^{(k)}_{3,u_1}$ by a path of length k (i.e., $u^{(k)}_{1,k}$ or v) does not have a successor w.r.t. r_1. Thus, there is a path in $\mathcal{G}^{(k+1)}_{3,u_1}$ that cannot be simulated by a path in $\mathcal{G}^{(k)}_{3,u_1}$, which shows that $\mathcal{G}^{(k)}_{3,u_1} \not\sqsubseteq \mathcal{G}^{(k+1)}_{3,u_1}$ for even k.

The last example shows that, in order to find the number k required by Definition 13, one cannot simply test subsumption between $\mathcal{G}^{(i+1)}_u$ and $\mathcal{G}^{(i)}_u$ for $i = 0, 1, 2, \ldots$ until $\mathcal{G}^{(i)}_u \sqsubseteq \mathcal{G}^{(i+1)}_u$, and then stop with output $k = i$.

3.4 The lcs and the msc

We can now reformulate the characterization of the lcs and the msc given in [2] and [3], respectively, in terms of the notions introduced above.

Proposition 14 ([3]) *Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, $\mathcal{A}$ an $\mathcal{EL}$-ABox, and a an individual in $\mathcal{A}$. Then a has an msc in $\mathcal{A}$ and $\mathcal{T}$ w.r.t. descriptive semantics iff the two-level graph $\mathcal{G}_\mathcal{A}$ (where V_1 consists of the individual names in $\mathcal{A}$ and V_2 consists of the defined concepts in $\mathcal{T}$) is of bounded cycle depth w.r.t. a.*

Assume that $\mathcal{G}_\mathcal{A}$ is of bounded cycle depth and that k is the cycle depth of $\mathcal{G}_\mathcal{A}$ w.r.t. a. In [3] it is shown that the msc of a in $\mathcal{A}$ and $\mathcal{T}$ is given by the "concept" $a^{(k)}_0$ in the TBox corresponding to the $\mathcal{EL}$-description graph $\mathcal{G}^{(k)}_{\mathcal{A},a}$. Consequently, it can be computed in time linear in the size of $\mathcal{G}^{(k)}_{\mathcal{A},a}$, i.e., in time linear in $|\mathcal{A}| \cdot k + |\mathcal{T}|$, where $|\,.\,|$ denotes the size of a TBox/ABox.

In order to give a similar characterization of the existence of the lcs, we must define the right two-level graph. This graph is based on the product of the $\mathcal{EL}$-description graph $\mathcal{G}_\mathcal{T}$ with itself.

Definition 15 Let $\mathcal{G}_1 = (V_1, E_1, L_1)$ and $\mathcal{G}_2 = (V_2, E_2, L_2)$ be two graphs. Their *product* is the graph $\mathcal{G}_1 \times \mathcal{G}_2 := (V, E, L)$ where

- $V := V_1 \times V_2$;
- $E := \{((v_1, v_2), r, (v_1', v_2')) \mid (v_1, r, v_1') \in E_1 \wedge (v_2, r, v_2') \in E_2\}$;
- $L(v_1, v_2) := L_1(v_1) \cap L_2(v_2)$.

Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox and A, B defined concepts in $\mathcal{T}$. The description graph $\mathcal{G}_\mathcal{T} \times \mathcal{G}_\mathcal{T}$ yields a TBox $\mathcal{T}_1$ such that $\mathcal{G}_{\mathcal{T}_1} = \mathcal{G}_\mathcal{T} \times \mathcal{G}_\mathcal{T}$. It is easy to see that $\mathcal{T}_2 := \mathcal{T}_1 \cup \mathcal{T}$ is a conservative extension of $\mathcal{T}$. With respect to gfp-semantics, the defined concept (A, B) in $\mathcal{T}_2$ is the lcs of A and B in $\mathcal{T}$ (see [4]). With respect to descriptive semantics, this is not the case, due to the possible existence of cycles starting with (A, B) in the product graph $\mathcal{G}_\mathcal{T} \times \mathcal{G}_\mathcal{T}$. As with the msc, this problem is solved by unraveling these cycles up to a certain depth, but before doing this we must introduce additional edges between the product graph $G_\mathcal{T} \times \mathcal{G}_\mathcal{T}$ and $\mathcal{G}_\mathcal{T}$.

Definition 16 Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, $\mathcal{G}_\mathcal{T} = (V_2, E_2, L_2)$ the corresponding $\mathcal{EL}$-description graph, and $\mathcal{G}_1 = (V_1, E_1, L_1)$ the product graph of $\mathcal{G}_\mathcal{T}$ with itself. The two-level graph $\widehat{\mathcal{G}}_\mathcal{T} = (\widehat{V_1} \cup V_2, E, L)$ is defined as follows:

- $\widehat{V_1} := \{(u, v) \in V_1 \mid u \neq v\}$;
- $E := E_1 \cup E_2 \cup \{((u_1, u_2), r, v) \mid (u_1, r, v) \in E_2 \wedge (u_2, r, v) \in E_2 \wedge u_1 \neq u_2\}$;
- $L(u, v) := L_1(u, v)$ if $(u, v) \in V_1$, and $L(u) := L_2(u)$ if $u \in V_2$.

Proposition 17 ([2]) *Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, and A, B distinct defined concepts in $\mathcal{T}$. Then A, B have an lcs in $\mathcal{T}$ w.r.t. descriptive semantics iff the two-level graph $\widehat{\mathcal{G}}_\mathcal{T} = (\widehat{V_1} \cup V_2, E, L)$ is of bounded cycle depth w.r.t. (A, B).*

Assume that $\widehat{\mathcal{G}}_\mathcal{T}$ is of bounded cycle depth and that k is the cycle depth of this two-level graph w.r.t. (A, B). In [2] it is shown that the lcs of A, B in $\mathcal{T}$ is given by the "concept" $(A, B)_0^{(k)}$ in the TBox corresponding to the $\mathcal{EL}$-description graph $\widehat{\mathcal{G}}^{(k)}_{\mathcal{T},(A,B)}$. Consequently, it can be computed in time linear in the size of this graph, i.e., in time linear in $|\mathcal{T}|^2 \cdot k + |\mathcal{T}|$.

4 Deciding if a graph is of bounded cycle depth

Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph, and $u \in V_1$. We reduce the problem of deciding whether $\mathcal{G}$ is of bounded cycle depth w.r.t. u to the problem of deciding whether a certain formula $\phi^u_{\mathcal{G}}$ of monadic second-order logic (MSO) on infinite trees is satisfiable. As shown by Rabin [13], the satisfiability problem for MSO is decidable. In the following, we assume that the reader is familiar with MSO on infinite trees (see, e.g., [17] for an introduction). Before we define the formula $\phi^u_{\mathcal{G}}$, we describe the intuition underlying this reduction.

Encoding synchronized simulations by infinite trees. The main idea underlying our reduction is that all simulation chains starting with a given pair of nodes of a graph $\mathcal{G} = (V, E, L)$ and selected by some selection function (see Definition 5) can be represented by an infinite tree t. Basically, the nodes of this tree are labeled with pairs of nodes of $\mathcal{G}$. Assume that the node n of t has label (u, v). If $(u, r_1, u_1), \ldots, (u, r_p, u_p)$ are all the edges in $\mathcal{G}$ starting with u, then the node n has p successor nodes $n_1, \ldots, n_p$ that are respectively labeled with $(u_1, v_1), \ldots, (u_p, v_p)$, where v_i is the result of applying the selection function to the partial simulation chain determined by the path in t leading to the node n and the edge (u, r_i, u_i). Since in MSO one considers trees with a fixed branching factor, the node n may have some additional dummy successor nodes labeled with the dummy label $\sharp$. Note that the simulation relation Z itself is also encoded in the tree t: it consists of all tuples (u, v) such that $(u, v) \in V \times V$ is the label of a node n of t. Because of the definition of the successor nodes of the nodes in t, property (S2) in the definition of a simulation relation (Definition 4) is satisfied. To ensure that Z also satisfies (S1), it is enough to require $L(u) \subseteq L(v)$ for all labels $(u, v) \in V \times V$ of nodes in t. Given two nodes u, v of $\mathcal{G}$, how can we ensure that the simulation relation Z encoded by such a tree t contains (u, v) and is (u, v)-synchronized? To ensure that $(u, v) \in Z$, we require that (u, v) is the label of the root of t. To ensure synchronization, we must require that on all infinite paths in the tree t, we encounter a label of the form (v', v') or $\sharp$. This can easily be expressed in MSO.

What we have said until now can be used to show that the following decision problem is decidable: given a graph $\mathcal{G}$ and nodes u, v in $\mathcal{G}$, is there a (u, v)-synchronized simulation Z such that $(u, v) \in Z$. However, decidability of this problem (in polynomial time) was already shown directly in [5] without the need for a reduction to the (complex) logic MSO.

What we actually want to decide here is whether a given two-level graph $\mathcal{G} = (V_1 \cup V_2, E, L)$ is of bounded cycle depth w.r.t. a node $u \in V_1$. For this, we must consider not $\mathcal{G}$ itself but rather unravelings $\mathcal{G}^{(k)}_u$ and $\mathcal{G}^{(\ell)}_u$ of $\mathcal{G}$. In addition, we need to express the quantification on the numbers k and ℓ ("there exists a k such that for all ℓ") by (second-order) quantifiers in MSO.

Encoding unravelings $\mathcal{G}^{(k)}_u$ and $\mathcal{G}^{(\ell)}_u$ and the quantification on k and ℓ. Assume that we have an infinite tree t encoding a (u, u)-synchronized simulation Z on $\mathcal{G}$, as described above. If (v_1, v_2) is the label of a node n on some level i of t, then there are paths of length i from u to v_1 and from u to v_2, respectively. The first (second) path corresponds to a path in $\mathcal{G}_u^{(\ell)}$ ($\mathcal{G}_u^{(k)}$) iff $i \leq \ell$ or $v_1 \in V_2$ ($i \leq k$ or $v_2 \in V_2$). Thus, the idea could be to introduce two second-order variables X and Y (with the appropriate quantifier prefix $\exists Y.\forall X.$), and then ensure that X contains exactly the nodes of t up to some level ℓ, and Y contains exactly the nodes of t up to some level k. In order to ensure that the paths in $\mathcal{G}$ encoded in the tree t really belong to $\mathcal{G}_u^{(\ell)}$ (when considering the first component of the node labels) and $\mathcal{G}_u^{(k)}$ (when considering the second component of the node labels), we must require that, for a node n labeled with (v_1, v_2), we have $X(n)$ or $v_1 \in V_2$, and $Y(n)$ or $v_2 \in V_2$. Unfortunately, sets containing exactly the nodes of an infinite tree up to some depth bound are not expressible in MSO.[5] However, for our purposes it turns out to be sufficient to ensure that X and Y are finite prefix-closed sets (i.e., if a node n that is not the root node belongs to one of them, then its predecessor also does). Both "prefix-closed" and "finite" can easily be expressed in MSO.

The formal definition. Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph, $u \in V_1$, and assume that b is the maximal number of successors of the nodes in $\mathcal{G}$. To define the formula $\phi_{\mathcal{G}}^u$, we consider the infinite trees with branching factor b (i.e., we have b successor functions $s_1, \ldots, s_b$ in the signature of MSO). As usual, we will denote second-order variables (standing for sets of nodes) by upper-case letters, and first-order variables (standing for nodes) by lower-case letters. The second-order variables used in the following are

- the variables X and Y whose function was already explained above;
- variables $Q_{(u_1,u_2)}$ for $(u_1, u_2) \in (V_1 \cup V_2) \times (V_1 \cup V_2)$ and $Q_\sharp$. The values of these variables encode the selection function S by encoding all S-selected simulation chains. Intuitively, a node n of the tree belongs to $Q_{(u_1,u_2)}$ ($Q_\sharp$) iff it is labeled with (u_1, u_2) ($\sharp$);
- the variable P standing for an infinite path in the tree, which is used to express the synchronization property.

The formula $\phi_{\mathcal{G}}^u$ is defined as

$$\exists Y.(\mathit{PrefixClosed}(Y) \land \mathit{Finite}(Y) \land \forall X.(\mathit{PrefixClosed}(X) \land \mathit{Finite}(X) \Rightarrow \psi_{\mathcal{G}}^u)),$$

where *PrefixClosed*(.) and *Finite*(.) are the well-known MSO-formulae expressing that a set of nodes is prefix-closed and finite, respectively,[6] and $\psi_{\mathcal{G}}^u$ consists of an existential quantifier prefix on the variables $Q_{(u_1,u_2)}$ for $(u_1, u_2) \in (V_1 \cup V_2) \times (V_1 \cup V_2)$ and $Q_\sharp$, followed by the conjunction $\vartheta_{\mathcal{G}}^u$ of the following formulae:

[5] Since then one could also express that two nodes are on the same level, which is know to be inexpressible in MSO [17].

[6] Defining *PrefixClosed*(.) is a simple exercise. A definition of *Finite*(.) can be found in [17].

- *A formula expressing that any node has exactly one label.*

$$\forall x. \bigvee_{l_1 \in (V_1 \cup V_2) \times (V_1 \cup V_2) \cup \{\sharp\}} \left(Q_{l_1}(x) \wedge \bigwedge_{\substack{l_2 \in (V_1 \cup V_2) \times (V_1 \cup V_2) \cup \{\sharp\} \\ l_2 \neq l_1}} \neg Q_{l_2}(x) \right)$$

- *A formula expressing that the root has label (u, u).*

$$Q_{(u,u)}(root)$$

- *Formulae expressing the function of the sets X and Y.* For all $(u', u'') \in V_1 \times (V_1 \cup V_2)$ the formula

$$\forall x. Q_{(u',u'')}(x) \Rightarrow X(x)$$

and for all $(u', u'') \in (V_1 \cup V_2) \times V_1$ the formula

$$\forall x. Q_{(u',u'')}(x) \Rightarrow Y(x)$$

- *Formulae encoding the requirements on the selection function.* Let $(u', u'') \in (V_1 \cup V_2) \times (V_1 \cup V_2)$, and let $(u', r_1, v'_1), \ldots, (u', r_p, v'_p)$ be all the edges in E with source u'. First, for each $i, 1 \leq i \leq p$ we have one formula in the conjunction. If $v'_i \in V_2$, then we take the formula

$$\forall x. Q_{(u',u'')}(x) \Rightarrow \left(\bigvee_{(u'', r_i, v'') \in E \wedge L(v'_i) \subseteq L(v'')} Q_{(v'_i, v'')}(s_i(x)) \right)$$

Otherwise (i.e., if $v'_i \in V_1$), then we take the formula

$$\forall x. \left(Q_{(u',u'')}(x) \wedge X(s_i(x)) \right) \Rightarrow \left(\bigvee_{(u'', r_i, v'') \in E \wedge L(v'_i) \subseteq L(v'')} Q_{(v'_i, v'')}(s_i(x)) \right)$$

Second, we need formulae that fill in the appropriate dummy nodes:

$$\forall x. Q_{(u',u'')}(x) \Rightarrow \left(\bigwedge_{j=p+1}^{j=b} Q_\sharp(s_j(x)) \right)$$

and for all $i, 1 \leq i \leq p$, such that $v'_i \in V_1$

$$\forall x. \left(Q_{(u',u'')}(x) \wedge \neg X(s_i(x)) \right) \Rightarrow Q_\sharp(s_i(x))$$

- *A formula expressing that dummy nodes have only dummy successors.*

$$\forall x. Q_\sharp(x) \Rightarrow \left(\bigwedge_{j=1}^{j=b} Q_\sharp(s_j(x)) \right)$$

- *A formula expressing the synchronization property.*

$$\forall P.\mathit{Path}(P) \Rightarrow \exists x.P(x) \wedge \left(Q_\sharp(x) \vee \bigvee_{(v,v)\in V_2} Q_{(v,v)}(x)\right)$$

  where *Path*(.) is the well-known MSO-formula expressing that a set of nodes consists of the nodes on an infinite path starting with the root (see [17]).

Lemma 18 *Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph, and $u \in V_1$. Then $\mathcal{G}$ is of bounded cycle depth w.r.t. u iff the MSO-formula $\phi^u_\mathcal{G}$ is satisfiable.*[7]

Proof. First, assume that $\mathcal{G}$ is of bounded cycle depth w.r.t. u, and let k be the cycle depth of $\mathcal{G}$ w.r.t. u. To show that $\phi^u_\mathcal{G}$ is satisfiable, we take as set Y the set $\mathcal{K}$ of all nodes of depth at most k in the infinite tree with branching factor b. Now, let $\mathcal{L}$ be an arbitrary finite prefix-closed set of nodes of the infinite tree with branching factor b. Since $\mathcal{L}$ is finite, there is a number $\ell > k$ such that all nodes in $\mathcal{L}$ are on depth at most ℓ. Since k is the cycle depth of $\mathcal{G}$ w.r.t. u, we know that $\mathcal{G}^{(k)}_u \sqsubseteq \mathcal{G}^{(\ell)}_u$. Let $\mathcal{L}'$ be the set of all nodes of depth at most ℓ in the infinite tree with branching factor b. By our construction of the formula $\psi^u_\mathcal{G}$, $\mathcal{G}^{(k)}_u \sqsubseteq \mathcal{G}^{(\ell)}_u$ implies that the formula $\psi^u_\mathcal{G}$ is satisfiable with Y replaced by $\mathcal{K}$ and X replaced by $\mathcal{L}'$. Since $\mathcal{L} \subseteq \mathcal{L}'$, this is also true if we replace X by $\mathcal{L}$ instead of $\mathcal{L}'$. Consequently, we have shown that $\phi^u_\mathcal{G}$ is satisfiable.

Second, assume that $\phi^u_\mathcal{G}$ is satisfiable. Let $\mathcal{K}$ be a finite prefix-closed set of nodes such that the formula

$$\eta^u_\mathcal{G} := \forall X.(\mathit{PrefixClosed}(X) \wedge \mathit{Finite}(X) \Rightarrow \psi^u_\mathcal{G})$$

is satisfiable with Y replaced by $\mathcal{K}$. Since $\mathcal{K}$ is finite, there is a number $k \geq 0$ such that all nodes in $\mathcal{K}$ are on depth at most k. Let $\mathcal{K}'$ be the set of all nodes of depth at most k in the infinite tree with branching factor b. Since $\mathcal{K} \subseteq \mathcal{K}'$, $\eta^u_\mathcal{G}$ is also satisfiable with Y replaced by $\mathcal{K}'$. Thus, if $\ell > k$ and $\mathcal{L}$ denotes the set of all nodes of depth at most ℓ in the infinite tree with branching factor b, then $\psi^u_\mathcal{G}$ is satisfiable with Y replaced by $\mathcal{K}'$ and X replaced by $\mathcal{L}$. By our construction of the formula $\psi^u_\mathcal{G}$, this implies that $\mathcal{G}^{(k)}_u \sqsubseteq \mathcal{G}^{(\ell)}_u$. □

Since satisfiability in MSO on infinite trees is decidable, the lemma implies decidability of bounded cycle depth.

Theorem 19 *The problem of deciding whether a two-level graph is of bounded cycle depth w.r.t. one of its nodes is decidable.*

[7] Since we have only one possible model, the infinite tree with branching factor b, satisfiability and validity are actually the same here.

Unfortunately, the reduction does not give us a polynomial (or even a singly exponential) complexity bound for this decision problem. This is due to the fact that the formula $\phi_{\mathcal{G}}^{u}$ contains several quantifier changes.[8]

Together with Propositions 14 and 17, this theorem implies that the existence of the lcs and the msc is decidable in $\mathcal{EL}$ with descriptive semantics.

Corollary 20 *The following problems are decidable:*

1. *Given an $\mathcal{EL}$-TBox $\mathcal{T}$ and concepts A, B defined in $\mathcal{T}$. Do A, B in $\mathcal{T}$ have an lcs w.r.t. descriptive semantics?*
2. *Given an $\mathcal{EL}$-TBox $\mathcal{T}$, an $\mathcal{EL}$-ABox $\mathcal{A}$, and an individual a in $\mathcal{A}$. Does a in $\mathcal{A}$ and $\mathcal{T}$ have an msc w.r.t. descriptive semantics?*

5 A polynomial bound on the cycle depth

A given two-level graph need not be of bounded cycle depth, but if it is then we can show that its cycle depth is actually polynomial in the size of the graph.

Theorem 21 *Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph, $u \in V_1$, and let m be the cardinality of $V_1 \cup V_2$. Then $\mathcal{G}$ is of bounded cycle depth iff $\mathcal{G}$ has cycle depth d w.r.t. u for some $d \leq m^2$.*

The "if" direction of this theorem is trivial. To prove the "only-if" direction, assume that $k > m^2$ is such that $\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(\ell)}$ for all $\ell > k$. To show that the cycle depth of $\mathcal{G}$ w.r.t. u is at most m^2, it is sufficient to show that $\mathcal{G}_u^{(m^2)} \sqsubseteq \mathcal{G}_u^{(\ell)}$ holds for all $\ell > m^2$. To show this, it is in turn enough to show that $\mathcal{G}_u^{(m^2)} \sqsubseteq \mathcal{G}_u^{(k)}$. This is a consequence of the following two facts:

1. $\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(\ell)}$ is trivially true for all $\ell < k$ and it holds for all $\ell > k$ by our assumption on k.
2. The subsumption relation $\sqsubseteq$ is transitive. In fact, if we assume selection functions to be nice[9] (which we can do without loss of generality), then the composition of two synchronized simulations is again a synchronized simulation.

[8] In Rabin's decidability proof based on automata, every negation requires a worst-case exponential complementation operation, and expressing a universal quantifier by an existential one (as required by Rabin's decision procedure) introduces two negation signs.

[9] Recall that this means that $u_i = v_i$ in an S-selected simulation chain of the form depicted in Figure 1 implies $u_j = v_j$ for all $j \geq i$.

Thus, the above theorem is proved once we have shown the following lemma.

Lemma 22 *Let $\mathcal{G} = (V_1 \cup V_2, E, L)$ be a two-level graph containing the node $u \in V_1$, let m be the cardinality of $V_1 \cup V_2$, and let $k > m^2$ be such that $\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(\ell)}$ for all $\ell > k$. Then we have $\mathcal{G}_u^{(m^2)} \sqsubseteq \mathcal{G}_u^{(k)}$.*

Proof. By our assumption on k we know that $\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(2k)}$, i.e., there is a $(u_0^{(2k)}, u_0^{(k)})$-synchronized simulation Z such that $(u_0^{(2k)}, u_0^{(k)}) \in Z$. Without loss of generality we may assume that the corresponding selection function S is nice. As sketched in the previous section, the S-selected $(u_0^{(2k)}, u_0^{(k)})$-simulation chains can be encoded into an infinite tree.

To be more precise, let b be the maximal number of successors of a node in $\mathcal{G}$, and let $\mathcal{L}_{2k}$ ($\mathcal{L}_k$) be the set of all nodes up to level $2k$ (level k) of the infinite tree with branching factor b. Now, $\mathcal{G}_u^{(k)} \sqsubseteq \mathcal{G}_u^{(2k)}$ implies that the formula $\psi_{\mathcal{G}}^u$ is satisfiable with X replaced by $\mathcal{L}_{2k}$ and Y replaced by $\mathcal{L}_k$. We can use the sets assigned to the variables Q_l for $l \in (V_1 \cup V_2) \times (V_1 \cup V_2) \cup \{\sharp\}$ to label the nodes of the infinite tree with branching factor b by elements of $(V_1 \cup V_2) \times (V_1 \cup V_2) \cup \{\sharp\}$. Let t denote the labeled tree obtained this way. Our goal is to transform t into a new tree t' that encodes a $(u_0^{(k)}, u_0^{(m^2)})$-synchronized simulation containing $(u_0^{(k)}, u_0^{(m^2)})$. The main properties that this new tree must satisfy are:

1. If the node n of t' is labeled with an element of $(V_1 \cup V_2) \times V_1$, then n is of depth at most m^2.
2. If the node n of t' is labeled with $(u', v') \in V_1 \times (V_1 \cup V_2)$ and is of depth smaller than k, then its successor nodes must cover *all* the successors in $\mathcal{G}$ of u', i.e., not only the ones in V_2, but also the ones in V_1.
3. The synchronization property is satisfied, i.e., any infinite path in t' contains a node whose label is $\sharp$ or of the form (v', v') for some node $v' \in V_2$.

In order to satisfy the first property, we modify the tree t as follows. Assume that n is a node of t with label $(u', v') \in (V_1 \cup V_2) \times V_1$ that is on a level above m^2. By the definition of t, $v' \in V_1$ implies that n is at most at level k (since all such nodes must belong to $\mathcal{L}_k$). Now, consider the path in t from the root to n. Since this path is longer than m^2, there are two distinct nodes n_1, n_2 on this path such that their labels agree. Assume that n_1 comes before n_2 on this path. Then we replace the subtree at node n_1 by the subtree at node n_2.

We continue this replacement process until all nodes with a label in $(V_1 \cup V_2) \times V_1$ are on depth at most m^2. This process terminates since there were only finitely many such nodes in t (all of them have depth at most k), and the replacements do not increase the depth of a node, but strictly decrease the depth of at least one node with a label in $(V_1 \cup V_2) \times V_1$. In addition, since all nodes with a label in

$(V_1 \cup V_2) \times V_1$ are of depth at most k in t, the depth of a given node can decrease by at most k over the whole replacement process.

Let t' denote the labeled tree obtained this way. We claim that that t' encodes a $(u_0^{(k)}, u_0^{(m^2)})$-synchronized simulation that contains $(u_0^{(k)}, u_0^{(m^2)})$.

First, consider a node n with label (u', v') such that $v' \in V_1$. By our construction of t', this node is on depth at most m^2 in t'. Thus, if p is an infinite path in t', then the second components of the labels of the nodes on this path yield a path in $\mathcal{G}$ such that nodes in V_1 can only occur during the first m^2 steps of this path. Consequently, this path corresponds to a path in $\mathcal{G}_u^{(m^2)}$.

Second, consider a node n in t' whose label (u', v') belongs to $(V_1 \cup V_2) \times (V_1 \cup V_2)$. We need all the successors nodes of u' in $\mathcal{G}_u^{(k)}$ to be encoded by successor nodes of n. We know that (u', v') was the label of a node n' in t, and there all the "relevant" successor where encoded by successor nodes of n'. If $u' \in V_2$ or if n is of depth at least k, then it is easy to see that this implies that also n has all the relevant successor nodes. If $u' \in V_1$ and n is of depth smaller than k, then we also need all the successors nodes of u' in V_1 to be covered. It is easy to see that this is the case if n' (the original node in t with label (u', v')) was at depth smaller than $2k$ in t. However, we have already observed that the depth of a given node can decrease by at most k over the whole replacement process. Thus, the fact that n is of depth smaller than k in t' implies that n' was at depth smaller than $2k$ in t.

Third, recall that we can assume without loss of generality that the selection function used by the $(u_0^{(2k)}, u_0^{(k)})$-synchronized simulation that is encoded by t is nice. For the tree t this means the following: for every path p in t, there is a depth from which on p either contains only nodes with label $\sharp$ or p contains only nodes with labels of the form (v', v') for some $v' \in V_2$. Since our replacement process changes only finite prefixes of paths, this property is also satisfied by t', which shows that the synchronization property is still satisfied by t'. □

One might think that this polynomial bound on the cycle depth of a two-level graph can be used to show that the problem of deciding whether a graph is of bounded cycle depth or not can also be decided in polynomial time. However, this does not appear to be the case. In fact, assume that $\mathcal{G} = (V_1 \cup V_2, E, L)$ is a two-level graph with m nodes, and let $u \in V_1$. Then we know that $\mathcal{G}$ is of bounded cycle depth iff $\mathcal{G}_u^{(m^2)} \sqsubseteq \mathcal{G}_u^{(\ell)}$ for all $\ell > m^2$. However, testing this directly is still not possible since we would need to check infinitely many subsumption relationships. We could, of course, also try to use Theorem 21 to modify the reduction given in Section 4. However, all we would gain by this is that we could avoid the existential quantification over Y; the (expensive) universal quantification over X would still remain.

Theorem 21, together with the results in [2] and [3] (see Subsection 3.4), implies that the lcs (msc) in $\mathcal{EL}$ with descriptive semantics can be computed in polynomial time, provided that it exists.

Corollary 23 *1. Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox $\mathcal{T}$, and A, B concepts defined in $\mathcal{T}$. If the lcs of A, B in $\mathcal{T}$ w.r.t. descriptive semantics exists, then it can be computed in time polynomial in the size of $\mathcal{T}$.*

2. Let $\mathcal{T}$ be an $\mathcal{EL}$-TBox, $\mathcal{A}$ an $\mathcal{EL}$-ABox, and a an individual in $\mathcal{A}$. If the msc of a in $\mathcal{A}$ and $\mathcal{T}$ w.r.t. descriptive semantics exists then it can be computed in time polynomial in the size of $\mathcal{A}$ and $\mathcal{T}$.

6 Conclusion

We have introduced the notion "bounded cycle depth" of so-called two-level graphs, and have shown that the corresponding decision problem (i.e.: Given a two-level graph, is it of bounded cycle depth?) is decidable. In addition, we have shown that the cycle depth of a two-level graph of bounded cycle depth is polynomial in the size of the graph.

These results solve the two main problems that were left open in the previous papers [2, 3] on the lcs and the msc in $\mathcal{EL}$ with descriptive semantics. The existence of the lcs (msc) is decidable, and if it exists, then it can be computed in polynomial time.

What remains open is the exact complexity of the decision problems. Though this may seem unsatisfactory from a theoretical point of view, it is probably not very relevant in practice. In fact, independent of whether the lcs of A, B in $\mathcal{T}$ exists or not, the results in [2] show how to compute common subsumers P_i $(i \geq 0)$ of A, B in $\mathcal{T}$. The results of Section 5 show that we can compute a number k that is polynomial in the size of $\mathcal{T}$ such that A, B in $\mathcal{T}$ have an lcs w.r.t. descriptive semantics iff P_k is the lcs. Thus, we may just dispense with deciding whether the lcs exists, and return P_k. If the lcs exits, then P_k is the lcs. Otherwise, P_k is a common subsumer, and we can take it as an approximation of the lcs. The same is true for the msc.

Another interesting question is whether two-level graphs and the problem of deciding whether they are of bounded cycle depth also has applications in other areas.

References

[1] Franz Baader. Using automata theory for characterizing the semantics of terminological cycles. *Ann. of Mathematics and Artificial Intelligence*, 18:175–219, 1996.

[2] Franz 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] Franz Baader. The instance problem and the most specific concept in the description logic $\mathcal{EL}$ w.r.t. terminological cycles with descriptive semantics. In *Proceedings of the 26th Annual German Conference on Artificial Intelligence, KI 2003*, volume 2821 of *Lecture Notes in Artificial Intelligence*, pages 64–78, Hamburg, Germany, 2003. Springer-Verlag.

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

[5] Franz Baader. Terminological cycles in a description logic with existential restrictions. In Georg Gottlob and Toby Walsh, editors, *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 325–330. Morgan Kaufmann, 2003.

[6] Franz Baader and Ralf Küsters. Computing the least common subsumer and the most specific concept in the presence of cyclic $\mathcal{ALN}$-concept descriptions. In *Proc. of the 22nd German Annual Conf. on Artificial Intelligence (KI'98)*, volume 1504 of *Lecture Notes in Computer Science*, pages 129–140. Springer-Verlag, 1998.

[7] Franz Baader, Ralf Küsters, and Ralf Molitor. Computing least common subsumers in description logics with existential restrictions. In *Proc. of the 16th Int. Joint Conf. on Artificial Intelligence (IJCAI'99)*, pages 96–101, 1999.

[8] Monika R. Henzinger, Thomas A. Henzinger, and Peter W. Kopke. Computing simulations on finite and infinite graphs. In *36th Annual Symposium on Foundations of Computer Science*, pages 453–462, Milwaukee, Wisconsin, 1995. IEEE Computer Society Press.

[9] Yevgeny Kazakov and Hans de Nivelle. Subsumption of concepts in $\mathcal{FL}_0$ for (cyclic) terminologies with respect to descriptive semantics is pspace-complete. In *Proceedings of the 2003 International Workshop on Description Logics (DL2003)*, CEUR-WS, 2003.

[10] Ralf Küsters. *Non-standard Inferences in Description Logics*, volume 2100 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 2001.

[11] Bernhard Nebel. *Reasoning and Revision in Hybrid Representation Systems*, volume 422 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1990.

[12] Bernhard Nebel. Terminological cycles: Semantics and computational properties. In John F. Sowa, editor, *Principles of Semantic Networks*, pages 331–361. Morgan Kaufmann, Los Altos, 1991.

[13] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. of the Amer. Mathematical Society*, 141:1–35, 1969.

[14] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[15] K.A. Spackman. Normal forms for description logic expressions of clinical concepts in SNOMED RT. *J. of the American Medical Informatics Association*, pages 627–631, 2001. Symposium Supplement.

[16] K.A. Spackman, K.E. Campbell, and R.A. Cote. SNOMED RT: A reference terminology for health care. *J. of the American Medical Informatics Association*, pages 640–644, 1997. Fall Symposium Supplement.

[17] Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, volume B, chapter 4, pages 134–189. Elsevier Science Publishers (North-Holland), Amsterdam, 1990.