Optimisation of Terminological Reasoning

Ian Horrocks
Department of Computer Science
University of Manchester, UK
`horrocks@cs.man.ac.uk`

Stephan Tobies
LuFG Theoretical Computer Science
RWTH Aachen, Germany
`tobies@informatik.rwth-aachen.de`

1 Motivation

The problem of computing concept subsumption relationships has been the subject of much research, and sound and complete algorithms are now known for a wide range of DLs (for example [9, 2, 7, 11]). However, in spite of the fundamental importance of terminologies in DLs, most of these algorithms deal only with the problem of deciding subsumption between two concepts (or, equivalently, concept satisfiability), without reference to a terminology (but see [4, 5, 8, 11]). By restricting the kinds of assertion that can appear in a terminology, concepts can be syntactically expanded so as to explicitly include all relevant terminological information. This procedure, called *unfolding*, has mostly been applied to less expressive DLs. With more expressive DLs, in particular those supporting universal roles, it is often possible to encapsulate an arbitrary terminology in a single concept. This technique can be used with satisfiability testing to ensure that the result is valid with respect to the assertions in the terminology, a procedure called *internalisation*.

Although the above mentioned techniques suffice to demonstrate the theoretical adequacy of satisfiability decision procedures for terminological reasoning, experiments with implementations have shown that, for reasons of (lack of) efficiency, they are highly unsatisfactory as a practical methodology for reasoning with DL terminologies. Firstly, experiments with the Kris system have shown that integrating unfolding with the (tableaux) satisfiability algorithm (*lazy unfolding*) leads to a significant improvement in performance [1]. More recently, experiments with the FaCT system have shown that reasoning becomes hopelessly intractable when internalisation is used to deal with larger terminologies [10]. However, the FaCT system has also demonstrated that this problem can be dealt with (at least for realistic terminologies) by using a combination of lazy unfolding and internalisation, having first manipulated the terminology in order to minimise the number of assertions that must be dealt with by internalisation (a technique called *absorption*). It should be noted that, although these techniques were discovered while developing DL systems, they are applicable to a whole range of reasoning systems, independent of the concrete logic and type of algorithm.

In this paper we seek to improve our theoretical understanding of these important techniques which has, until now, been very limited. In particular we would like to know exactly when and how they can be applied, and be sure that the answers we get from the algorithm are still correct. This is achieved by defining a formal framework that allows the techniques to be precisely described, establishing conditions under which they can be safely applied, and proving that, provided these conditions are respected, satisfiability algorithms will still function correctly. Finally, we identify several interesting directions for future research, in particular the problem of finding the "best" absorption possible. Due to space limitations, we have omitted most proofs. Please refer to [12] for full details.

2 Preliminaries

Firstly, we will establish some basic definitions that clarify what we mean by a DL, a terminology (subsequently called a TBox), and subsumption and satisfiability with respect to a terminology. The results in this paper are uniformly applicable to a whole range of DLs, as long as some basic criteria are met:

Definition 2.1 (Description Logic) *Let* L *be a DL based on infinite sets of atomic concepts* NC *and atomic roles* NR. *We will identify* L *with the sets of its well-formed concepts and require* L *to be closed under boolean operations and sub-concepts.*

An interpretation $\mathcal{I}$ *is a pair* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, *where* $\Delta^{\mathcal{I}}$ *is a non-empty set and* $\cdot^{\mathcal{I}}$ *is a function mapping* NC *to* $2^{\Delta^{\mathcal{I}}}$ *and* NR *to* $2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}}$. *With each DL* L *we associate a set* $\mathsf{Int}(\mathsf{L})$ *of* admissible *interpretations for* L. $\mathsf{Int}(\mathsf{L})$ *must be closed under isomorphisms, and, for any two interpretations* $\mathcal{I}$ *and* $\mathcal{I}'$ *that agree on* NR, *it must satisfy* $\mathcal{I} \in \mathsf{Int}(\mathsf{L}) \Leftrightarrow \mathcal{I}' \in \mathsf{Int}(\mathsf{L})$. *Additionally, we assume that each DL* L *comes with a semantics that allows any interpretation* $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *to be extended to each concept* $C \in \mathsf{L}$ *such that it satisfies the following conditions:*

(I1) it maps the boolean combination of concepts to the corresponding boolean combination of their interpretations, and

(I2) the interpretation $C^{\mathcal{I}}$ *of a compound concept* $C \in \mathsf{L}$ *depends only on the interpretation of those atomic concepts and roles that appear syntactically in* C.

This definition captures a whole range of DLs, namely, the important DL $\mathcal{ALC}$ [15] and its many extensions. $\mathsf{Int}(\mathsf{L})$ hides restrictions on the interpretation of certain roles like transitivity, functionality, or role hierarchies, which are imposed by more expressive DLs (e.g., [11]), as these are irrelevant for our purposes. We will use $C \rightarrow D$ as an abbreviation for $\neg C \sqcup D$, $C \leftrightarrow D$ as an abbreviation for $(C \to D) \sqcap (D \to C)$, and $\top$ as a tautological concept, e.g., $A \sqcup \neg A$ for an arbitrary $A \in \mathsf{NC}$.

A TBox consists of a set of axioms asserting subsumption or equality relations between (possibly complex) concepts.

Definition 2.2 (TBox, Satisfiability) *A* TBox $\mathcal{T}$ *for* L *is a finite set of axioms of the form* $C_1 \sqsubseteq C_2$ *or* $C_1 \doteq C_2$*, where* $C_i \in \mathsf{L}$*. If, for some* $A \in \mathsf{NC}$*,* $\mathcal{T}$ *contains an axiom of the form* $A \sqsubseteq C$ *or* $A \doteq C$*, then we say that* A *is* defined *in* $\mathcal{T}$*.*

Let L *be a DL and* $\mathcal{T}$ *a TBox. An interpretation* $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *is a* model *of* $\mathcal{T}$ *iff, for each* $C_1 \sqsubseteq C_2 \in \mathcal{T}$*,* $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ *holds, and, for each* $C_1 \doteq C_2 \in \mathcal{T}$*,* $C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$ *holds. In this case we write* $\mathcal{I} \models \mathcal{T}$*. A concept* $C \in \mathsf{L}$ *is* satisfiable *with respect to a TBox* $\mathcal{T}$ *iff there is an* $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *with* $\mathcal{I} \models \mathcal{T}$ *and* $C^{\mathcal{I}} \neq \emptyset$*. A concept* $C \in \mathsf{L}$ subsumes *a concept* $D \in \mathsf{L}$ *w.r.t.* $\mathcal{T}$ *iff, for all* $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *with* $\mathcal{I} \models \mathcal{T}$*,* $C^{\mathcal{I}} \supseteq D^{\mathcal{I}}$ *holds. Two TBoxes* $\mathcal{T}, \mathcal{T}'$ *are called* equivalent *(*$\mathcal{T} \equiv \mathcal{T}'$*), iff, for all* $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$*,* $\mathcal{I} \models \mathcal{T}$ iff $\mathcal{I} \models \mathcal{T}'$*.*

We will only deal with concept satisfiability as concept subsumption can be reduced to it for DLs that are closed under boolean operations: C subsumes D w.r.t. $\mathcal{T}$ iff $(D \sqcap \neg C)$ is not satisfiable w.r.t. $\mathcal{T}$.

For temporal or modal logics, satisfiability with respect to a set of formulae $\{C_1, \ldots, C_k\}$ asserted to be universally true corresponds to satisfiability w.r.t. the TBox $\{\top \doteq C_1, \ldots, \top \doteq C_n\}$.

Many decision procedures for DLs base their judgement on the existence of models or pseudo-models for concepts. A central rôle in these algorithms is played by a structure that we will call a *witness*. It generalises the notions of *tableaux* that appear in DL tableau-algorithms [9, 11].

Definition 2.3 (Witness) *Let* L *be a DL and* $C \in \mathsf{L}$ *a concept. A* witness $\mathcal{W} = (\Delta^{\mathcal{W}}, \cdot^{\mathcal{W}}, \mathcal{L}^{\mathcal{W}})$ *for* C *consists of a non-empty set* $\Delta^{\mathcal{W}}$*, a function* $\cdot^{\mathcal{W}}$ *that maps* NR *to* $2^{\Delta^{\mathcal{W}} \times \Delta^{\mathcal{W}}}$*, and a function* $\mathcal{L}^{\mathcal{W}}$ *that maps* $\Delta^{\mathcal{W}}$ *to* 2^{L} *such that:*

(W1) there is some $x \in \Delta^{\mathcal{W}}$ *with* $C \in \mathcal{L}^{\mathcal{W}}(x)$*,*

(W2) there is an interpretation $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *that* stems *from* $\mathcal{W}$*, and*

(W3) for each interpretation $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *that* stems *from* $\mathcal{W}$*, it holds that* $D \in \mathcal{L}^{\mathcal{W}}(x)$ *implies* $x \in D^{\mathcal{I}}$*.*

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *is said to stem from* $\mathcal{W}$ *if* $\Delta^{\mathcal{I}} = \Delta^{\mathcal{W}}$*,* $\cdot^{\mathcal{I}}|_{\mathsf{NR}} = \cdot^{\mathcal{W}}$*, and for each* $A \in \mathsf{NC}$*,* $A \in \mathcal{L}^{\mathcal{W}}(x)$ implies $x \in A^{\mathcal{I}}$ *and* $\neg A \in \mathcal{L}^{\mathcal{W}}(x)$ implies $x \notin A^{\mathcal{I}}$*.*

A witness $\mathcal{W}$ *is called* admissible *with respect to a TBox* $\mathcal{T}$ *if there is an interpretation* $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ *that stems from* $\mathcal{W}$ *with* $\mathcal{I} \models \mathcal{T}$*.*

Please note that, for any witness $\mathcal{W}$, (W2) together with Condition 3 of "stemming" implies that, there exists no $x \in \Delta^{\mathcal{W}}$ and $A \in \mathsf{NC}$, such that

$\{A, \neg A\} \subseteq \mathcal{L}^{\mathcal{W}}(x)$. Also note that, in general, more than one interpretation may stem from a witness. This is the case if, for an atomic concept $A \in \mathsf{NC}$ and an element $x \in \Delta^{\mathcal{W}}$, $\mathcal{L}^{\mathcal{W}}(x) \cap \{A, \neg A\} = \emptyset$ holds. The existence of admissible witnesses is closely related to the satisfiability of concepts w.r.t. TBoxes:

Lemma 2.4 *Let L be a DL. A concept $C \in \mathsf{L}$ is satisfiable w.r.t. a TBox $\mathcal{T}$ iff it has a witness that is admissible w.r.t. $\mathcal{T}$.*

From this it follows that one can test the satisfiability of a concept w.r.t. to a TBox by checking for the existence of an admissible witness. We call algorithms that utilise this approach *model-building algorithms.*

This notion captures tableau-based decision procedures [9, 11] and, due to their direct correspondence with tableaux algorithms [13, 3], even resolution based and sequent calculus algorithms. This work develops a technique applicable to all these algorithm types.

Many decision procedures for DLs deal with TBoxes by exploiting the following lemma.

Lemma 2.5 *Let L be a DL, $C \in \mathsf{L}$ a concept, and $\mathcal{T}$ a TBox. Let $\mathcal{W}$ be a witness for C. $\mathcal{W}$ is admissible w.r.t. $\mathcal{T}$ if, for each $x \in \Delta^{\mathcal{W}}$,*

$$\begin{array}{lll} C_1 \sqsubseteq C_2 \in \mathcal{T} & \text{implies} & C_1 \rightarrow C_2 \in \mathcal{L}^{\mathcal{W}}(x) \\ C_1 \doteq C_2 \in \mathcal{T} & \text{implies} & C_1 \leftrightarrow C_2 \in \mathcal{L}^{\mathcal{W}}(x). \end{array}$$

Examples of algorithms that exploit this lemma to deal with axioms can be found in [8, 6, 11], where, for each axiom $C_1 \sqsubseteq C_2$ ($C_1 \doteq C_2$) the concept $C_1 \rightarrow C_2$ ($C_1 \leftrightarrow C_2$) is added to every node of the generated tableau.

Dealing with general axioms in this manner is costly due to the high degree of nondeterminism introduced. This can best be understood by looking at tableaux algorithms, which try to build witnesses in an incremental fashion. For a concept C to be tested for satisfiability, they start with $\Delta^{\mathcal{W}} = \{x_0\}$, $\mathcal{L}^{\mathcal{W}}(x_0) = \{C\}$ and $\cdot^{\mathcal{W}}(R) = \emptyset$ for each $R \in \mathsf{NR}$. Subsequently, the concepts in $\mathcal{L}^{\mathcal{W}}$ are decomposed and, if necessary, new nodes are added to $\Delta^{\mathcal{W}}$, until either $\mathcal{W}$ is a witness for C, or an obvious contradiction of the form $\{A, \neg A\} \subseteq \mathcal{L}^{\mathcal{W}}(x)$, which violates (W2), is generated. In the latter case, backtracking search is used to explore alternative non-deterministic decompositions (e.g., of disjunctions), one of which could lead to the discovery of a witness.

When applying Lemma 2.5, disjunctions are added to the label of each node of the tableau for each general axiom in the TBox (one disjunction for axioms of the form $C_1 \sqsubseteq C_2$, two for axioms of the form $C_1 \doteq C_2$). This leads to an exponential increase in the search space as the number of nodes and axioms increases. For example, with 10 nodes and a TBox containing 10 general axioms (of the form $C_1 \sqsubseteq C_2$) there are already 100 disjunctions, and they can be non-deterministically decomposed in 2^{100} different ways. For a TBox containing large numbers of general axioms (there are 1,214 in the GALEN medical terminology KB [14]) this can degrade performance to the extent that subsumption testing is effectively non-terminating. To reason with this kind of TBox we must find a more efficient way to deal with axioms.

3 Absorptions

We start our considerations with an analysis of a technique that can be used to deal more efficiently with so-called primitive or acyclic TBoxes.

Definition 3.1 (Absorption) *Let L be a DL and $\mathcal{T}$ a TBox. An* absorption *of $\mathcal{T}$ is a pair of TBoxes $(\mathcal{T}_u, \mathcal{T}_g)$ such that $\mathcal{T} \equiv \mathcal{T}_u \cup \mathcal{T}_g$ and $\mathcal{T}_u$ contains only axioms of the form $A \sqsubseteq D$ and $\neg A \sqsubseteq D$ where $A \in \mathsf{NC}$.*

An absorption $(\mathcal{T}_u, \mathcal{T}_g)$ of $\mathcal{T}$ is called correct *if it satisfies the following condition. For each each witness $\mathcal{W}$ and $x \in \Delta^{\mathcal{W}}$, if*

$$\left.\begin{array}{rll} A \sqsubseteq D \in \mathcal{T}_u \text{ and } A \in \mathcal{L}^{\mathcal{W}}(x) & \text{implies} & D \in \mathcal{L}^{\mathcal{W}}(x) \\ \neg A \sqsubseteq D \in \mathcal{T}_u \text{ and } \neg A \in \mathcal{L}^{\mathcal{W}}(x) & \text{implies} & D \in \mathcal{L}^{\mathcal{W}}(x) \\ C_1 \sqsubseteq C_2 \in \mathcal{T}_g & \text{implies} & C_1 \rightarrow C_2 \in \mathcal{L}^{\mathcal{W}}(x) \\ C_1 \doteq C_2 \in \mathcal{T}_g & \text{implies} & C_1 \leftrightarrow C_2 \in \mathcal{L}^{\mathcal{W}}(x) \end{array}\right\} (*)$$

then $\mathcal{W}$ is admissible w.r.t. $\mathcal{T}$. A witness that satisfies $(*)$ will be called unfolded.

How does a correct absorption enable an algorithm to deal with axioms more efficiently? This is best described by returning to tableaux algorithms. Instead of dealing with axioms as previously described, which may lead to an exponential increase in the search space, axioms in $\mathcal{T}_u$ can now be dealt with in a deterministic manner. Assume, for example, that we have to handle the axiom $A \doteq C$. If the label of a node already contains A (resp. $\neg A$), then C (resp. $\neg C$) is added to the label; if the label contains neither A nor $\neg A$, then *nothing* has to be done. Dealing with the axioms in $\mathcal{T}_u$ this way avoids the necessity for additional non-deterministic choices and leads to a gain in efficiency. A witness produced in this manner will be unfolded and is a certificate for satisfiability w.r.t. $\mathcal{T}$. This technique is generally known as *lazy unfolding* of primitive TBoxes [10]; formally, it is justified by the following lemma:

Lemma 3.2 *Let $(\mathcal{T}_u, \mathcal{T}_g)$ be a correct absorption of $\mathcal{T}$. For any $C \in \mathsf{L}$, C has a witness that is admissible w.r.t. $\mathcal{T}$ iff C has an unfolded witness.*

A family of TBoxes where absorption can successfully be applied are *primitive* TBoxes, the most simple form of TBox usually studied in the literature.

Definition 3.3 (Primitive TBox) *A TBox $\mathcal{T}$ is called* primitive *iff it consists entirely of axioms of the form $A \doteq D$ with $A \in \mathsf{NC}$, each $A \in \mathsf{NC}$ appears as at most one left-hand side of an axiom, and $\mathcal{T}$ is acyclic. Acyclicity is defined as* *follows:* $A \in \mathsf{NC}$ *is said to* directly use $B \in \mathsf{NC}$ *if* $A \doteq D \in \mathcal{T}$ *and* B *occurs in* D*;* uses *is the transitive closure of "directly uses". We say that* $\mathcal{T}$ *is* acyclic *if there is no* $A \in \mathsf{NC}$ *that uses itself.*

For primitive TBoxes a correct absorption can easily be given.

Theorem 3.4 *Let* $\mathcal{T}$ *be a primitive TBox,* $\mathcal{T}_g = \emptyset$*, and* $\mathcal{T}_u$ *defined by*

$$\mathcal{T}_u = \{A \sqsubseteq D, \neg A \sqsubseteq \neg D \mid A \doteq D \in \mathcal{T}\}.$$

Then $(\mathcal{T}_u, \mathcal{T}_g)$ *is a correct absorption of* $\mathcal{T}$*.*

Lazy unfolding is a well-known and widely used technique for optimising reasoning w.r.t. primitive TBoxes [1]. It is a relatively simple approach, although one that is independent of a specific DL or reasoning algorithm. With the next lemma we show how we can extend correct absorptions and hence how lazy unfolding can be applied to a broader class of TBoxes.

Lemma 3.5 *Let* $(\mathcal{T}_u, \mathcal{T}_g)$ *be a correct absorption of a TBox* $\mathcal{T}$*.*

1. *If* $\mathcal{T}'$ *is an arbitrary TBox, then* $(\mathcal{T}_u, \mathcal{T}_g \cup \mathcal{T}')$ *is a correct absorption of* $\mathcal{T} \cup \mathcal{T}'$*.*
2. *If* $\mathcal{T}'$ *is a TBox that consists entirely of axioms of the form* $A \sqsubseteq D$*, where* $A \in \mathsf{NC}$ *and* A *does not occur on the left-hand side of any axiom in* $\mathcal{T}_u$*, then* $(\mathcal{T}_u \cup \mathcal{T}', \mathcal{T}_g)$ *is a correct absorption of* $\mathcal{T} \cup \mathcal{T}'$*.*

4 Application to FaCT

In the preceding section we have defined correct absorptions and discussed how they can be exploited in order to optimise satisfiability procedures. However, we have said nothing about the problem of how to find an absorption given an arbitrary terminology. In this section we will describe the absorption algorithm used by FaCT and prove that it generates correct absorptions.

Given a TBox $\mathcal{T}$ containing arbitrary axioms, the absorption algorithm used by FaCT constructs a triple of TBoxes $(\mathcal{T}_g, \mathcal{T}_{\text{prim}}, \mathcal{T}_{\text{inc}})$ such that

- $\mathcal{T} \equiv \mathcal{T}_g \cup \mathcal{T}_{\text{prim}} \cup \mathcal{T}_{\text{inc}}$,
- $\mathcal{T}_{\text{prim}}$ is primitive, and
- $\mathcal{T}_{\text{inc}}$ consists only of axioms of the form $A \sqsubseteq D$ where $A \in \mathsf{NC}$ and A is not defined in $\mathcal{T}_{\text{prim}}$.

We refer to these properties by $(*)$. From Theorem 3.4 together with Lemma 3.5 it follows that, for

$$\mathcal{T}_u := \{A \sqsubseteq D, \neg A \sqsubseteq \neg D \mid A \doteq D \in \mathcal{T}_{\text{prim}}\} \cup \mathcal{T}_{\text{inc}}$$

$(\mathcal{T}_u, \mathcal{T}_g)$ is a correct absorption of $\mathcal{T}$; hence satisfiability for a concept C w.r.t. $\mathcal{T}$ can be decided by checking for an unfolded witness for C.

In a first step, FaCT distributes axioms from $\mathcal{T}$ amongst $\mathcal{T}_{\text{inc}}$, $\mathcal{T}_{\text{prim}}$, and $\mathcal{T}_g$, trying to minimise the number of axioms in $\mathcal{T}_g$ while still maintaining $(*)$. To do this, it initialises $\mathcal{T}_{\text{prim}}, \mathcal{T}_{\text{inc}}$, and $\mathcal{T}_g$ with $\emptyset$, and then processes each axiom $X \in \mathcal{T}$ as follows.

1. If X is of the form $A \sqsubseteq C$, then
   (a) if $A \in \mathsf{NC}$ and A is not defined in $\mathcal{T}_{\text{prim}}$ then X is added to $\mathcal{T}_{\text{inc}}$,
   (b) otherwise X is added to $\mathcal{T}_g$
2. If X is of the form $A \doteq C$, then
   (a) if $A \in \mathsf{NC}$, A is not defined in $\mathcal{T}_{\text{prim}}$ or $\mathcal{T}_{\text{inc}}$ and $\mathcal{T}_{\text{prim}} \cup \{X\}$ is primitive, then X is added to $\mathcal{T}_{\text{prim}}$,
   (b) otherwise, the axioms $A \sqsubseteq C$ and $C \sqsubseteq A$ are added to $\mathcal{T}_g$
3. If X is of the form $C \sqsubseteq D$, then add $C \sqsubseteq D$ to $\mathcal{T}_g$.
4. If X is of the form $C \doteq D$, then add $C \sqsubseteq D$ and $D \sqsubseteq C$ to $\mathcal{T}_g$.

It is easy to see that the resulting TBoxes $\mathcal{T}_g, \mathcal{T}_{\text{prim}}, \mathcal{T}_{\text{inc}}$ satisfy $(*)$. In a second step, FaCT processes the axioms in $\mathcal{T}_g$ one at a time, trying to absorb them into axioms in $\mathcal{T}_{\text{inc}}$. Those axioms that are not absorbed remain in $\mathcal{T}_g$. To give a simpler formulation of the algorithm, each axiom $(C \sqsubseteq D) \in \mathcal{T}_g$ is viewed as a clause $\mathbf{G} = \{D, \neg C\}$, corresponding to the axiom $\top \sqsubseteq C \rightarrow D$, which is equivalent to $C \sqsubseteq D$. For each such axiom FaCT applies the following absorption procedure.

1. Try to absorb $\mathbf{G}$. If there is a concept $\neg A \in \mathbf{G}$ such that $A \in \mathsf{NC}$ and A is not defined in $\mathcal{T}_{\text{prim}}$, then add $A \sqsubseteq B$ to $\mathcal{T}_{\text{inc}}$, where B is the disjunction of all the concepts in $\mathbf{G} \setminus \{\neg A\}$, remove $\mathbf{G}$ from $\mathcal{T}_g$, and exit.
2. Try to simplify $\mathbf{G}$.
   (a) If there is some $\neg C \in \mathbf{G}$ such that C is of the form $C_1 \sqcap \ldots \sqcap C_n$, then substitute $\neg C$ with $\neg C_1 \sqcup \ldots \sqcup \neg C_n$, and continue with step 2b.
   (b) If there is some $C \in \mathbf{G}$ such that C is of the form $(C_1 \sqcup \ldots \sqcup C_n)$, then apply associativity by setting $\mathbf{G} = \mathbf{G} \cup \{C_1, \ldots, C_n\} \setminus \{(C_1 \sqcup \ldots \sqcup C_n)\}$, and return to step 1.
3. Try to unfold $\mathbf{G}$. If, for some $A \in \mathbf{G}$ (resp. $\neg A \in \mathbf{G}$), there is an axiom $A \doteq C$ in $\mathcal{T}_{\text{prim}}$, then substitute $A \in \mathbf{G}$ (resp. $\neg A \in \mathbf{G}$) with C (resp. $\neg C$) and return to step 1.
4. If none of the above were possible, then absorption of $\mathbf{G}$ has failed. Leave $\mathbf{G}$ in $\mathcal{T}_g$, and exit.

We have to show that each step maintains $(*)$. Dealing with clauses instead of axioms causes no problems. In the first step, axioms are moved from $\mathcal{T}_g$ to $\mathcal{T}_{\text{inc}}$ as long as this does not violate $(*)$. The second and the third step replace a clause by an equivalent one and hence do not violate $(*)$. Termination is obvious: each axiom is considered only once and, for a given axiom, simplification and unfolding can only be applied finitely often before the procedure is exited, either by absorbing the axiom into $\mathcal{T}_{\text{inc}}$ or leaving it in $\mathcal{T}_g$. For simplification, this is obvious; for unfolding, this holds because $\mathcal{T}_{\text{prim}}$ is acyclic.

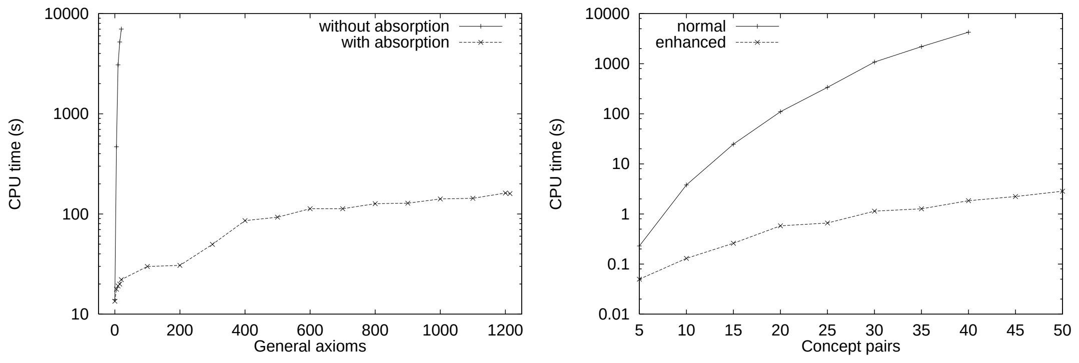

Figure 1: Classification times with(out) absorption (left) and enhanced absorption

Theorem 4.1 *For any TBox $\mathcal{T}$, FaCT computes a correct absorption of $\mathcal{T}$.*

The absorption algorithm employed by FaCT already leads to a dramatic improvement in performance. This is illustrated by Figure 1 (left), which shows the times taken by FaCT to classify versions of the GALEN KB with some or all of the general axioms removed. Without absorption, classification time increased rapidly with the number of general axioms, and exceeded 10,000s with only 25 general axioms in the KB; with absorption, only 160s was taken to classify the KB with all 1,214 general axioms.

However, there is still considerable scope for further gains. In particular, the following definition for a *stratified* TBox allows lazy unfolding to be more generally applied, while still allowing for correct absorptions.

Definition 4.2 (Stratified TBox) *A TBox $\mathcal{T}$ is called* stratified *iff it consists entirely of axioms of the form $A \doteq D$ with $A \in \mathsf{NC}$, each $A \in \mathsf{NC}$ appears at most once on the left-hand side of an axiom, and $\mathcal{T}$ can be arranged monotonously, i.e., there is a disjoint partition $\mathcal{T}_1 \dot{\cup} \mathcal{T}_2 \dot{\cup} \ldots \dot{\cup} \mathcal{T}_k$ of $\mathcal{T}$, such that:*

- *for all $1 \leq j < i \leq k$, if $A \in \mathsf{NC}$ is defined in $\mathcal{T}_i$, then it does not occur in $\mathcal{T}_j$,*
- *for all $1 \leq i \leq k$, all concepts which appear on the right-hand side of axioms in $\mathcal{T}_i$ are monotone in all atomic concepts defined in $\mathcal{T}_i$.*

A concept C is monotone in an atomic concept A if, for any interpretation $\mathcal{I} \in \mathsf{Int}(\mathsf{L})$ and any two sets $X_1, X_2 \subseteq \Delta^{\mathcal{I}}$, $X_1 \subseteq X_2$ implies $C^{\mathcal{I}[A \mapsto X_1]} \subseteq C^{\mathcal{I}[A \mapsto X_2]}$, where, for some interpretation $\mathcal{I}$, $\mathcal{I}[A \mapsto X]$ denotes the interpretation that maps A to X and agrees with $\mathcal{I}$ on all other atomic concepts and roles.

For many DLs, a sufficient condition for monotonicity is *syntactic* monotonicity, i.e., a concept C is syntactically monotone in some atomic concept A if A appears in C only in the scope of an even number of negations. This holds, e.g., for $\mathcal{SHIQ}$ [11], if *at-most* qualifying number restrictions $(\leq n\ R\ C)$ are counted as one negation.

Theorem 4.3 *Let $\mathcal{T}$ be a stratified TBox, $\mathcal{T}_g = \emptyset$ and $\mathcal{T}_u$ defined by*

$$\mathcal{T}_u = \{A \sqsubseteq D, \neg A \sqsubseteq \neg D \mid A \doteq D \in \mathcal{T}\}.$$

Then $(\mathcal{T}_u, \mathcal{T}_g)$ is a correct absorption of $\mathcal{T}$.

Please note, that the partition of $\mathcal{T}$ into strata is necessary only to guarantee the correctness of the absorption and does not need to be taken into account for the lazy unfolding itself. Lazy unfolding is generally applicable to all correct absorptions without any modifications. Also note that it is possible that a TBox is stratified with only a single stratum, in which case the first condition of Definition 4.2 is trivially satisfied.

The effectiveness of this enhanced absorption can be demonstrated by a simple experiment with the new FaCT system, which implements the $\mathcal{SHIQ}$ logic [11] and is thus able to deal with inverse roles. Figure 1 (right) shows the classification time in seconds using the normal and enhanced absorption algorithms for terminologies consisting of between 5 and 50 pairs of cyclical definitions. With only 10 pairs the gain in performance is already a factor of 30, while for 45 and 50 pairs it has reached several orders of magnitude: with the enhanced lazy unfolding the terminology is classified in 2–3 seconds whereas with the original algorithm the time required exceeded the 10,000 second limit imposed in the experiment.

It is worth pointing out that it is by no means trivially true that cyclical definitions can be dealt with by lazy unfolding. It is clear that $A \doteq \neg A$ (or more subtle variants) force the domain to be empty and would lead to an incorrect absorption if dealt with by lazy unfolding. With converse roles, definitions like $A \doteq \forall R.(\forall R^{-}.\neg A)$ force the interpretation of a role R to be empty, again leading to an incorrect absorption if dealt with by lazy unfolding.

5 Optimal Absorptions

Our results show that absorption is a highly effective and widely applicable technique, and by formally defining correctness criteria for absorptions we can prove that the procedure used by FaCT finds correct absorptions. Moreover, by establishing more precise correctness criteria we have demonstrated how the effectiveness of this procedure could be further enhanced.

However, the absorption algorithm used by FaCT is clearly sub-optimal, in the sense that changes could be made that would, in general, allow more axioms to be absorbed (e.g., by also giving special consideration to axioms of the form $\neg A \sqsubseteq C$ with $A \in \mathsf{NC}$). Moreover, the procedure is non-deterministic, and, while it is guaranteed to produce a correct absorption, its specific result depends on the order of the axioms in the original TBox $\mathcal{T}$. Since the semantics of a TBox $\mathcal{T}$ does not depend on the order of its axioms, there is no reason to suppose that they will be arranged in a way that yields a “good” absorption. Given the effectiveness of absorption, it would be desirable to have an algorithm that was guaranteed to find the "best" absorption possible for any set of axioms, irrespective of their ordering in the TBox. Unfortunately, it is not even clear how to define a sensible optimality criterion for absorptions. It is obvious that simplistic approaches based on the number or size of axioms remaining in $\mathcal{T}_g$ will not lead to a useful solution for this problem. Consider, for example, the cyclical TBox experiment from the previous section. Both the original FaCT absorption algorithm and the enhanced algorithm, which performs the absorption of cyclical TBoxes, are able to compute a complete absorption of the axioms used in the experiment (i.e., a correct absorption with $\mathcal{T}_g = \emptyset$), but the enhanced algorithm leads to much better performance, as shown in Figure 1 (right). An important issue for future work is, therefore, the identification of a suitable optimality criterion for absorptions, and the development of an algorithm that is able to compute absorptions that are optimal with respect to this criterion.

References

[1] F. Baader, E. Franconi, B. Hollunder, B. Nebel, and H.-J. Profitlich. An empirical analysis of optimization techniques for terminological representation systems, or: Making KRIS get a move on. *Applied Artificial Intelligence*, 4:109–132, 1994.

[2] F. Baader and B. Hollunder. A terminological knowledge representation system with complete inference algorithms. In *Proc. of PDK'91*, Springer-Verlag.

[3] A. Borgida, E. Franconi, I. Horrocks, D. McGuinness, and P. F. Patel-Schneider. Explaining $\mathcal{ALC}$ subsumption. In *Proc. of DL'99*, pages 37–40, 1999.

[4] M. Buchheit, F. M. Donini, and A. Schaerf. Decidable reasoning in terminological knowledge representation systems. *J. of Artificial Intelligence Research*, 1:109–138, 1993.

[5] D. Calvanese. Reasoning with inclusion axioms in description logics: Algorithms and complexity. In Wolfgang Wahlster, editor, *Proc. of ECAI'96*, pages 303–307. John Wiley & Sons Ltd., 1996.

[6] G. De Giacomo and M. Lenzerini. TBox and ABox reasoning in expressive description logics. In *Proc. of KR-96*, pages 316–327. M. Kaufmann, Los Altos, 1996.

[7] G. De Giacomo and F. Massacci. Combining deduction and model checking into tableaux and algorithms for converse-PDL. *Information and Computation*, to appear.

[8] F. Donini, G. De Giacomo, and F. Massacci. EXPTIME tableaux for $\mathcal{ALC}$. In *Collected Papers from DL'96.*

[9] B. Hollunder, W. Nutt, and M. Schmidt-Schauss. Subsumption algorithms for concept description languages. In *Proc. of ECAI-90*, Pitman Publishing, London, 1990.

[10] I. Horrocks. Using an expressive description logic: FaCT or fiction? In *Proc. of KR'98*, pages 636–647. Morgan Kaufmann Publishers, 1998.

[11] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Proc. of LPAR'99*, 1999.

[12] I. Horrocks and S. Tobies. Reasoning with Axioms: Theory and Practice. In *Proc. of KR 2000.* Morgan Kaufman Publishers, 2000.

[13] U. Hustadt and R. A. Schmidt. On the relation of resolution and tableaux proof systems for description logics. In *Proc. of IJCAI-99*, pages 110–115, 1999.

[14] A. L. Rector, W A Nowlan, and A Glowinski. Goals for concept representation in the Galen project. In *Proc. of SCAMC'93*, pages 414–418, Washington DC, USA, 1993.

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