Precompiling $\mathcal{ALC}$ TBoxes and Query Answering

Ulrich Furbach and **Claudia Obermaier** [1]

Abstract. Knowledge compilation is a common technique for propositional logic knowledge bases. The idea is to transform a given knowledge base into a special normal form ([11],[6]), for which queries can be answered efficiently. This precompilation step is very expensive but it only has to be performed once. We propose to apply this technique to knowledge bases defined in Description Logics. For this, we introduce a structure called linkless graph, for $\mathcal{ALC}$ concepts. Further we present an algorithm, based on path dissolution, which can be used for this precompilation step. We discuss an efficient satisfiability test as well as a subsumption test for precompiled concept descriptions. Finally we show how to extend this approach in order to precompile Tboxes and to use the precompiled Tboxes for efficient Tbox reasoning.

1 Introduction

Knowledge compilation is a technique for dealing with computational intractability of propositional reasoning. It has been used in various AI systems for compiling knowledge bases offline into systems, that can be queried more efficiently after this precompilation. An overview about techniques for propositional knowledge bases is given in [7]; more recently [6] discusses, how knowledge compilation techniques can be seen as DPLL-procedures. One of the most prominent successful applications of knowledge compilation is certainly in the context of belief networks ([5]). In this context the precompilation step, although it is very expensive, pays off because it only has to be performed once to the network, which is not changing too frequently. In the context of Description Logics, knowledge compilation has firstly been investigated in [1], where $\mathcal{FL}$ concept descriptions are approximated by $\mathcal{FL}^-$concept descriptions.

In this paper we propose to apply a similar technique to knowledge bases defined in Description Logics. There are several techniques for Description Logics which are related to our approach. An overview on precompilation techniques for description logics such as structural subsumption, normalization and absorption is given in [8]. To perform a subsumption check on two concepts, structural subsumption algorithms ([2]) transform both concepts into a normal form and compare the structure of these normal forms. However these algorithms typically have problems with more expressive Description Logics. Especially general negation, which is an important feature in the application of Description Logics, is a problem for those algorithms. The technique of structural subsumption algorithms is used in CLASSIC [12], GRAIL [13] and LOOM [9]. In contrast to structural subsumption algorithms our approach is able to handle general negation without problems.

Normalization ([3]) is another preprocessing technique for Description Logics, which eliminates redundant operators in order to determine contradictory as well as tautological parts of a concept. In many cases this technique is able to simplify subsumption and satisfiability problems.

Absorption ([14]) is a technique which tries to eliminate general inclusion axioms from a knowledge base. Both absorption and normalization have the aim of increasing the performance of tableau based reasoning procedures. In contrast to that, our approach extends the use of preprocessing. We suggest to transform the concept into a normal form called linkless graph which allows an efficient consistency test. For this consistency test a tableau procedure is not necessary anymore. Some subsumption queries can also be solved without a tableau algorithm. We will discuss that in Section 4.

In this paper we will consider the Description Logic $\mathcal{ALC}$ [2] and we adopt the concept of linkless formulae, as it was introduced in [10, 11]. The following section shortly introduces the idea of our precompilation. In Section 2 we describe linkless concept descriptions and give a transformation of $\mathcal{ALC}$ concept descriptions into linkless ones. This transformation is extended to precompile $\mathcal{ALC}$ Tboxes in Section 3. Further in Section 4 we discuss an efficient consistency test for precompiled concept descriptions.

2 Precompilation of $\mathcal{ALC}$ Concept Descriptions

The precompilation technique we use for an $\mathcal{ALC}$ concept C consists of two steps. In the first step C is transformed into a normal form by removing so called *links* occurring in C. The notion of a link has first been introduced for propositional logic ([10]). Intuitively links are contradictory parts of a concepts which therefore can be removed preserving equivalence.

In the second step of the precompilation process we consider role restrictions. Given for example $C = \exists R.B \sqcap \forall R.D$. According to the semantic of $\mathcal{ALC}$ it follows from $x \in C^I$ that there is an individual y with $(x, y) \in R^I$ and $y \in (B \sqcap D)^I$. The concept $B \sqcap D$ is precompiled in the second step of the precompilation. The second step is repeated recursively until all concept descriptions of reachable individuals are precompiled.

In the following we assume that concept descriptions in $\mathcal{ALC}$ are given in NNF, i.e., negation occurs only in front of concept names. Further the term *concept literal* denotes either a concept name or a negated concept name.

Definition 1 *For a given concept C, the set of its paths is defined as follows:*

$$\begin{aligned}
&\textit{paths}(\bot) = \emptyset\\
&\textit{paths}(\top) = \{\emptyset\}\\
&\textit{paths}(C) = \{\{C\}\}, \textit{ if C is a literal}\\
&\textit{paths}(C_1 \sqcap C_2) = \{X \cup Y | X \in \textit{paths}(C_1) \; and \; Y \in \textit{paths}(C_2)\}\\
&\textit{paths}(C_1 \sqcup C_2) = \textit{paths}(C_1) \cup \textit{paths}(C_2)
\end{aligned}$$

[1] University of Koblenz-Landau Germany, email: obermaie@uni-koblenz.de

The concept description $C = \neg A \sqcap (A \sqcup B) \sqcap \forall R.(E \sqcap F)$ has the two paths $p_1 = \{\neg A, A, \forall R.(E \sqcap F)\}$ and $p_2 = \{\neg A, B, \forall R.(E \sqcap F)\}$. We typically use p to refer to both the path and the conjunction of the elements of the path when the meaning is evident from the context.

In propositional logic a link means that the formula has a contradictory part. Furthermore if all paths of a formula contain a link, the formula is unsatisfiable. In Description Logics other concepts apart from complementary concept literals are able to form a contradiction. It is possible to construct an inconsistent concept description by using role restrictions. For example the concept $\exists R.C \sqcap \forall R.\neg C$ is inconsistent since it a) claims that there has to be an individual which is reachable via the role R and belongs to the concept C and b) claims that all individuals which are reachable via the role R have to belong to the concept $\neg C$. This clearly is not possible. We could say that the concept contains a link in the description of a reachable individual. Therefore in Description Logics it is not sufficient to consider links constructed by concept literals. We will take a closer look at role restrictions in Section 2.2.

Definition 2 *For a given concept C a* link *is a set of two complementary concept literals occurring in a path of C. The positive (negative) part of a link denotes its positive (negative) concept literal.*

Note that we regard $\bot$ and $\top$ as a complementary pair of concept literals. Obviously a path p is inconsistent, iff it contains a link or alternatively $\{\exists R.A, \forall R.B_1, \ldots, \forall R.B_n\} \subseteq p$ and all paths in $A \sqcap B_1 \sqcap \ldots \sqcap B_n$ are inconsistent. Note that a set of consistent paths uniquely determines a class of semantically equivalent concept descriptions. Further given a concept description C with a consistent path p, it is obvious that the interpretation of p is a model of C. And the other way around each model I of C is also a model of a consistent path of C.

Now we are able to define the term linkless.

Definition 3 *A concept C is called* linkless, *if C is in NNF and there is no path in C which contains a link.*

This special structure of linkless concepts allows us to consider each conjunct of a conjunction separately. Therefore satisfiability can be decided in linear time and it is possible to enumerate models very efficiently.

Note that a linkless concept description can still be inconsistent. Take $\forall R.B \sqcap \exists R.\neg B$ as an example. This example makes clear that it is not sufficient to remove links from a concept description. We also have to consider role restrictions. But first we learn how to remove links from a given concept description.

2.1 Removing Links

In this section a method to transform an $\mathcal{ALC}$ concept into an equivalent linkless $\mathcal{ALC}$ concept is introduced. In propositional logic one possibility to remove links from a formula is to use path dissolution ([10]). The idea of this algorithm is to eliminate paths containing a link. This technique will be used in our context as well.

Definition 4 *Let G be a concept description and A be a concept literal. The* path extension *of A in G, denoted by $CPE(A, G)$, is a concept G' containing exactly those paths in G which contain A. The* path complement *of A in G, denoted by $CPC(A, G)$, is the concept G' containing exactly those paths in G which do not contain A.*

Note that Definition 4 does not mention how to construct $CPE(A, G)$ and $CPC(A, G)$. The naive way would be to construct the disjunction of all respective paths in G. However there are more elaborate methods ([10]), producing a far more compact results.

Lemma 5 *For a concept G and a set of literals A, where all elements of A occur in G, the following holds:*
$G \equiv CPE(A, G) \sqcup CPC(A, G)$

We want to construct $CPE(D, G)$ and $CPC(D, G)$ for $G = (D \sqcup \forall R.E) \sqcap (C \sqcup \forall R.B)$. G has the paths: $c_1 = \{D, C\}$, $c_2 = \{D, \forall R.B\}$, $c_3 = \{\forall R.E, C\}$ and $c_4 = \{\forall R.E, \forall R.B\}$. This leads to $CPE(D, G) = D \sqcap (C \sqcup \forall R.B)$ and $CPC(D, G) = \forall R.E \sqcap (C \sqcup \forall R.B)$.

In the following $\overline{C}$ denotes the complement of a concept C, which is given in NNF and can be calculated simply by transforming $\neg C$ in NNF. Our next aim is to remove a link from a concept description. Therefore we define a dissolution step for a link $\{L, \overline{L}\}$ through a concept expression $G = G_1 \sqcap G_2$ (such that $\{L, \overline{L}\}$ is neither a link for G_1 nor G_2). Note that each path p through $G_1 \sqcap G_2$ can be split into the paths p_1 and p_2, where p_1 is a path through G_1 and p_2 is a path through G_2.

Definition 6 *Given a concept description $G = G_1 \sqcap G_2$ which contains the link $\{L, \overline{L}\}$. Further $\{L, \overline{L}\}$ is neither a link for G_1 nor G_2. W.l.o.g. L occurs in G_1 and $\overline{L}$ occurs in G_2. The dissolvent of G and $\{L, \overline{L}\}$ denoted by $Diss(\{L, \overline{L}\}, G)$, is*

$$\begin{aligned} Diss(\{L, \overline{L}\}, G) = &(CPE(L, G_1) \sqcap CPC(\overline{L}, G_2)) \sqcup \\ &(CPC(L, G_1) \sqcap CPC(\overline{L}, G_2)) \sqcup \\ &(CPC(L, G_1) \sqcap CPE(\overline{L}, G_2)) \end{aligned}$$

Note that $Diss(\{L, \overline{L}\}, G)$ removes exactly those paths from G which contain the link $\{L, \overline{L}\}$. Since these paths are inconsistent, $Diss(\{L, \overline{L}\}, G)$ is equivalent to G. This is stated in the next lemma where we use the standard set-theoretic semantics for $\mathcal{ALC}$. The interpretation of a concept C denoted by C^I is a subset of the domain and can be understood as the set of individuals belonging to the concept C in the interpretation I.

Lemma 7 *Let G be a concept description and $\{L, \overline{L}\}$ be a link in G such that $Diss(\{L, \overline{L}\}, G)$ is defined. Then for all x in the domain holds: $x \in G^{\mathcal{I}}$ iff $x \in Diss(\{L, \overline{L}\}, G)^{\mathcal{I}}$.*

By equivalence transformations and with the help of Lemma 5 the following lemma follows.

Proposition 8 *Let $\{L, \overline{L}\}$ and G be defined as in Definition 6. Then the following holds:*

$$\begin{aligned} Diss(\{L, \overline{L}\}, G) \equiv &(G_1 \sqcap CPC(\overline{L}, G_2)) \sqcup \\ &(CPC(L, G_1) \sqcap CPE(\overline{L}, G_2)) \\ Diss(\{L, \overline{L}\}, G) \equiv &(CPE(L, G_1) \sqcap CPC(\overline{L}, G_2)) \sqcup \\ &(CPC(L, G_1) \sqcap G_2) \end{aligned}$$

Now it is easy to see how to remove links: Suppose a concept description C in NNF is given and it contains a link $\{L, \overline{L}\}$. Then there must be conjunctively combined subconcepts G_1 and G_2 of C where the positive part L of the link occurs in G_1 and the negative part $\overline{L}$ occurs in G_2. In the first step we construct $CPE(L, G_1), CPC(L, G_1), CPE(\overline{L}, G_2)$ as well as

$CPC(\overline{L}, G_2)$. By replacing $G_1 \sqcap G_2$ in C by $Diss(\{L, \overline{L}\}, G_1 \sqcap G_2)$ we are able to remove the link.

Next we give an algorithm to remove all links in the way it is described above. In the following definition $G[G_1/G_2]$ denotes the concept one obtains by substituting all occurrences of G_1 in G by G_2.

Algorithm 9 *Let G be a concept description.*

$linkless(G) \overset{def}{=} G$, *if G is linkless.*

$linkless(G) \overset{def}{=} linkless(G[H/Diss(\{L, \overline{L}\}, H)])$, *where H is a subconcept of G and $\{L, \overline{L}\}$ is a link in H, such that $Diss(\{L, \overline{L}\}, H)$ is defined.*

Theorem 10 *Let G be a concept description. Then $linkless(G)$ is equivalent to G and is linkless.*

Note that in the worst case this transformation leads to an exponential blowup of the concept description.

2.2 Handling Role Restrictions

In the previous section we learned how to remove all links form a given concept. Now we turn to the second step of the precompilation and consider role restrictions.

Definition 11 *Let C be a linkless concept, p be a path in C with $\exists R.A \in p$ and $A, B_1, \ldots, B_n$ be concepts. Further let $\{\forall R.B_1, \ldots, \forall R.B_n\} \subseteq p$ be the (possibly empty) set of all universal role restrictions w.r.t. R in p. Then the concept $C' \equiv A \sqcap B_1 \sqcap \ldots \sqcap B_n$ is called* R-reachable *from C. Further the concept $C'' \equiv B_1 \sqcap \ldots \sqcap B_n$ is called* potentially R-reachable *from C. p is called a path used to reach C' (potentially reach C") from C.*

Note that it is possible that a concept description C' is (potentially) reachable from a concept description C via several paths. A concept description C' is called (potentially) reachable from a linkless concept description C, if it is (potentially) R-reachable from C for some role R. Further *universally (potentially) reachable* is the transitive reflexive closure of the relation *(potentially) reachable*. Given a concept C and a concept C' which is reachable from C. Since the concept C' is equivalent to the concept $linkless(C')$, we call both C' and $linkless(C')$ reachable from C.

For example the following linkless concept description:
$C = (\exists R.(D \sqcup E) \sqcup A) \sqcap \forall R.\neg D \sqcap \forall R.E \sqcap B$ which has the two different paths $p_1 = \{\exists R.(D \sqcup E), \forall R.\neg D, \forall R.E, B\}$ and $p_2 = \{A, \forall R.\neg D, \forall R.E, B\}$. The concept $C' \equiv (D \sqcup E) \sqcap \neg D \sqcap E$ is reachable from C via path p_1 using $\{\exists R.(D \sqcup E), \forall R.\neg D, \forall R.E\}$. However C' is not linkless.

In the second step of the precompilation we precompile, i.e remove all links from, all universally (potentially) reachable concepts. Further it is necessary to precompile all potentially universally reachable concepts as soon as we want to answer queries. For example the concept $\forall R.\neg D \sqcap \forall R.\neg E$ does not have any reachable concept descriptions since no existential role restriction w.r.t. the role R is present. However asking a query to this concept can introduce the missing existential role restriction and can make a concept description reachable. For example asking the subsumption query $\forall R.\neg D \sqcap \forall R.\neg E \sqsubseteq \forall R.(\neg D \sqcap \neg E)$ leads to checking the consistency of $\forall R.\neg D \sqcap \forall R.\neg E \sqcap \exists R.(D \sqcup E)$. So the transformation of the subsumption query to a consistency test introduced the missing existential role restriction and therefore makes a concept description reachable. Therefore those concepts have to be precompiled as well.

Since the concepts which are (potentially) reachable from another concept via a path p only depends on the role restrictions in occurring in p, we regard all paths containing the same role restrictions as equivalent. The result of the precompilation of a concept C can be represented by a rooted directed graph (N, E) i.e a directed graph with exactly one source. The graph consists of two different types of nodes: path nodes PN and concept nodes CN. So $N = CN \cup PN$. Whereas each path node in PN is a set of paths in C and each node in the CN is a linkless concept description. The set of edges is $E \subset (CN \times PN) \cup (PN \times CN)$. A concept node C_i has a successor node for each set of equivalent paths in C_i and further there is an edge from each path node to the concept nodes of (potentially) reachable concepts. These edges are labeled by a set of universally quantified role restrictions or by a set containing universally quantified role restrictions and one existential role restriction. This label indicates the role restrictions used to (potentially) reach a concept.

Definition 12 *The linkless graph of a concept C is defined as follows:*

- *If C does not contain any role restrictions, the precompilation of C is a rooted directed graph consisting of the one node $linkless(C)$ with one successor which is the set of paths of C.*
- *If C contains role restrictions, the precompilation of C is a rooted directed graph with root $linkless(C)$ and for each set P_i of equivalent paths in C there is a subsequent path node. There is an edge form a path node P_i to the linkless graph of concept node C', if C' is (potentially) reachable from C via one of the paths in P_i. This edge is labeled by the set of role restrictions used to reach C' from C.*

Since the depth of the linkless graph of a given concept C corresponds to the depth of nested role restrictions in C, the linkless graph is always finite. Further in case of the precompilation of a single concept description, the linkless graph is a rooted dag.

Consider for example the following concept with its four paths:

$$C \equiv (B \sqcap \neg E) \sqcup ((B \sqcup \neg A \sqcup (\exists R.A \sqcap A)) \sqcap \exists R.E \sqcap \forall R.F)$$

$p_1 = \{B, \neg E\}$ $\quad$ $p_2 = \{B, \exists R.E, \forall R.F\}$

$p_3 = \{\neg A, \exists R.E, \forall R.F\}$ $\quad$ $p_4 = \{\exists R.A, A, \exists R.E, \forall R.F\}$

There are three sets of equivalent paths: $\{p_1\}$, $\{p_2, p_3\}$ and $\{p_4\}$. The root of the linkless graph is C. For each set of equivalent paths, there is a successor path node. In the next step, reachable concepts are considered: for instance the concept $E \sqcap F$ is reachable via the paths in the second set of paths using the role restrictions $\{\exists R.E, \forall R.F\}$. Therefore there is an edge from the second path node to the concept node $E \sqcap F$ with $label(\langle\{p_2, p_3\}, E \sqcap F\rangle) = \{\exists R.E, \forall R.F\}$. In the same way, the precompilation of all (potentially) reachable concepts are combined with the path nodes. The result is the graph depicted in Fig. 1.

3 Precompilation of General Tboxes

When answering queries with respect to a general Tbox it is necessary to restrict reasoning such that only models of this Tbox are considered. As described in [2] we transform the given Tbox $\mathcal{T} = \{C_1 \sqsubseteq D_1, \ldots, C_n \sqsubseteq D_n\}$ into a meta constraint $\mathcal{M}$ with

$$\mathcal{M} = (\neg C_1 \sqcup D_1) \sqcap \ldots \sqcap (\neg C_n \sqcup D_n)$$

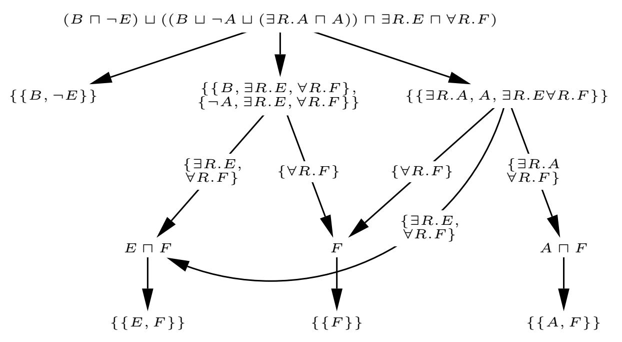
$(B \sqcap \neg E) \sqcup ((B \sqcup \neg A \sqcup (\exists R.A \sqcap A)) \sqcap \exists R.E \sqcap \forall R.F)$
$\{\{B, \neg E\}\}$
$\{\{B, \exists R.E, \forall R.F\}, \{\neg A, \exists R.E, \forall R.F\}\}$
$\{\{\exists R.A, A, \exists R.E \forall R.F\}\}$
$\{\exists R.E, \forall R.F\}$
$\{\forall R.F\}$
$\{\forall R.F\}$
$\{\exists R.A \forall R.F\}$
$\{\exists R.E, \forall R.F\}$
$E \sqcap F$
F
$A \sqcap F$
$\{\{E, F\}\}$
$\{\{F\}\}$
$\{\{A, F\}\}$

Figure 1. Example for a linkless graph

The idea of the linkless graph can be directly extended to represent precompiled Tboxes. We just construct the linkless graph for $\mathcal{M}$. Further, instead of just considering the concept nodes, each concept node C must also fulfill $\mathcal{M}$. So whenever there is a (potentially) reachable concept C', we precompile $C' \sqcap \mathcal{M}$ instead of just C'. In the case of precompiling a single concept description, the result is a linkless dag. In contrast to that the precompilation of a Tbox in general contains cycles and therefore leads to a linkless graph.

4 Properties of Precompiled Concept / Tboxes

Now we will consider some properties of a linkless graph in order to show that it is worthwhile to precompile a given concept description or a given Tbox into a linkless graph. We start by giving an efficient consistency check.

4.1 Consistency

Theorem 13 *Let C be a concept description and (N, E) its linkless graph with the root node H. Then holds: C is inconsistent iff $H = \bot$ or for each P with $\langle H, P\rangle \in E$ there is a concept C' which is reachable from C via one of the paths in P and the subgraph with root $linkless(C')$ is inconsistent.*

In the following we also use the term *inconsistent* for a linkless graph of an inconsistent concept. By adding a label sat to each concept node in the linkless graph, it can be ensured that no subgraph has to be checked more then once. At the beginning the sat label is set to the value $unknown$. Whenever during the consistency check a subgraph with root node C' is found to be (consistent) inconsistent, we set its sat label to ($true$) $false$. Only if it has the value $unknown$ it is necessary to perform a consistency check for this subgraph. Note that the use of the sat label does not only increase the efficiency of the consistency check. It furthermore prevents getting caught in cycles of the graph.

The consistency check described in Theorem 13 can be used to check the consistency of a precompiled Tbox as well. However it is important to use the sat label mentioned above, in order to ensure termination.

So to show that a precompiled concept is inconsistent, we have to compare all universally reachable concept description to $\bot$. Each of these checks can be done in constant time. Therefore the whole consistency check takes time linear to the number of universally reachable concepts.

As mentioned above, when precompiling a Tbox, the respective metaconstraint has to be added to every universally reachable concept. In the worst case there can be exponentially many universally reachable concepts. Given r different roles each with n existential role restrictions, m universal role restrictions which are all nested with depth d, in the worst case the number of universally reachable concepts is $r \cdot m \cdot 2^n \cdot d$. However in real world ontologies the number of universally reachable worlds is smaller. Furthermore precompiling a Tbox never increases the number of reachable concepts, contrariwise it usually decreases the number of reachable concepts. For example for the amino-acid [1] ontology $r = 5$, $d = 1$, $m = 3$ and $n = 5$. So in the worst case, there are 480 universally reachable worlds. But in reality, before the precompilation there are 170 and after the precompilation 154 reachable concepts.

4.2 Using the Linkless Graph to Answer Queries

Given the precompilation of a concept description, it is possible to answer certain subsumption queries very efficiently. In [4] an operator called conditioning is used as a technique to answer queries for a precompiled knowledge base. The idea of the conditioning operator is to consider $C \sqcap \alpha$ for a concept literal α and to simplify C according do α. Given for example $C = (B \sqcup E) \sqcap D$ and $\alpha = \neg B$, $C \sqcap \alpha$ can be simplified to $E \sqcap D$.

Definition 14 *Let C be a linkless concept description and $\alpha = C_1 \sqcap \ldots \sqcap C_n$ with C_i a concept literal. Then C conditioned by α, denoted by $C|\alpha$, is the concept description obtained by replacing each occurrence of C_i in C by $\top$ and each occurrence of $\overline{C_i}$ by $\bot$ and simplifying the conjunction according to the following simplifications:*

$\top \sqcap C = C$	$\top \sqcup C = \top$	$\bot \sqcap C = \bot$
$\bot \sqcup C = C$	$\exists R.\bot = \bot$	$\forall R.\top = \top$

It is clear that the conditioning operation is linear in the size of the concept description C. From the way $C|\alpha$ is constructed, it follows that $C|\alpha \sqcap \alpha$ is equivalent to $C \sqcap \alpha$ and obviously $C|\alpha \sqcap \alpha$ is linkless.

Definition 15 *Each concept literal is a* conditioning literal. *For each conditioning literal B, $\exists R.B$ and $\forall R.B$ are conditioning literals.*

Given a concept and a set of conditioning literals, in order to use conditioning for precompiled concepts we have to know how conditioning changes the set of paths in a concept description.

Definition 16 *Let P be a set of paths and α a set of concept literals. Then $\hat{P}$ denotes the set of paths obtained form P by*

1. *removing all elements of α from paths in P,*
2. *removing all paths from P, which contain an element whose complement is in α and*
3. *removing all paths p_1 from the remaining paths, if $p_2 \subset p_1$ for some $p_2 \in P$.*

Proposition 17 *Let C be a linkless concept description, P be the set of all paths in C and α a set of conditioning literals. Then the set of minimal paths of $C|\alpha$ is equal to $\hat{P}$.*

Next we want to use conditioning on linkless graphs. We give an algorithm for the conditioning operator for an arbitrary node of a linkless graph.

[1] http://www.co-ode.org/ontologies/amino-acid/2006/05/18/amino-acid.owl

Algorithm 18 *Let (N, E) be a linkless graph, $C \in N$ be a concept node and α a conditioning literal. Then C conditioned by α w.r.t. (N, E) denoted by $C|_{node}\alpha$ is the linkless graph obtained from (N, E) as follows:*

- *If α is a concept literal, then substitute $C|\alpha \sqcap \alpha$ for C. Further for each P with $\langle C, P\rangle \in E$ substitute $\hat{P}$ for P and add α to all paths in $\hat{P}$. If $\hat{P} = \emptyset$, remove its node and all its in- and outgoing edges.*
- *If $\alpha = QR.B$ with $Q \in \{\exists, \forall\}$ and B a conditioning literal, then substitute $C \sqcap QR.B$ for C and for each P with $\langle C, P\rangle \in E$ add $QR.B$ to all paths in P. Further*
  - *For all P whose paths do not contain a role restriction w.r.t. R: Create the linkless graph for B [1], add an edge from P to its root and label the edge with $\{QR.B\}$.*
  - *For all P whose paths contain role restrictions w.r.t. R:*
    - *For all P whose paths do not contain a universal role restriction w.r.t. R: Create the linkless graph for B, add an edge from P to its root and label the edge with $\{QR.B\}$.*
    - *If $Q = \forall$: add $QR.B$ to the labels of all edges $\langle P, C'\rangle \in E$ whose label contains a role restriction w.r.t. R and calculate $C'|_{node}B$.*
    - *If $Q = \exists$: for all edges $\langle P, C'\rangle \in E$ whose label contains only universal role restrictions w.r.t. R, copy the subgraph w.r.t. C' producing a new subgraph w.r.t. node C''. Create an edge from P to C'', label it with $label(\langle P, C'\rangle) \cup \{\exists R.B\}$ and calculate $C''|_{node}B$.*

During the calculation of $C|_{node}\alpha$ the depth of nested role restrictions in α decreases, hence the calculation always terminates. In fact, if the role restrictions are nested with maximal depth d, the conditioning only affects concept nodes which are reachable with d steps from the root node. Further the conditioning changes path nodes an labels of edges. So the complexity of the conditioning operator is linear to the number of concepts which are (potentially) universally reachable from the root node with d steps.

The conditioning operator for nodes can easily be extended to handle sets of conditioning literals.

Lemma 19 *Let C be concept description, H the root node of its linkless graph and α a set of conditioning literals. Then $C|\alpha$ is consistent iff $H|_{node}\alpha$ is consistent.*

Theorem 20 *Given a concept C, its linkless graph with root H and a subsumption query $C \sqsubseteq D$. If D is a concept literal, then $C \sqsubseteq D$ holds, iff $H|_{node}\neg D$ is inconsistent.*

Theorem 20 follows directly from Lemma 19, since $C \sqsubseteq D$ is equivalent to $C \sqcap \neg D$. We can use conditioning to combine C and $\neg D$. According to Lemma 19 $C|\neg D$ is consistent iff $H|_{node}\neg D$ is consistent. So we can construct $H|_{node}\neg D$ and test its consistency. If $H|_{node}\neg D$ is inconsistent, the subsumption query $C \sqsubseteq D$ holds. Theorem 20 can be easily extended for subsumption queries with a concept D, which is a in NNF and is constructed only using the connectives disjunction and negation.

Let's now consider the concept C whose linkless graph is presented in Fig. 1. If we want to answer the query $C \sqsubseteq B \sqcup \exists R.E \sqcup \exists R.A$ we have to condition the root of the linkless graph with $\{\neg B, \forall R.\neg E, \forall R.\neg A\}$. With the help of the consistency check mentioned above, we find out that the resulting graph is inconsistent. Therefore the subsumption query holds.

The linkless graph of a given Tbox $\mathcal{T}$ can be easily used to do Tbox reasoning. Let A and B be concepts both given in NNF and further A is constructed only using the connectives conjunction and negation and B is constructed only using the connectives disjunction and negation. If we want to check whether a subsumption $A \sqsubseteq_{\mathcal{T}} B$ holds, we have to check the consistency of $\mathcal{T} \sqcap A \sqcap \neg B$. Assuming that we have the linkless graph of $\mathcal{T}$, we only have to condition the root of the graph with the set of conjuncts in $A \sqcap \neg B$. We perform a consistency check for the resulting graph and if the graph is inconsistent, the subsumption holds.

Since in Tbox reasoning many queries are asked to the same Tbox, it is worthwhile to precompile the Tbox into a linkless graph. After that precompilation step, we can answer subsumption queries with the above mentioned structure very efficiently.

5 Future Work / Conclusion

In the next step, we want to investigate how to extend our approach to more expressive Description Logics for example $\mathcal{SHOIN}$, which is very important in the context of semantic web. Further it would be interesting to consider the satisfiability of concept descriptions which are almost linkless. In this context almost linkless means that the concept description is linkless outside of a certain scope.

Projection is a very helpful technique when different TBoxes have to be combined. Therefore we will investigate how to project linkless concept descriptions on a set of literals. Since linkless concept descriptions are closely related to a normal form which allows efficient projection, it is very likely that our normal form has this property too.

[1] Due to the structure of B, its linkless graph has a linear structure an can be constructed in time linear to the depth of nested role restrictions in B.

REFERENCES

[1] B. Selman and H. Kautz, 'Knowledge Compilation and Theory Approximation', *J. ACM*, **43**(2), 193–224, (1996).

[2] F. Baader et al., eds. *The Description Logic Handbook*. Cambridge University Press, 2003.

[3] P. Balsiger and A. Heuerding, 'Comparison of Theorem Provers for Modal Logics - Introduction and Summary.', in *TABLEAUX*, volume 1397 of *LNCS*, pp. 25–26. Springer, (1998).

[4] A. Darwiche, 'Decomposable Negation Normal Form', *Journal of the ACM*, **48**(4), (2001).

[5] A. Darwiche, 'A Logical Approach to Factoring Belief Networks', in *Proceedings of KR*, pp. 409–420, (2002).

[6] A. Darwiche and J. Huang, 'DPLL with a Trace: From SAT to Knowledge Compilation', in *Proceedings of IJCAI 05*, (2005).

[7] A. Darwiche and P. Marquis, 'A Knowlege Compilation Map', *Journal of Artificial Intelligence Research*, **17**, 229–264, (2002).

[8] I. Horrocks, 'Implementation and Optimization Techniques.', In Baader et al. [2], pp. 306–346.

[9] Robert M. MacGregor, 'Inside the LOOM Description Classifier.', *SIGART Bulletin*, **2**(3), 88–92, (1991).

[10] N. Murray and E. Rosenthal, 'Dissolution: Making Paths Vanish', *J. ACM*, **40**(3), 504–535, (1993).

[11] N. Murray and E. Rosenthal, 'Tableaux, Path Dissolution, and Decomposable Negation Normal Form for Knowledge Compilation', in *Proceedings of TABLEAUX 2003*, volume 1397 of *LNCS*. Springer, (2003).

[12] P. Patel-Schneider, D. McGuinness, and A. Borgida, 'The CLASSIC Knowledge Representation System: Guiding Principles and Implementation Rationale.', *SIGART Bulletin*, **2**(3), 108–113, (1991).

[13] A. Rector, S. Bechhofer, C. Goble, I. Horrocks, W. A. Nowlan, and W. D. Solomon, 'The GRAIL concept modelling language for medical terminology.', *Artificial Intelligence in Medicine*, **9**(2), 139–171, (1997).

[14] D. Tsarkov and I. Horrocks, 'Description Logic Reasoner: System Description.', in *IJCAR*, eds., U. Furbach and N. Shankar, volume 4130 of *Lecture Notes in Computer Science*, pp. 292–297. Springer, (2006).