Knowledge Compilation for Description Logics

Ulrich Furbach and Claudia Obermaier

Universität Koblenz-Landau
D56070 Koblenz, Germany
{uli,obermaie}@uni-koblenz.de

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 ([MR03],[DH05]), 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 normal form, called linkless concept descriptions, for $\mathcal{ALC}$ concepts. Further we present an algorithm, based on path dissolution, which can be used to transform a given concept description into an equivalent linkless concept description. Finally we discuss a linear satisfiability test as well as a subsumption test for linkless concept descriptions.

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 off-line into systems, that can be queried more efficiently after this pre-compilation. An overview about techniques for propositional knowledge bases is given in [DM02]; more recently [DH05] and [Wer07] discuss, 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 ([Dar02]). In this context the pre-compilation 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 was firstly been investigated in [Bar96], 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. The situation is very similar: there is a given ontology, which does not change frequently. In a typical scenario we ask many queries to the same ontology. Therefore it makes sense to use the idea of knowledge compilation for Description Logics as well.

Unfortunately it is not possible to use propositional logic as a target language, like in the case of belief networks, because Description Logics are more expressive and hence we would have to use a decidable fragment of predicate logic. Instead we propose to transform a given ontology directly into a normal form, which allows for an efficient reasoning.

Projection is a technique which is helpful for combining different TBoxes, that is why one of our future interests is to use our approach in the context of combining different TBoxes. Depending on the normal form, precompiled knowledge bases can be projected on a set of literals very efficiently. One example can be found in [Dar01]. Since the normal form we suggest is closely related to a normal form which allows efficient projection, it is very likely that our normal form has this poperty as well.

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 [Hor03]. To perform a subsumption check on two concepts, structural subsumption algorithms ([BN03]) 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 [PSMB91], GRAIL [RBG$^+$97] and LOOM [Mac91]. In contrast to structural subsumption algorithms our approach is able to handle general negation without problems.

Normalization is another preprocessing technique for Description Logics. Description Logics often have redundant operators to form concepts. In $\mathcal{ALC}$ for example it is possible to build concepts without using the $\sqcup$ operator since $C \sqcup D \equiv \neg(\neg C \sqcap \neg D)$. Normalization eliminates those 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. [BH98] provides more information about the use of this technique for modal logics.

Absorption is a technique which tries to eliminate general inclusion axioms from a knowledge base. It is a known fact that general inclusion axioms in a TBox have the effect of worsening the performance of tableau based satisfiability and subsumption checking procedures since they introduce a high degree of non-determinism ([Hor98]). This is why absorption is a widely used technique ([TH06]). However the technique of absorption can not be applied for all general inclusion axioms.

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 descriptions into a normal form which allows a linear satisfiability check. For this satisfiability check 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 simple Description Logic $\mathcal{ALC}$ [BN03] and we adopt the concept of linkless formulae, as it was introduced in [MR93,MR03].

The following section describes the concept of linkless DL-formulae and in Section 3 we describe the transformation of $\mathcal{ALC}$ concept descriptions into linkless ones and in Section 4 we discuss a linear satisfiability test for linkless concept descriptions. On top of that we introduce a linear subsumption check for some concept descriptions.

2 Linkless Concepts

In propositional logic a linkless formula is a formula F, which is in Negation Normal Form (NNF) and for every conjunction $\alpha = \alpha_1 \wedge \ldots \wedge \alpha_n$ in F holds: for all i, j with $1 \leq i, j \leq n$ and $i \neq j$ there is no a with $a \in literals(\alpha_i)$ and $\neg a \in literals(\alpha_j)$. This special structure of linkless formulae 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. Linkless formulae are very similar to formulae in Decomposable Negation Normal Form (DNNF). Actually DNNF is a special case of linkless formulae. For most purposes it is sufficient to work with linkless formulae. Since it is easier to transform a formula into a linkless formula than into DNNF we will focus on linkless formulae.

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 an atomic concept or a negated atomic concept. By the term *role literal* we describe a concept description of the form $\forall R.E$ or $\exists R.E$ with E a concept in NNF. Further by *literal* we mean either a concept literal or a role literal.

Definition 1. *(**c-path**) For a given concept C, the set of its c-paths is defined as follows:*

$$\textit{c-paths}(C) = \{\{C\}\}, \textit{ if } C \textit{ is a literal}$$
$$\textit{c-paths}(C_1 \sqcap C_2) = \{X \cup Y | X \in \textit{c-paths}(C_1) \textit{ and } Y \in \textit{c-paths}(C_2)\}$$
$$\textit{c-paths}(C_1 \sqcup C_2) = \textit{c-paths}(C_1) \cup \textit{c-paths}(C_2)$$

Example 1. The concept description $C = \neg A \sqcap (A \sqcup B) \sqcap \forall R.(E \sqcap F)$ has the two c-paths $cp_1 = \{\neg A, A, \forall R.(E \sqcap F)\}$ and $cp_2 = \{\neg A, B, \forall R.(E \sqcap F)\}$.

In propositional logic a link means that the formula has a contradictory part. Furthermore if all c-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 link. It is also possible to construct an inconsistent concept description by using role restrictions. For example the concept description $\exists R.C \sqcap \forall R.\neg C$ 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. On the other hand the concept $\forall R.C \sqcap \forall R.\neg C$ is consistent because an individual which is reachable via the role R is not required. Therefore in order to construct a link using role restriction we need one existential role restriction and at least one universal role restriction. Now we are able to translate the term of a link from propositional logic to Description Logic.

Definition 2. ***(Link)*** *For a given concept C a link is either a **concept link** or a **role link**.*

- *A **concept link** is a set of two complementary concept literals occuring in a c-path of C.*
- *A **role link** is a set $\{\exists R.D, \forall R.E_1, \ldots, \forall R.E_n\}$ of literals occuring in a c-path of C and where all c-paths in $D \sqcap E_1 \sqcap \ldots \sqcap E_n$ contain $\bot$ or a concept link or a role link and no subset of $\{\exists R.D, \forall R.E_1, \ldots, \forall R.E_n\}$ is a role link.*

The positive (negative) part of a concept link denotes its positive (negative) concept literal. Further the positive (negative) part of a role link denotes the existentially (universally) quantified elements of the role link.

Note that we regard $\bot$ and $\top$ as a complementary pair of concept literals. This definition is closely related to the definition of a link in a propositional logic formula. In example 1 above the path cp_1 contains the concept

link $\{A, \neg A\}$. Obviously the conjunction of the elements of a concept link is inconsistent.

Example 2. Consider the following concept description:

$$C = A \sqcap (\neg A \sqcup \forall R.(\exists R.(\neg D \sqcap \neg E)) \sqcap \exists R.(\forall R.(D \sqcup E))$$

C has the following c-paths:

$$cp_1 = \{A, \neg A, \exists R.(\forall R.(D \sqcup E))\}$$
$$cp_2 = \{A, \forall R.(\exists R.(\neg D \sqcap \neg E)), \exists R.(\forall R.(D \sqcup E))\}$$

In order to find out if there is a role link in cp_2 we have to consider the concept $C' = \exists R.(\neg D \sqcap \neg E) \sqcap \forall R.(D \sqcup E)$. C' only has the c-path $cp' = \{\exists R.(\neg D \sqcap \neg E), \forall R.(D \sqcup E)\}$. To find out if cp' contains a role link, we consider all c-paths in $C'' = \neg D \sqcap \neg E \sqcap (D \sqcup E)$. C'' contains the two c-paths $cp''_1 = \{\neg D, \neg E, D\}$ and $cp''_2 = \{\neg D, \neg E, E\}$. Both of these c-paths contain a concept link and therefore we conclude that our original concept C contains a role link.

Definition 3. ***(Inconsistent c-path)*** *A c-path is called* ***inconsistent****, if it contains a link. Otherwise it is called* ***consistent****.*

Note that a set of consistent c-paths uniquely determines a class of semantically equivalent concept descriptions. Now we are able to define the term linkless.

Definition 4. ***(Linkless Concept Description)*** *We call a concept description C linkless, if C is in NNF and there is no c-path in C which contains a link and if for each occurrence of $QR.E$ in C with $Q \in \{\exists, \forall\}$ the concept E is linkless as well.*

3 Transformation

In this section we introduce a method to transform an $\mathcal{ALC}$ concept into an equivalent linkless $\mathcal{ALC}$ concept. In propositional logic one possibility to remove links from a formula is to use path dissolution ([MR93]). The idea of this algorithm is to eliminate paths containing a link. This technique will be used in our context as well.

Definition 5. *Let G be a concept description and A be a set of literals where each element of A occurs in G.*

- *The c-path extension of A in G, denoted by $CPE(A, G)$, is a concept G' containing exactly those c-paths in G which contain A.*
- *The c-path complement of A in G, denoted by $CPC(A, G)$, is the concept G' containing exactly those c-paths in G which do not contain A.*

Note that Definition 5 does not mention how to construct $CPE(A, G)$ and $CPC(A, G)$. One possibility would be to construct the disjunction of all respective c-paths in G. If A only contains one element $CPE(A, G)$ and $CPC(A, G)$ are parts of G.

Lemma 1. *For a concept G and a set of literals A, where all elements of A occur in G, the following holds:*

$$G \equiv CPE(A, G) \sqcup CPC(A, G)$$

Example 3. Let's consider the following concept description

$$G_1 = (D \sqcup \forall R.E) \sqcap (C \sqcup \forall R.B)$$

We want to construct $CPE(\{\forall R.E, \forall R.B\}, G_1)$ and $CPC(\{\forall R.E, \forall R.B\}, G_1)$. Therefore we determine the different c-paths in G_1:

$$c_1 = \{D, C\} \qquad c_3 = \{\forall R.E, C\}$$
$$c_2 = \{D, \forall R.B\} \qquad c_4 = \{\forall R.E, \forall R.B\}$$

Now we are able to construct

$$CPE(\{\forall R.E, \forall R.B\}, G_1) = \forall R.E \sqcap \forall R.B$$
$$CPC(\{\forall R.E, \forall R.B\}, G_1) = (D \sqcap C) \sqcup (D \sqcap \forall R.B) \sqcup (\forall R.E \sqcap C)$$

Our next aim is to remove a link from a concept description. Therefore we define a dissolution step for a link A through a concept expression $G = G_1 \sqcap G_2$ (such that A is neither a link for G_1 nor G_2). First, we note that each c-path p through $G_1 \sqcap G_2$ can be split into the paths p_1 and p_2, where p_1 is a c-path through G_1 and p_2 is a c-path through G_2.

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

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

Note that $Diss(A, G)$ removes exactly those c-paths from G which contain the link A. Since these c-paths are inconsistent, $Diss(A, G)$ is equivalent to G. Since conjunction is commutative, it is no constriction to claim that the possitive part of the link occurs in G_1.

In the following lemma 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 2. *Let G be a concept description and A be a link in G such that $Diss(A, G)$ is defined. Then for all x in the domain holds: $x \in G^{\mathcal{I}}$ iff $x \in Diss(A, G)^{\mathcal{I}}$.*

Proof. First we divide the link A into its positive part L and its negative part $\overline{L}$. By precondition of $Diss(A, G)$, G must be of the form $G_1 \sqcap G_2$. W.l.o.g. L only occurs in G_1 and $\overline{L}$ only occurs in G_2. So $x \in G^I$ iff $x \in (G_1 \sqcap G_2)^{\mathcal{I}}$. By semantics of $\sqcap$ this is the case iff $x \in G_1^{\mathcal{I}}$ and $x \in G_2^{\mathcal{I}}$. Applying Lemma 1 to G_1 and G_2 leads to $x \in (CPE(L, G_1) \sqcup CPC(L, G_1))^{\mathcal{I}}$ and $x \in (CPE(\overline{L}, G_2) \sqcup CPC(\overline{L}, G_2))^{\mathcal{I}}$. Again by semantics of $\sqcap$ this is the case iff $x \in \big((CPE(L, G_1) \sqcup CPC(L, G_1)) \sqcap (CPE(\overline{L}, G_2) \sqcup CPC(\overline{L}, G_2))\big)^{\mathcal{I}}$. By distributivity this is equivalent to $x \in (Diss(A, G) \sqcup (CPE(L, G_1) \sqcap CPE(\overline{L}, G_2)))^{\mathcal{I}}$. But by construction the c-paths through $CPE(L, G_1) \sqcap CPE(\overline{L}, G_2)$ are exactly those containing the link A, and thus the interpretation of this concept expression is the empty set. Therefore, $x \in (Diss(A, G) \sqcup (CPE(L, G_1) \sqcap CPE(\overline{L}, G_2)))^{\mathcal{I}}$ is equivalent to $x \in (Diss(A, G))^{\mathcal{I}}$ which is what we wanted to show. □

By equivalence transformations and with the help of Lemma 1 the following lemma follows.

Lemma 3. *Let A and G be defined as in Definition 6. Then the following holds:*

$$Diss(A, G) \equiv (G_1 \sqcap CPC(\overline{L}, G_2)) \sqcup (CPC(L, G_1) \sqcap CPE(\overline{L}, G_2))$$
$$Diss(A, G) \equiv (CPE(L, G_1) \sqcap CPC(\overline{L}, G_2)) \sqcup (CPC(L, G_1) \sqcap G_2)$$

With the help of these terms it is easy to see how to remove links: Suppose a concept description C in NNF is given and it contains a link A. 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 $Diss(A, G_1 \sqcap G_2)$ we are able to remove the link.

The following example illustrates how to remove a role link.

Example 4. We consider the concept

$$G = (\exists R.(\neg E \sqcup \neg B) \sqcup D) \sqcap (A \sqcup \forall R.E) \sqcap (C \sqcup \forall R.B)$$

It is easy to see that G contains the role link $\{\exists R.(\neg E \sqcup \neg B), \forall R.B, \forall R.E\}$. As mentioned above we divide G according to this role link into $G_1 = \exists R.(\neg E \sqcup \neg B) \sqcup D$ and $G_2 = (A \sqcup \forall R.E) \sqcap (C \sqcup \forall R.B)$. In the next step we construct:

$$\begin{aligned} CPE(\{\exists R.(\neg E \sqcup \neg B)\}, G_1) &= \exists R.(\neg E \sqcup \neg B) \\ CPC(\{\exists R.(\neg E \sqcup \neg B)\}, G_1) &= D \\ CPE(\{\forall R.E, \forall R.B\}, G_2) &= \forall R.E \sqcap \forall R.B \\ CPC(\{\forall R.E, \forall R.B\}, G_2) &= (A \sqcap C) \sqcup (A \sqcap \forall R.B) \sqcup (\forall R.E \sqcap C) \end{aligned}$$

According to Lemma 3 we can substitute $G_1 \sqcap G_2$ in G which leads to the linkless concept description G', which is equivalent to G.

$$\begin{aligned} G' = &\big((\exists R.(\neg E \sqcup \neg B) \sqcup D) \sqcap ((A \sqcap C) \sqcup (A \sqcap \forall R.B) \sqcup (\forall R.E \sqcap C))\big) \sqcup \\ &\big(D \sqcap \forall R.E \sqcap \forall R.B\big) \end{aligned}$$

Next we give an algorithm to remove all concept and role 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.

Definition 7. *(make_linkless) Let G be a concept description.*

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

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

make_linkless(G) $\stackrel{def}{=}$ *make_linkless(G[B/make_linkless(B)]*
$QR.B$ with $Q \in \{\exists, \forall\}$ is a subconcept of G, B is not linkless and G contains neither concept nor role links.

Theorem 1. *Let G be a concept description. Then* make_linkless*(G) is equivalent to G and is linkless.*

Proof. The equivalence of G and the result of make_linkless(G) follows directly from Lemma 2. It remains to show that make_linkless(G) is actually linkless. Whenever there is a c-path p through a concept expression that contains a link A, by construction of a c-path the concept expression or one of its subexpression must consist of a conjunction $G_1 \sqcap G_2$, such that w.l.o.g. the positive part L of the link occurs in G_1 and the negative part $\overline{L}$ occurs in G_2. Thus $Diss$ is applicable. $Diss$ always removes c-paths, without introducing new c-paths. As there is a finite number of links, $Diss$ can only be applied a finite number of times and thus terminates. □

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

4 Properties of Linkless Concept Descriptions

In this section we consider the properties of linkless concepts in order to understand why it is desirable to transform a concept into a linkless concept.

4.1 Satisfiability

The first property we will investigate is the satisfiability of linkless concept descriptions.

Definition 8. *For a linkless concept description C the predicate $Sat(C)$ is defined as follows:*

1. $Sat(C) \stackrel{def}{=} \begin{cases} true, & \text{if } C \text{ is a concept literal,} \\ & \text{of the form } \forall R.D \text{ or } \top; \\ false, & \text{if } C \text{ is } \bot. \end{cases}$
2. $Sat(\exists R.D) \stackrel{def}{=} Sat(D)$.
3. $Sat(C = \sqcap_i \alpha_i) \stackrel{def}{=} true$, *iff* $\mathsf{Sat}(\alpha_i) = true$ *for all* i.
4. $Sat(C = \sqcup_i \alpha_i) \stackrel{def}{=} true$, *iff* $\mathsf{Sat}(\alpha_i) = true$ *for at least one* i.

Definition 8 can be directly transformed into an algorithm which checks the satisfiability of a linkless concept description. It is obvious that the Sat predicate has a linear time complexity. If we further assume that the simplifications in Figure 1 are applied to exhaustion after each step during the transformation of a concept C into a linkless concept C', $Sat(C')$ can be calculated in constant time. This is obvious since after these simplification a linkless concept C' description can only be inconsistent, if $C' = \bot$.

$$\begin{array}{ll} \top \sqcap C = C & \top \sqcup C = \top \\ \bot \sqcap C = \bot & \bot \sqcup C = C \\ \exists R.\bot = \bot & \forall R.\top = \top \end{array}$$

Fig. 1. Simplifications

4.2 Subsumption Queries

In Description Logics, besides the satisfiability of a concept, we are interested in subsumption checks as well. In [Dar01] an operator called conditioning is used as a technique to answer queries for a precompiled knowledge base. In Definition 9, $\overline{C}$ denotes the complement of a concept C, which is given in NNF and can be calculated simply by transforming $\neg C$ in NNF.

We will now transfer the idea of the conditioning operator to Description Logics.

Definition 9. ***(Conditioning Operator)*** *Let C be a linkless concept description and $\alpha = C_1 \sqcap \ldots \sqcap C_n$ where C_i is either a concept literal or has the form $\exists R.C_i'$ or $\forall R.C_i'$ with C_i' a concept literal. Then C conditioned by α denoted by $C|\alpha$ is the concept description one gets by replacing each occurrence of C_i in C by $\top$ and each occurrence of $\overline{C_i}$ by $\bot$ and simplifying the result according to Figure 1.*

The simplifications mentioned in the definition above can be easily performed on the fly during the calculation of conditioning. It is obvious that C' does not contain any concepts which occur in α as well as in C. 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.

Proposition 1. *Let C be a linkless concept description and D be a disjunction of concept literals and role restrictions $QR.B$ with $Q \in \{\exists, \forall\}$ where B is a concept literal. Then $C \sqsubseteq D$ can be tested in linear time.*

Because of the structure proposition 1 claims for the concept D, $\neg D$ has the structure of α in Definition 9. Since the subsumption $C \sqsubseteq D$ holds iff $C \sqcap \neg D$ is unsatisfiable, it is sufficient to calculate $Sat(C|\neg D \sqcap \neg D)$ instead. The conditioning as well as the Sat operator are linear, therefore the subsumption $C \sqsubseteq D$ can be checked in linear time as well.

5 Precompilation of TBoxes

So far we only considered single concept descriptions. We transformed a concept description into an equivalent linkless one and performed subsumption checks. However this approach can be easily extended to handle unfoldable TBoxes. In the first step we unfold the TBox as described in [BN03]. Next we use the introduced make_linkless operator to transform the right side of each concept description of the unfolded TBox into an equivalent linkless concept description. After that transformation subsumption queries $C \sqsubseteq D$ can be tested. This can be done by unfolding $C \sqcap \neg D$ according to the precompiled TBox. In general the result of this unfolding is not linkless. If the result of this unfolding is linkless, we can use the Sat predicate afterwards for a linear satisfiability test.

Further, if after the unfolding of $C \sqcap \neg D$, C is linkless and $\neg D$ is a disjunction of concept literals and role restrictions $QR.B$ with $Q \in \{\exists, \forall\}$ where B is a concept literal, the subsumption check $C \sqsubseteq D$ can performed in linear time by using the conditioning operator.

Precompiling every single concept description of the TBox seperately has the advantage that the linkless TBox can be extended very easily. New concept descriptions can be simply added after being precompiled as described above. So it is not necessary to precompile the whole TBox again in order to add concept descriptions.

6 Future Work / Conclusion

In the next step we want investigate how to extend our approach for TBoxes which are not unfoldable. Another very interesting point would be the extension of our normal form to more expressive Description Logics for example those allowing transitive roles. 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 poperty as well.

Acknowledgements

The authors are grateful to Alexander Fuchs for many helpful remarks during the development of this work.

References

[Bar96] Bart Selman and Henry Kautz. Knowledge Compilation and Theory Approximation. *J. ACM*, 43(2):193–224, 1996.

[BH98] Peter Balsiger and Alain Heuerding. Comparison of Theorem Provers for Modal Logics - Introduction and Summary. In Harrie C. M. de Swart, editor, *TABLEAUX*, volume 1397 of *Lecture Notes in Computer Science*, pages 25–26. Springer, 1998.

[BN03] F. Baader and W. Nutt. Basic Description Logics. In Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 43–95. Cambridge University Press, 2003.

[Dar01] Adnan Darwiche. Decomposable Negation Normal Form. *Journal of the ACM*, 48(4), 2001.

[Dar02] Adnan Darwiche. A Logical Approach to Factoring Belief Networks. In *Proceedings of KR*, pages 409–420, 2002.

[DH05] Adnan Darwiche and Jinbo Huang. DPLL with a Trace: From SAT to Knowledge Compilation. In *19th International Joint Conference on Artificial Intelligence (IJCAI 05)*, Nagoya, 2005.

[DM02] Adnan Darwiche and Pierre Marquis. A Knowlege Compilation Map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[Hor98] Ian R. Horrocks. Using an Expressive Description Logic: FaCT or Fiction? In Anthony G. Cohn, Lenhart Schubert, and Stuart C. Shapiro, editors, *KR'98: Principles of Knowledge Representation and Reasoning*, pages 636–645. Morgan Kaufmann, San Francisco, California, 1998.

[Hor03] Ian Horrocks. Implementation and Optimization Techniques. In Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors, *Description Logic Handbook*, pages 306–346. Cambridge University Press, 2003.

[Mac91] Robert M. MacGregor. Inside the LOOM Description Classifier. *SIGART Bulletin*, 2(3):88–92, 1991.

[MR93] Neil V. Murray and Erik Rosenthal. Dissolution: Making Paths Vanish. *J. ACM*, 40(3):504–535, 1993.

[MR03] Neil V. Murray and Erik Rosenthal. Tableaux, Path Dissolution, and Decomposable Negation Normal Form for Knowledge Compilation. In Marta Cialdea Mayer and Fiora Pirri, editors, *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 2796 of *Lecture Notes in Computer Science*. Springer, 2003.

[PSMB91] Peter F. Patel-Schneider, Deborah L. McGuinness, and Alexander Borgida. The CLASSIC Knowledge Representation System: Guiding Principles and Implementation Rationale. *SIGART Bulletin*, 2(3):108–113, 1991.

[RBG+97] Alan L. Rector, Sean Bechhofer, Carole A. Goble, Ian 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.

[TH06] Dmitry Tsarkov and Ian Horrocks. Description Logic Reasoner: System Description. In Ulrich Furbach and Natarajan Shankar, editors, *IJCAR*, volume 4130 of *Lecture Notes in Computer Science*, pages 292–297. Springer, 2006.

[Wer07] Christoph Wernhard. Tableau between proving projection and compilation. Technical Report 18/2007, Universität Koblenz-Landau, 2007.