Structural Subsumption Considered from an Automata-Theoretic Point of View

F. Baader, R. Küsters, and R. Molitor
LuFg Theoretical Computer Science, RWTH Aachen, Germany,
email: {baader, kuesters, molitor}@informatik.rwth-aachen.de

Abstract. This paper compares two approaches for deriving subsumption algorithms for the description logic $\mathcal{ALN}$: structural subsumption and an automata-theoretic characterization of subsumption. It turns out that structural subsumption algorithms can be seen as special implementations of the automata-theoretic characterization.

1 Introduction

Structural subsumption algorithms are efficient methods for deciding subsumption in description logics without full negation, disjunction, and existential restrictions. The structural subsumption algorithm employed by the system Classic [4, 5] is based on a specific data structure for representing concept descriptions, called *description graphs*. The subsumption problem is reduced to a structural comparison of description graphs.

Another approach for deciding subsumption in sublanguages of Classic can be obtained from the automata-theoretic characterizations of subsumption w.r.t. greatest fixed-point (gfp) semantics in cyclic terminologies [1, 7], which reduce the subsumption problem to an inclusion problem for certain *regular languages*. In the case of acyclic terminologies (and thus in particular for concept descriptions), these languages turn out to be finite.

At first sight, there is no connection between these two approaches since they are based on rather different normal forms for concept descriptions. Intuitively speaking, structural subsumption is based on a normal form that applies the equivalence $\forall R.(A \sqcap B) \equiv \forall R.A \sqcap \forall R.B$ as a rewrite rule from right to left, i.e., the descriptions are grouped w.r.t. role names, whereas the finite languages considered in the automata-theoretic approach correspond to a normal form obtained by applying the above equivalence from left to right.

Another difference between the two approaches is that they describe decision procedures for subsumption on two different levels of abstraction. The structural subsumption algorithm for Classic is presented in [4, 5] on the level of the data structure (namely, description graphs) used in the implementation. This provides a description of the algorithm that is very close to its actual implementation. Consequently, both the formal description of the algorithm and the proof of its correctness are quite complex [4, 5, 8]. In contrast, the automata-theoretic approach reduces the subsumption problem to a formal language problem (namely, inclusion of finite or regular languages), which means that the description of the subsumption algorithm (and thus also the proof of its correctness) can be split into two independent parts: (i) the characterization of subsumption on the abstract formal language level, and (ii) an algorithm that decides the formal language problem.

The goal of this paper is to show that structural subsumption algorithms based on description graphs can be seen as "parallel" implementations of the language inclusion tests required by the automata-theoretic characterization of subsumption. We illustrate the connection between the two approaches starting with the small language $\mathcal{FL}_0$, which allows for value restrictions and conjunction, and then extend the comparison to $\mathcal{ALN}$, which additionally provides us with atomic negation and number restrictions.

2 The automata-theoretic approach

In order to obtain the automata-theoretic characterization of subsumption in $\mathcal{FL}_0$, one translates (possibly cyclic) $\mathcal{FL}_0$-terminologies T into corresponding finite automata $\mathcal{A}_T$ (with ε-transitions). The concept names in T are the states of $\mathcal{A}_T$, and the transitions of $\mathcal{A}_T$ are induced by the value restrictions in T (see [1] for details). For a defined concept A and a primitive concept P in T, the language $L_{\mathcal{A}_T}(A, P)$ is the set of all words labeling paths in $\mathcal{A}_T$ from A to P. The languages $L_{\mathcal{A}_T}(A, P)$ represent all the value restrictions that must be satisfied by instances of the concept A.

To apply this approach to the subsumption problem for concept descriptions, we must first translate a given concept description C into a corresponding (acyclic) $\mathcal{FL}_0$-terminology T_C. In principle, this is achieved by introducing new concept names and corresponding concept definitions for all sub-descriptions occurring inside value restrictions (see [3] for details).

Example 1.
Consider the concept descriptions $C := \forall R.P \sqcap \forall R.Q \sqcap \forall R.\forall S.P \sqcap \forall S.Q$ and $D := \forall R.\forall S.\forall R.P \sqcap \forall S.Q$. These descriptions can be represented by the defined concepts A and B in the following acyclic $\mathcal{FL}_0$-terminologies:

$$\begin{aligned} T_1\colon A &= \forall R.A_1 \sqcap \forall R.A_2 \sqcap \forall R.A_3 \sqcap \forall S.A_4, \\ A_1 &= P,\ A_2 = Q,\ A_4 = Q \\ A_3 &= \forall S.A_{31},\ A_{31} = P. \\ T_2\colon B &= \forall R.B_1 \sqcap \forall S.B_2,\ B_2 = Q \\ B_1 &= \forall S.B_{11},\ B_{11} = \forall R.B_{111},\ B_{111} = P. \end{aligned}$$

The terminologies T_1 and T_2 yield the automata $\mathcal{A}_1$ and $\mathcal{A}_2$ of Fig. 1. In this example, we have $L_{\mathcal{A}_1}(A,P) = \{R, RS\}$, $L_{\mathcal{A}_1}(A,Q) = \{R,S\}$, $L_{\mathcal{A}_2}(B,P) = \{RSR\}$, and $L_{\mathcal{A}_2}(B,Q) = \{S\}$.

Informally, the characterization of subsumption in $\mathcal{FL}_0$ [1] says that the defined concept A is subsumed by the defined concept B iff the set of all value restrictions that must be satisfied by A is a superset of the set of value restrictions that must be satisfied by B. Using the automata introduced above, this can be stated more formally (as a special case of the results in [1]) as follows:

Theorem 2.
Let T_1, T_2 be acyclic $\mathcal{FL}_0$-terminologies not sharing any defined concept names, let $\mathcal{A}_1, \mathcal{A}_2$ be the corresponding automata, and let A and B be defined in T_1 and T_2, respectively. Then $A \sqsubseteq_{T_1 \cup T_2} B$ iff $L_{\mathcal{A}_1}(B,P) \subseteq L_{\mathcal{A}_2}(A,P)$ for all primitive concepts P. □

In Example 1, we have $L_{\mathcal{A}_2}(B,P) = \{RSR\} \not\subseteq L_{\mathcal{A}_1}(A,P)$, and therefore $A \not\sqsubseteq B$. In order to decide subsumption based on this approach, one must decide the inclusion problem for the corresponding languages.

3 Deciding inclusion of regular languages

In automata theory, the inclusion problem for regular languages is usually reduced to the emptiness problem: $L_2 \subseteq L_1$ iff $L_2 \cap \overline{L}_1 = \emptyset$. In order to construct an automaton for $L_2 \cap \overline{L}_1$ from given (nondeterministic) automata for L_1, L_2, one first computes a deterministic automaton for L_1 by applying the powerset construction. For the automaton $\mathcal{A}_1$ of our example, the powerset construction yields the automaton $\mathcal{P}(\mathcal{A}_1)$ (see Fig. 1). This automaton accepts the complement $\overline{L_{\mathcal{A}_1}(A,P)}$ of the language $L_{\mathcal{A}_1}(A,P)$ if we make $\{A\}$ the initial state and all states *not* containing P the final states. Thus, we have $L_{\mathcal{A}_2}(B,P) \cap \overline{L_{\mathcal{A}_1}(A,P)} \neq \emptyset$ iff there exists a word W such that (1) there is a path with label W in $\mathcal{P}(\mathcal{A}_1)$ leading from $\{A\}$ to a state not containing P, and (2) there is a path with label W in $\mathcal{A}_2$ leading from B to P. In the example, RSR is such a word.

In the general case, the existence of such a word can be decided in time polynomial in the size of $\mathcal{P}(\mathcal{A}_1)$ and $\mathcal{A}_2$. It should be noted that, even for acyclic terminologies, the powerset automaton $\mathcal{P}(\mathcal{A}_1)$ may be exponential in the size of $\mathcal{A}_1$ (see Nebel's coNP-hardness result for subsumption w.r.t. acyclic $\mathcal{FL}_0$-terminologies [9]). However, for terminologies constructed from concept descriptions (as in our example) it can be shown that this exponential blow-up cannot occur [3].

Instead of testing the required inclusion relationships separately for each primitive concept, one can realize this test in parallel, i.e., we look for a primitive concept P and a word W such that (1) and (2) from above hold. We shall show below that this is exactly what the structural subsumption algorithms based on description graphs do.

4 Structural subsumption algorithms based on description graphs

Description graphs are rooted directed acyclic graphs whose nodes are labeled by sets of primitive concepts and whose edges are labeled by roles. Concept descriptions can be turned into description graphs by a straightforward translation of the syntactic structure of the descriptions.

Example 3 (Example 1 continued).
Fig. 2 shows the description graphs $\mathcal{G}_C$ and $\mathcal{G}_D$ corresponding to the descriptions C, D of Example 1. The label $\emptyset$ at the root of $\mathcal{G}_C$ expresses that no primitive concept occurs in the top-level conjunction of C. The edge labeled S from the root to the node labeled Q says that there is a value restriction $\forall S.C'$ in the top-level conjunction of C such that Q is the only primitive concept occurring in the top-level conjunction of C', etc.

Before we can decide whether C is subsumed by D based on a structural comparison of the description graphs, the graph for the subsumee C must be normalized by merging successor nodes reached by edges labeled by the same role name. (This corresponds to applying the rewrite rule $\forall R.A \sqcap \forall R.B \to \forall R.(A \sqcap B)$ to the descriptions.) In our example, the *canonical* description graph $\widehat{\mathcal{G}}_C$ obtained this way is also shown in Fig. 2.

The structural comparison of description graphs is formalized in Theorem 4, based on the notion of more specific paths, as introduced in [5]. The rooted path p (i.e., a path starting at the root) in $\mathcal{G}_C$ is *more specific* than the rooted path q in $\mathcal{G}_D$ iff (1) the word over the alphabet of role names labeling q is a prefix of the word labeling p (i.e., p may be longer than q), and (2) the node labels in p are supersets of the corresponding node labels in q.

Theorem 4 (Structural subsumption [5]).
Let C, D be $\mathcal{FL}_0$-concept descriptions, $\widehat{\mathcal{G}}_C$ the canonical description graph of C and $\mathcal{G}_D$ the description graph of

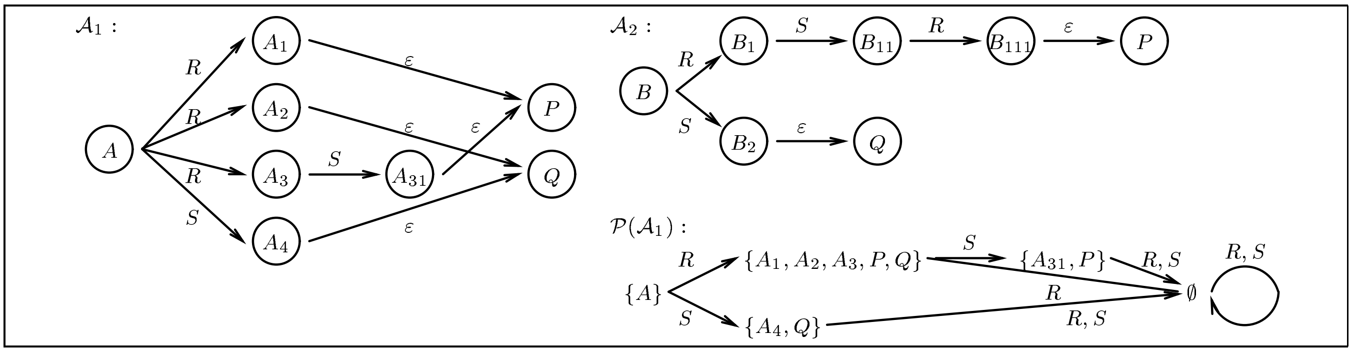

Figure 1: The automata corresponding to C and D.

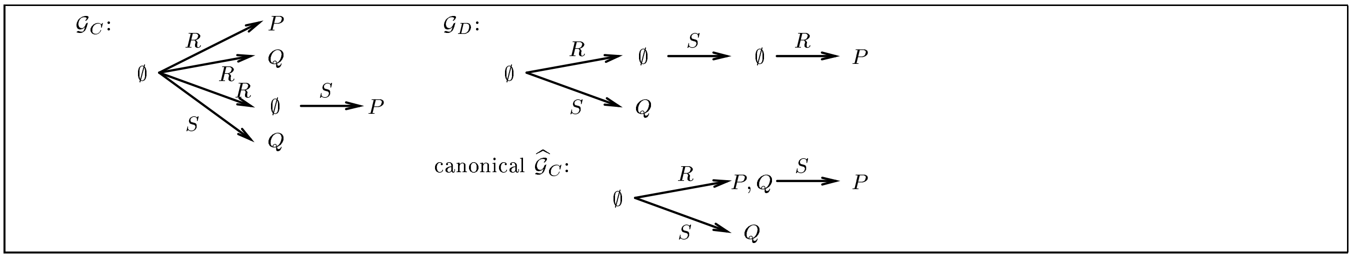

Figure 2: The description graphs corresponding to C and D.

D. Then $C \sqsubseteq D$ iff for each rooted path q in $\mathcal{G}_D$ there exists a more specific rooted path p in $\widehat{\mathcal{G}}_C$. $\square$

As an example, consider the description graphs in Fig. 2. The path with label RS in $\widehat{\mathcal{G}}_C$ is more specific than the path with label RS in $\mathcal{G}_D$. However, for the path with label RSR in $\mathcal{G}_D$ there does not exist a more specific path in $\widehat{\mathcal{G}}_C$. Consequently, the structural subsumption test recognizes that C is not subsumed by D.

5 Comparing the approaches

If we compare Fig. 1 with Fig. 2, then we see that the description graphs $\mathcal{G}_C$ and $\mathcal{G}_D$ essentially agree with the automata $\mathcal{A}_1$ and $\mathcal{A}_2$. The only difference is that in the automata there is only one state for every primitive concept, and the primitive concepts (here P, Q) are reached via ε-transition from intermediate defined concepts (e.g., A_1). To make this observation more precise, we can show that there exists an isomorphism between the automaton corresponding to a concept description C and the description graph $\mathcal{G}_C$ [3]. This "isomorphism" is a bijective mapping φ from the nodes of $\mathcal{G}_C$ onto the set of states corresponding to defined concepts in the automaton. This mapping is such that the label of node v consists of exactly those primitive concepts reached from $\varphi(v)$ via ε-transitions.

Another obvious similarity between the automata-theoretic and the structural approach is that in both cases the automaton/graph for the subsumee C must be modified. A closer look at Fig. 1 and 2 reveals that the powerset automaton $\mathcal{P}(\mathcal{A}_1)$ and the canonical description graph $\widehat{\mathcal{G}}_C$ are also essentially identical. To be more precise, $\widehat{\mathcal{G}}_C$ can be obtained from $\mathcal{P}(\mathcal{A}_1)$ by (1) removing the names of defined concepts from the node labels, and (2) by removing the sink state $\emptyset$ and the edges leading to this sink.

Because of this similarity between the automata and the description graphs, structural subsumption on description graphs can be seen as a special implementation of the language inclusion tests required by the automata-theoretic characterization of subsumption of concept descriptions. To make this more precise, let C, D be concept descriptions and assume that $C \not\sqsubseteq D$. By Theorem 4, the structural subsumption algorithm detects non-subsumption by finding a rooted path q in $\mathcal{G}_D$ with label W such that there does not exist a more specific rooted path p in $\widehat{\mathcal{G}}_C$. There are two possible reasons why this more specific path does not exist in $\widehat{\mathcal{G}}_C$:

(1) There is no path with label W in $\widehat{\mathcal{G}}_C$. Without loss of generality we may assume that the path q in $\mathcal{G}_D$ ends in a node with non-empty label set (otherwise, the path could be extended appropriately). Assume that the primitive concept P is contained in this label set. Then we have $W \in L_{\mathcal{A}_2}(B, P) \cap \overline{L_{\mathcal{A}_1}(A, P)}$. In fact, $W \in L_{\mathcal{A}_2}(B, P)$ because the path q in $\mathcal{G}_D$ to a node containing P yields a path in $\mathcal{A}_2$ from B to P. The fact that there is no (rooted) path with label W in $\widehat{\mathcal{G}}_C$ implies that the path with label W in $\mathcal{P}(\mathcal{A}_1)$ leads from the initial state to the sink state $\emptyset$. Since $\emptyset$ does not

contain P, it is a final state for the automaton accepting $\overline{L_{\mathcal{A}_1}(A,P)}$.

(2) For q and the (unique) path p with label W in $\widehat{\mathcal{G}}_C$, the inclusion condition between the labels is violated by some primitive concept P, i.e., P belongs to the label of a node in q, but not to the label of the corresponding node in p. An argument similar to the one employed in the first case can be used to show the following: if U is the prefix of W leading to the node where the inclusion between labels is violated because of P, then $U \in L_{\mathcal{A}_2}(B,P) \cap \overline{L_{\mathcal{A}_1}(A,P)}$.

To sum up, we have shown that the existence of a rooted path in $\mathcal{G}_D$ without a more specific path in $\widehat{\mathcal{G}}_C$ implies that there is a primitive concept P such that $L_{\mathcal{A}_2}(B,P) \not\subseteq L_{\mathcal{A}_1}(A,P)$. The converse of this implication can be shown analogously.

6 Extending the comparison to $\mathcal{ALN}$

In both approaches, number restrictions and negated primitive concepts are treated like new primitive concepts. In the automata-theoretic approach, they give rise to new states in the automaton of C, and to additional inclusion conditions. In the description graphs, they may also occur in node labels. However, since both primitive negation and number restrictions may cause inconsistencies, this straightforward extension is not sufficient to obtain a complete subsumption algorithm.

Example 5.
If we treat primitive negations and number restrictions in the $\mathcal{ALN}$-concept description $C' := \forall S.Q \sqcap \forall R.(P \sqcap Q \sqcap \forall S.\forall S.(Q \sqcap \neg Q) \sqcap \forall S.(\geq 1\ S))$ like primitive concepts, then we obtain the powerset automaton $\mathcal{P}(\mathcal{A}_3)$ and the $\mathcal{FL}_0$-canonical description graph $\widehat{\mathcal{G}}_{C'}$ depicted in Fig. 3.

For the concept description C of Example 1 we have $C' \sqsubseteq C$, even though (1) $RS \in L_{\mathcal{A}_1}(A,P) \cap \overline{L_{\mathcal{P}(\mathcal{A}_3)}(A',P)}$, and (2) the path with label RS in $\mathcal{G}_C$ does not have a corresponding more specific path in $\widehat{\mathcal{G}}_{C'}$, i.e., neither the automata-theoretic approach nor the structural approach detects the subsumption relationship.

The automata-theoretic approach for $\mathcal{ALN}$
For an $\mathcal{FL}_0$-terminology T, the language $L_{\mathcal{A}_T}(A',P)$ represents exactly those value restrictions on P that subsume A', i.e., $A' \sqsubseteq_T \forall W.P$ iff $W \in L_{\mathcal{A}_T}(A',P)$.[1] Since the inconsistent concept $\bot$ is expressible in $\mathcal{ALN}$, the language $L_{\mathcal{A}_T}(A',P)$ is no longer sufficient to capture all these value restrictions. In addition, one must consider so-called *A'-excluding words*, i.e., words W such that $A' \sqsubseteq_T \forall W.\bot$. A formal definition of the set $E(A')$ of A'-excluding words can be found in [7]. Here, we just illustrate it using our example. Obviously, $RSS \in E(A')$ since this word leads to a state (in the powerset automaton) that contains both Q and $\neg Q$, i.e., any RSS-successor of an individual in A' must belong both to Q and $\neg Q$, which is impossible. In addition, $RSSU \in E(A')$ for all words U, since the existence of an $RSSU$-successor would imply the existence of an RSS-successor. Finally, at-least restrictions can also force prefixes of RSS to belong to $E(A')$: since RS leads to a state containing $(\geq 1\ S)$, every RS-successor of an individual in A' also has an RSS-successor; however, since $RSS \in E(A')$ means that individuals in A' cannot have RSS-successors, this implies that they cannot have RS-successors.

Since $\forall W.\bot$ is subsumed by $\forall W.P$, A'-excluding words yield additional value restrictions that are not explicitly represented by $L_{\mathcal{A}_T}(A',P)$. Thus, in order to represent all value restrictions that are satisfied by instances of A', we consider $L_{\mathcal{A}_T}(A',P) \cup E(A')$ instead of $L_{\mathcal{A}_T}(A',P)$. In our example, the inclusion condition "$L_{\mathcal{A}_1}(A,P) \subseteq L_{\mathcal{P}(\mathcal{A}_3)}(A',P)$" for P is replaced by "$L_{\mathcal{A}_1}(A,P) \subseteq L_{\mathcal{P}(\mathcal{A}_3)}(A',P) \cup E(A')$." Consequently, RS no longer violates the inclusion condition for P.

Structural subsumption algorithms for $\mathcal{ALN}$
Structural subsumption algorithms deal with the problems caused by inconsistencies by applying additional normalization rules when computing the canonical description graph [4, 5, 8]. These additional rules must take care of nodes labeled by inconsistent sets, i.e., nodes whose label contain $\{P, \neg P\}$ for some primitive concept P or $\{(\geq l\ S), (\leq r\ S)\}$ for numbers $l > r$. Such inconsistent nodes and the edges leading to these nodes are removed. In addition, if there was an edge labeled R from node v to the inconsistent node, the label of v is extended by $(\leq 0\ R)$. This is due to the equivalence $\forall R.\bot \equiv (\leq 0\ R)$. For the same reason, we remove subgraphs with root v if the label of the R-predecessor of v contains $(\leq 0\ R)$.

Example 6 (Example 5 continued).
Applying these rules to the $\mathcal{FL}_0$-canonical description graph $\widehat{\mathcal{G}}_{C'}$ in Fig. 3 yields the $\mathcal{ALN}$-canonical description graph $\tilde{\mathcal{G}}_{C'}$ depicted in the same figure: Since v_3' is labeled by $\{Q, \neg Q\}$, the node v_3' and the edge $v_2' S v_3'$ are removed, and $(\leq 0\ S)$ is added to the label of v_2'. Consequently, v_2' is now labeled by $\{(\geq 1\ S), (\leq 0\ S)\}$, which is again inconsistent. Thus, v_2' and the edge leading to it are removed, and $(\leq 0\ S)$ is added to the label of v_1'.

The definition of more specific paths must be adapted as well. In fact, due to number restrictions of the form $(\leq 0\ R)$, even a path p that is shorter than q may be more specific than q. To be more precise, if the label of the last node v in p contains $(\leq 0\ R)$, then all value restrictions on R expressed by an R-successor of the node corresponding to v in q are trivially satisfied. In our example, the path with label R in $\tilde{\mathcal{G}}_{C'}$ (Fig. 3) is thus

[1] $\forall R_1 \cdots R_k.P$ abbreviates $\forall R_1. \cdots \forall R_k.P$.

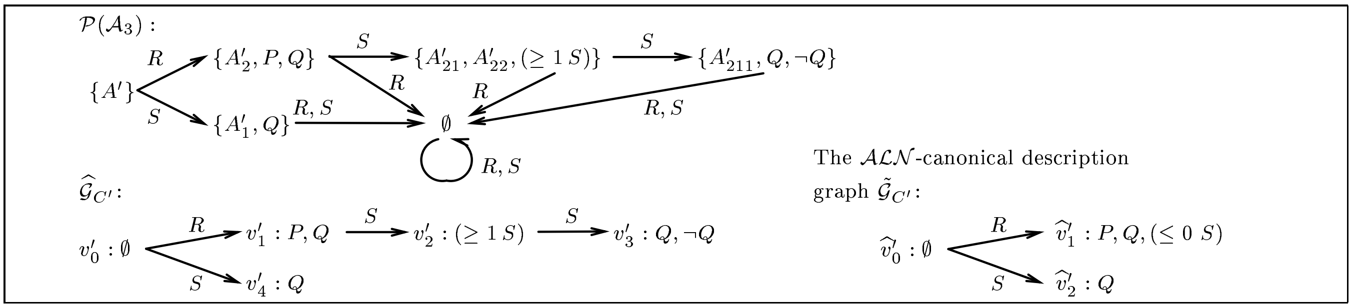

Figure 3: The powerset automaton and canonical description graph of C'.

more specific than the one with label RS in $\mathcal{G}_C$ (Fig. 2). Consequently, the structural subsumption algorithm for $\mathcal{ALN}$ also recognizes $C' \sqsubseteq C$.

Comparing the approaches

A formal description of both approaches to subsumption of $\mathcal{ALN}$-concept descriptions as well as a detailed comparison can be found in [3]. Here, we can only point out some of the main aspects of this comparison.

As before, there exists an "isomorphism" between the automaton $\mathcal{A}$ and the description graph $\mathcal{G}$ corresponding to an $\mathcal{ALN}$-concept description C. Furthermore, the correspondence between the $\mathcal{FL}_0$-canonical description graph $\widehat{\mathcal{G}}$ and the deterministic automaton $\mathcal{P}(\mathcal{A})$ also carries over.

In addition to this close relationship between the data structures employed by the two approaches, there is also a 1–1-correspondence between the additional normalization rules on the one hand, and the definition of the set of A'-excluding words on the other hand. When applying the normalization rules to the graph $\widehat{\mathcal{G}}_{C'}$ in Example 6, we have added $(\leq 0\; S)$ first to the label of the RS-successor and then to the label of the R-successor of the root. Both RSS and RS are A'-excluding words. More generally, we show in [3] that adding $(\leq 0\; S)$ to the label of the W-successor of the root in $\widehat{\mathcal{G}}_C$ corresponds to the fact that WS is an A'-excluding word. Thus, adding the set $E(A')$ to the right-hand side of the inclusion statements in the automata-theoretic approach corresponds to the additional normalization steps and the extended definition of more specific paths employed by the structural approach.

7 Conclusion and future work

We have shown that structural subsumption algorithms are special implementations of the language inclusion tests required by the automata-theoretic characterization of subsumption. This provides us with a more abstract understanding of how structural subsumption algorithms work. We will extend this comparison to cyclic terminologies, by comparing the automata-theoretic characterization of subsumption w.r.t. cyclic $\mathcal{ALN}$-terminologies [7] with the structural subsumption algorithm for cyclic terminologies realized in K-Rep [6].

The comparison between the structural and the automata-theoretic approach can also be extended to other inference tasks such as computing the least common subsumer (lcs) of $\mathcal{ALN}$-concept descriptions. Again, the algorithm for computing the lcs based on description graphs [5] can be seen as a special implementation of the automata theoretic characterization of the lcs [2]. An advantage of the automata-theoretic approach is that it easily carries over to computing the lcs for concepts defined by cyclic terminologies [2].

References

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

[2] F. Baader and R. Küsters. Least common subsumer computation w.r.t. cyclic $\mathcal{ALN}$-terminologies. In Proc. of DL'98, Trento, Italy, 1998.

[3] F. Baader, R. Küsters, and R. Molitor. Structural Subsumption Considered from an Automata Theoretic Point of View. LTCS-Report 98-04, LuFg Theoretical Computer Science, RWTH Aachen, Germany, 1998. Available at `http://www-lti.informatik.rwth-aachen.de/Forschung/Papers.html`.

[4] A. Borgida and P. Patel-Schneider. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *JAIR*, 1:277–308, 1994.

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

[6] R. Dionne, E. Mays, and F.J. Ohles. The equivalence of model-theoretic and structural subsumption in description logics. In Proc. of IJCAI'93, Chambery, France, 1993.

[7] R. Küsters. Characterizing the semantics of terminological cycles in $\mathcal{ALN}$ using finite automata. In Proc. of KR'98, Trento, Italy, 1998.

[8] R. Molitor. Structural Subsumption for $\mathcal{ALN}$. LTCS-Report 98-03, LuFg Theoretical Computer Science, RWTH Aachen, Germany, 1998.

[9] B. Nebel. Terminological reasoning is inherently intractable. *AIJ*, 43(2):235–249, 1990.