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, TU Dresden, D-01062 Dresden, Germany
baader@tcs.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

Description Logics (DLs) [6] are a class of knowledge representation formalisms in the tradition of semantic networks and frames, which can be used to represent the terminological knowledge of an application domain in a structured and formally well-understood way. DL systems provide their users with standard inference services (like subsumption and instance checking) that deduce implicit knowledge from the explicitly represented knowledge. More recently, non-standard inferences [8] 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

[*] Partially supported by DFG (BA 1122/4-3) and by National ICT Australia Limited.

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 motivation for the graph-theoretic problem solved in the present paper comes from non-standard inferences in the DL $\mathcal{EL}$, which is rather inexpressive, but nevertheless has significant applications. For example, SNOMED, the Systematized Nomenclature of Medicine [11, 10] employs $\mathcal{EL}$. Unfortunately, the most specific concept of a given individual need not exist in $\mathcal{EL}$. For other DLs, this problem had been overcome by allowing for cyclic concept definitions [7]. In order to adapt this approach also to $\mathcal{EL}$, 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 [4, 3, 1, 2] that gives an almost complete picture of the computational properties of the above mentioned standard and non-standard inferences in $\mathcal{EL}$ with cyclic concept definitions.[1] Regarding standard inferences, the subsumption and the instance problem turned out to be polynomial for both types of semantics. Regarding non-standard inferences, w.r.t. gfp-semantics the lcs and the msc always exist and can be computed in polynomial time. Descriptive semantics is less well-behaved. In [1] it was shown that, in general, the lcs need not exist. The paper gave a characterization for the existence of the lcs, but the question of how to decide this condition remained open. In [2], analogous results were shown for the msc.

The present paper introduces a common graph-theoretic generalization of these open problems: the problem whether a so-called two-level graph is of bounded cycle depth. Then it shows that this problem is decidable by reducing it to monadic second-order logic on infinite trees [9]. Finally, it shows that, if a two-level graph is of bounded cycle depth, then its cycle depth is polynomially bounded by the size of the graph. This implies that the lcs/msc can be computed in polynomial time, provided that it exists.

Because of the space constraints, we concentrate on the graph-theoretic problems. The reader is referred to [6] for more information on DLs in general, to [4, 3, 1, 2] for previous results on $\mathcal{EL}$ with cyclic definitions, and to [5] for a long version of this paper containing full proofs and the connection to the lcs/msc.

2 The cycle depth of two-level graphs

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

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.

[1] Cyclic definitions in $\mathcal{EL}$ can either be interpreted with greatest fixpoint (gfp) or with descriptive semantics, which is the usual first-order semantics for DLs.

$$\begin{tikzcd}
p_1 : & u = u_0 \arrow[r, "r_1"] \arrow[d, "Z"'] & u_1 \arrow[r, "r_2"] \arrow[d, "Z"'] & u_2 \arrow[r, "r_3"] \arrow[d, "Z"'] & u_3 \arrow[r, "r_4"] \arrow[d, "Z"'] & \cdots \\
p_2 : & v = v_0 \arrow[r, "r_1"] & v_1 \arrow[r, "r_2"] & v_2 \arrow[r, "r_3"] & v_3 \arrow[r, "r_4"] & \cdots
\end{tikzcd}$$

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

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

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

Definition 1. *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 $(v_2, r, v_2') \in E$.*

Here, we are not interested in arbitrary simulations containing a given pair of nodes, 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 2. *Let $\mathcal{G}$ be a graph, Z a simulation on $\mathcal{G}$, and $(u, v) \in Z$.*
(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 v_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 $u = u_0 \xrightarrow{r_1} u_1 \xrightarrow{r_2} u_2 \xrightarrow{r_3} u_3 \xrightarrow{r_4} \cdots$, one can use the selection function S to construct a simulating path. In this case we say that the resulting infinite (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 $u_i = v_i$.*

As shown in [4, 2], the subsumption and the instance problem in $\mathcal{EL}$ can be reduced to the problem of deciding whether there exists a synchronized simulation on a given graph (which is a problem decidable in polynomial time [4]).

To define the main graph-theoretic problem addressed in this paper, we must first introduce two-level graphs.

Definition 3. *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 $\mathcal{G}_2$. Next, we consider graphs obtained from $\mathcal{G}$ by unraveling cycles in $\mathcal{G}_1$ up to a certain length.

Definition 4. *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*

$$\begin{aligned}
V_1^{(k)} &:= \{u_0^{(k)}\} \cup \{v_i^{(k)} \mid v \in V_1 \textit{ and } 1 \leq i \leq k\};\\
E^{(k)} &:= \{(v, r, w) \mid (v, r, w) \in E \textit{ and } v, w \in V_2\} \;\cup\\
&\quad\;\; \{(v_i^{(k)}, r, w_{i+1}^{(k)}) \mid (v, r, w) \in E \textit{ and } v_i^{(k)}, w_{i+1}^{(k)} \in V_1^{(k)}\} \cup\\
&\quad\;\; \{(v_i^{(k)}, r, w) \mid (v, r, w) \in E \textit{ and } v_i^{(k)} \in V_1^{(k)}, w \in V_2\};
\end{aligned}$$

$$\begin{aligned}
L^{(k)}(v) &:= L(v) \quad \textit{if } v \in V_2,\\
L^{(k)}(v_i^{(k)}) &:= L(v) \quad \textit{if } v_i^{(k)} \in V_1^{(k)}.
\end{aligned}$$

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.[2]

Definition 5. *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 [2]). Given a node $u \in V_1$ of a two-level graph $\mathcal{G} = (V_1 \cup V_2, E, L)$, we are interested in finding an index k such that the subsumption relationship also holds in the other direction.

Definition 6. *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 $u \in V_1$.

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

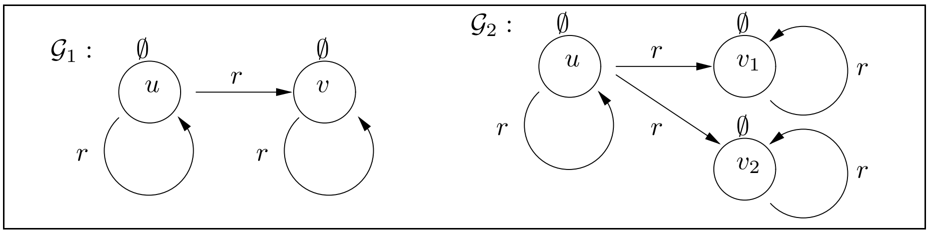

Fig. 3. Two two-level graphs, one of bounded and one of unbounded cycle depth.

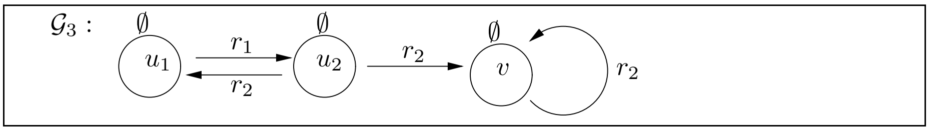

Fig. 4. Another two-level graph of unbounded cycle depth.

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

Before stating 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 6 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 $\mathcal{G}_{3,u_1}^{(k)} \not\sqsubseteq \mathcal{G}_{3,u_1}^{(k+1)}$ if k is even. First, assume that k is odd. Then there are no infinite paths in $\mathcal{G}_{3,u_1}^{(k+1)}$ that use the node $u_{1,k+1}^{(k+1)}$ since this node does not have a successor node. As an easy consequence, every infinite path in $\mathcal{G}_{3,u_1}^{(k+1)}$ can be simulated by "the same" path in $\mathcal{G}_{3,u_1}^{(k)}$. In addition, the finite path to $u_{1,k+1}^{(k+1)}$ can be simulated by a path in $\mathcal{G}_{3,u_1}^{(k)}$ that ends with v. Consequently, $\mathcal{G}_{3,u_1}^{(k)} \sqsubseteq \mathcal{G}_{3,u_1}^{(k+1)}$

for odd k. In contrast, if k is even, then $u_{1,k}^{(k+1)}$ has a successor node in $\mathcal{G}_{3,u_1}^{(k+1)}$ (namely $u_{2,k+1}^{(k+1)}$) reached by an edge with label r_1. Any node reachable from $u_{1,0}^{(k)}$ in $\mathcal{G}_{3,u_1}^{(k)}$ by a path of length k (i.e., $u_{1,k}^{(k)}$ or v) does not have a successor w.r.t. r_1. Thus, there is a path in $\mathcal{G}_{3,u_1}^{(k+1)}$ that cannot be simulated by a path in $\mathcal{G}_{3,u_1}^{(k)}$, which shows that $\mathcal{G}_{3,u_1}^{(k)} \not\sqsubseteq \mathcal{G}_{3,u_1}^{(k+1)}$ for even k.

The last example shows that, in order to find the number k required by Definition 6, one can*not* simply test subsumption between $\mathcal{G}_u^{(i+1)}$ and $\mathcal{G}_u^{(i)}$ for $i = 0, 1, 2, \ldots$ until $\mathcal{G}_u^{(i)} \sqsubseteq \mathcal{G}_u^{(i+1)}$, and then stop with output $k = i$.

The characterization of the lcs and the msc given in [1] and [2], respectively, can easily be reformulated in terms of the notions introduced above. As an easy consequence, the existence problem can be reduced to the main decision problem introduced in this paper (see [5] for detail).

Proposition 1. *The problems of deciding the existence of the lcs (msc) in $\mathcal{EL}$ with descriptive semantics can be reduced in polynomial time to the problem of deciding whether a two-level graph $\mathcal{G}$ is of bounded cycle depth. In addition, if the cycle depth of $\mathcal{G}$ is polynomial in the size of $\mathcal{G}$, then the lcs (msc) can be computed in polynomial time.*

3 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 $\phi_{\mathcal{G}}^u$ of monadic second-order logic (MSO) on infinite trees is satisfiable. As shown by Rabin [9], 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., [12] for an introduction). Before we define the formula $\phi_{\mathcal{G}}^u$, 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 2) 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 1) 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 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 [4] 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 $\mathcal{G}_u^{(k)}$ and $\mathcal{G}_u^{(\ell)}$ 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 $\mathcal{G}_u^{(k)}$ and $\mathcal{G}_u^{(\ell)}$ and the quantification on k and ℓ. Assume that we have an infinite tree t encoding a (u, u)-synchronized simulation Z on the two-level graph $\mathcal{G}$, 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 in $\mathcal{G}$ 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.[3] 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 tree 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

[3] Since then one could also express that two nodes are on the same level, which is know to be inexpressible in MSO [12].

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 ϕ^u_G is defined as

$$\exists Y.(\mathit{PrefixClosed}(Y) \wedge \mathit{Finite}(Y) \wedge \forall X.(\mathit{PrefixClosed}(X) \wedge \mathit{Finite}(X) \Rightarrow \psi^u_G)),$$

where *PrefixClosed*(.) and *Finite*(.) are the well-known MSO-formulae expressing that a set of nodes is prefix-closed and finite, respectively,[4] and ψ^u_G 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 ϑ^u_G of the following formulae:

- *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).

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

- *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

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

[4] Defining *PrefixClosed*(.) is a simple exercise. A definition of *Finite*(.) can be found in [12].

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.*

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

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 [12]).

Lemma 1. *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.*

Since satisfiability in MSO on infinite trees is decidable, the lemma (whose proof can be found in [5]) implies decidability of bounded cycle depth.

Theorem 1. *The problem of deciding whether a two-level graph is of bounded cycle depth w.r.t. one of its nodes is decidable.*

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 $\phi^u_\mathcal{G}$ contains several quantifier changes.[5]

Together with Propositions 1, this theorem implies:

Corollary 1. *The existence of the lcs and the msc is decidable in $\mathcal{EL}$ with descriptive semantics.*

[5] 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.

4 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 2. *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 $\mathcal{G}_u^{(m^2)} \sqsubseteq \mathcal{G}_u^{(k)}$. The fact that is enough 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 (see [5]).

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

Lemma 2. *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 $(u_0^{(2k)}, u_0^{(k)}) \in Z$. Let S be the corresponding selection function. 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. Then we can show that t' satisfies the properties 1, 2, 3 mentioned above, and thus encodes a $(u_0^{(k)}, u_0^{(m^2)})$-synchronized simulation that contains $(u_0^{(k)}, u_0^{(m^2)})$ (see [5] for details). □

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 2 to modify the reduction given in Section 3. 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.

Together with Propositions 1, Theorem 2 implies:

Corollary 2. *The lcs (msc) in $\mathcal{EL}$ with descriptive semantics can be computed in polynomial time, provided that it exists.*

5 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 [1, 2] 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 the concepts A, B defined in a terminology $\mathcal{T}$ exists or not, the results in [1] show how to compute common subsumers P_i $(i \geq 0)$ of A, B in $\mathcal{T}$. The results of Section 4 imply 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. Is the cycle depth of a two-level graph an artifact of the characterization of the lcs and the msc in $\mathcal{EL}$ with descriptive semantics given in [1, 2], or is it a natural notion that is of interest in its own right?

References

1. F. Baader. Computing the least common subsumer in the description logic $\mathcal{EL}$ w.r.t. terminological cycles with descriptive semantics. In *Proc. ICCS 2003*, Springer LNAI 2746, 2003.
2. F. Baader. The instance problem and the most specific concept in the description logic $\mathcal{EL}$ w.r.t. terminological cycles with descriptive semantics. In *Proc. KI 2003*, Springer LNAI 2821, 2003.
3. F. Baader. Least common subsumers and most specific concepts in a description logic with existential restrictions and terminological cycles. In *Proc. IJCAI 2003*, Morgan Kaufmann, 2003.
4. F. Baader. Terminological cycles in a description logic with existential restrictions. In *Proc. IJCAI 2003*, Morgan Kaufmann, 2003.
5. F. Baader. A graph-theoretic generalization of the least common subsumer and the most specific concept in the description logic $\mathcal{EL}$. LTCS-Report 04-02, TU Dresden, Germany, 2004. See http://lat.inf.tu-dresden.de/research/reports.html.
6. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P.F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.
7. 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. KI'98*, Springer LNAI 1504, 1998.
8. R. Küsters. *Non-standard Inferences in Description Logics*, Springer LNAI 2100, 2001.
9. M.O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. of the Amer. Mathematical Society*, 141, 1969.
10. K.A. Spackman. Normal forms for description logic expressions of clinical concepts in SNOMED RT. *J. of the American Medical Informatics Association*, 2001. Symposium Supplement.
11. 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*, 1997. Fall Symposium Supplement.
12. W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, Volume B. Elsevier Science Publishers, Amsterdam, 1990.