Combining Tableaux and Algebraic Methods for Reasoning with Qualified Number Restrictions

Volker Haarslev, Martina Timmann, Ralf Möller
University of Hamburg, Computer Science Department
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany

Abstract

This paper investigates an optimization technique for reasoning with *qualified number restrictions* in the description logic $\mathcal{ALCQH}_{R^+}$. We present a hybrid architecture where a standard tableaux calculus is combined with a procedure deciding the satisfiability of linear (in)equations derived from qualified number restrictions. The advances are demonstrated by an empirical evaluation using the description logic system RACER which implements TBox and ABox reasoning for $\mathcal{ALCQHI}_{R^+}$. The evaluation demonstrates a dramatic speed up compared to other known approaches.

1 Introduction

Over the last few years many optimizations techniques have been proposed for dealing with expressive description logics (DLs). These techniques turned out to be very effective for synthesized TBoxes as well as application TBoxes. However, there exist only very few proposals for optimization techniques addressing the sources of complexity introduced by qualified number restrictions. In [7] mathematical programming and atomic decomposition is presented as the basic TBox inference technique for a large class of modal and description logics. The proposed techniques seem to be well suited for dealing with qualified number restrictions. However, the approach in [7] is not based on a tableaux calculus and cannot deal with the description logic $\mathcal{ALCQH}_{R^+}$. In this paper we report on the integration of an algebraic reasoner into the DL reasoner RACER[1] [4] whose architecture is based on a highly optimized tableaux calculus for the DL $\mathcal{ALCQH}_{R^+}$ which extends $\mathcal{ALCNH}_{R^+}$ [2] by qualified number restrictions.

In [3] a particular tableaux calculus for $\mathcal{ALCQH}_{R^+}$, the so-called signature calculus, is presented. The signature calculus addresses a source of inefficiency which is caused by standard tableaux calculi dealing with qualified number restrictions (e.g. see [5] for $\mathcal{ALCQ}$). The signature calculus offers a compact

[1] RACER download page: `http://kogs-www.informatik.uni-hamburg.de/~race/`

representation for role successors using so-called proxy individuals. A proxy individual represents a set of role successors and its corresponding signature can be understood as a representation for the cardinality of a set of qualified role successors (see [3] for details). This compact representation of role successors is independent from the values of numbers occurring in qualified number restrictions. The use of the signature calculus already indicates a dramatic performance gain of several orders of magnitude (see [3] and Figure 3).

However, there still exists a large class of problems (using qualified number restrictions) which cannot be efficiently dealt with by the signature calculus. One of these problems is illustrated as follows. Let us assume a role name R, the concept names $\mathsf{C}_1, \ldots, \mathsf{C}_7, \mathsf{P}_1, \ldots, \mathsf{P}_4$, the axioms $\mathsf{C}_3 \sqsubseteq \mathsf{C}_1$, $\mathsf{C}_4 \sqsubseteq \mathsf{C}_1$, $\mathsf{C}_5 \sqsubseteq \mathsf{C}_1 \sqcap \mathsf{C}_2$, $\mathsf{C}_6 \sqsubseteq \mathsf{C}_2$, $\mathsf{C}_7 \sqsubseteq \mathsf{C}_2$, and the concept D defined as follows.

$$
\begin{aligned}
\mathbf{D} \doteq\ & \exists_{\geq 2}\, \mathsf{R} \,.\, (\mathsf{C}_3 \sqcap \mathsf{P}_1) \sqcap \exists_{\geq 3}\, \mathsf{R} \,.\, (\mathsf{C}_4 \sqcap \mathsf{P}_1) \sqcap \exists_{\geq 1}\, \mathsf{R} \,.\, (\mathsf{C}_5 \sqcap \neg \mathsf{P}_1 \sqcap \mathsf{P}_2 \sqcap \mathsf{P}_3 \sqcap \neg \mathsf{P}_4) \sqcap \\
& \exists_{\geq 1}\, \mathsf{R} \,.\, (\mathsf{C}_5 \sqcap \neg \mathsf{P}_1 \sqcap \mathsf{P}_2 \sqcap \neg \mathsf{P}_3 \sqcap \neg \mathsf{P}_4) \sqcap \exists_{\geq 1}\, \mathsf{R} \,.\, (\mathsf{C}_5 \sqcap \neg \mathsf{P}_1 \sqcap \neg \mathsf{P}_2 \sqcap \neg \mathsf{P}_4) \sqcap \\
& \exists_{\geq 3}\, \mathsf{R} \,.\, (\mathsf{C}_6 \sqcap \mathsf{P}_1) \sqcap \exists_{\geq 2}\, \mathsf{R} \,.\, (\mathsf{C}_7 \sqcap \neg \mathsf{P}_1)
\end{aligned}
$$

Then, the concept term $\mathsf{D} \sqcap \exists_{\leq 7}\, \mathsf{R} \,.\, \mathsf{C}_1 \sqcap \exists_{\leq 7}\, \mathsf{R} \,.\, \mathsf{C}_2 \sqcap \exists_{\leq 7}\, \mathsf{R} \,.\, (\mathsf{C}_1 \sqcup \mathsf{C}_2)$ is *satisfiable* while the concept term $\mathsf{D} \sqcap (\exists_{\leq 6}\, \mathsf{R} \,.\, \mathsf{C}_1 \sqcup \exists_{\leq 6}\, \mathsf{R} \,.\, \mathsf{C}_2 \sqcup \exists_{\leq 6}\, \mathsf{R} \,.\, (\mathsf{C}_1 \sqcup \mathsf{C}_2))$ is *not satisfiable*. Using the signature calculus, RACER cannot compute the (un)satisfiability of either concept term within a reasonable amount of time (e.g. ≤ 100 seconds). In the next sections we present a new architecture which can compute the (un)satisfiability of these kinds of concept terms in almost constant time even if the values of the numbers occurring in the qualified number restrictions of D are appropriately increased to values around 1000.

2 Qualified Number Restrictions as Sets of Inequations

The inefficiency of tableaux algorithms for deciding the satisfiability of the above-mentioned concept terms is caused by their negligence of sets of inequations over set cardinalities induced by qualified number restrictions. A solution for this problem is presented in [7] where reasoning about sets of inequations is proposed. However, this approach is not based on a tableaux calculus but on atomic decomposition techniques. The contribution of our paper is two-fold. (1) We present a hybrid architecture which decides concept consistency for $\mathcal{ALCQH}_{R^+}$ by combining a tableaux calculus with a reasoner about sets of linear (in)equations. The architecture is inspired by [7] and [3]. (2) Furthermore, we integrated this architecture into the RACER system and present a first empirical evaluation which indicates a dramatic performance gain compared to other known approaches.

Syntax	Semantics
Concepts	
A	$\mathsf{A}^\mathcal{I} \subseteq \Delta^\mathcal{I}$, A is a concept name
$\neg\mathsf{C}$	$\Delta^\mathcal{I} \setminus \mathsf{C}^\mathcal{I}$
$\mathsf{C} \sqcap \mathsf{D}$	$\mathsf{C}^\mathcal{I} \cap \mathsf{D}^\mathcal{I}$
$\mathsf{C} \sqcup \mathsf{D}$	$\mathsf{C}^\mathcal{I} \cup \mathsf{D}^\mathcal{I}$
$\exists\, \mathsf{R} \,.\, \mathsf{C}$	$\{a \in \Delta^\mathcal{I} \mid \exists\, b \in \Delta^\mathcal{I} : (a,b) \in \mathsf{R}^\mathcal{I},\, b \in \mathsf{C}^\mathcal{I}\}$
$\forall\, \mathsf{R} \,.\, \mathsf{C}$	$\{a \in \Delta^\mathcal{I} \mid \forall\, b : (a,b) \in \mathsf{R}^\mathcal{I} \Rightarrow b \in \mathsf{C}^\mathcal{I}\}$
$\exists_{\geq n}\, \mathsf{S} \,.\, \mathsf{C}$	$\{a \in \Delta^\mathcal{I} \mid \mathsf{S}^\sharp(a, \mathsf{C}) \geq n\}$
$\exists_{\leq m}\, \mathsf{S} \,.\, \mathsf{C}$	$\{a \in \Delta^\mathcal{I} \mid \mathsf{S}^\sharp(a, \mathsf{C}) \leq m\}$
Roles	
R	$\mathsf{R}^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$

Terminol. Axioms	
Syntax	Satisfied if
$\mathsf{R} \in T$	$\mathsf{R}^\mathcal{I} = (\mathsf{R}^\mathcal{I})^+$
$\mathsf{R} \sqsubseteq \mathsf{S}$	$\mathsf{R}^\mathcal{I} \subseteq \mathsf{S}^\mathcal{I}$
$\mathsf{C} \sqsubseteq \mathsf{D}$	$\mathsf{C}^\mathcal{I} \subseteq \mathsf{D}^\mathcal{I}$

Figure 1: Syntax and Semantics of $\mathcal{ALCQH}_{R^+}$ ($n, m \in \mathbb{N}$, $n > 0$, $\|\cdot\|$ denotes set cardinality, $\mathsf{S} \in S$, and $\mathsf{S}^\sharp(a, \mathsf{C}) = \|\{b \in \Delta^\mathcal{I} \mid (a,b) \in \mathsf{S}^\mathcal{I},\, b \in \mathsf{C}^\mathcal{I}\}\|$).

2.1 The Language $\mathcal{ALCQH}_{R^+}$

We briefly introduce the description logic (DL) $\mathcal{ALCQH}_{R^+}$ (see the tables in Figure 1) using a standard Tarski-style semantics based on an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$. $\mathcal{ALCQH}_{R^+}$ extends the description logic $\mathcal{ALCNH}_{R^+}$ [2] by qualified number restrictions. The concept name $\top$ ($\bot$) is used as an abbreviation for $\mathsf{C} \sqcup \neg\mathsf{C}$ ($\mathsf{C} \sqcap \neg\mathsf{C}$). We assume a set of concept names C and a set of role names R. The mutually disjoint subsets P and T of R denote non-transitive and transitive roles, respectively ($R = P \cup T$).

If $\mathsf{R}, \mathsf{S} \in R$ are role names, then the terminological axiom $\mathsf{R} \sqsubseteq \mathsf{S}$ is called a *role inclusion axiom*. A *role hierarchy* $\mathcal{R}$ is a finite set of role inclusion axioms. Then, we define $\sqsubseteq^*$ as the reflexive transitive closure of $\sqsubseteq$ over such a role hierarchy $\mathcal{R}$. Given $\sqsubseteq^*$, the set of roles $\mathsf{R}^\downarrow = \{\mathsf{S} \in R \mid \mathsf{S} \sqsubseteq^* \mathsf{R}\}$ defines the *descendants* of a role R. $\mathsf{R}^\uparrow = \{\mathsf{S} \in R \mid \mathsf{R} \sqsubseteq^* \mathsf{S}\}$ is the set of *ancestors* of a role R. We also define the set $S = \{\mathsf{R} \in P \mid \mathsf{R}^\downarrow \cap T = \emptyset\}$ of *simple* roles that are neither transitive nor have a transitive role as descendant. A syntactic restriction holds for the combinability of number restrictions and transitive roles in $\mathcal{ALCQH}_{R^+}$. Number restrictions are only allowed for *simple* roles. This restriction is motivated by an undecidability result in case of an unrestricted combinability [6].

If C and D are concept terms, then $\mathsf{C} \sqsubseteq \mathsf{D}$ (*generalized concept inclusion* or *GCI*) is a terminological axiom. A finite set of terminological axioms $\mathcal{T}_\mathcal{R}$ is called a *terminology* or *TBox* w.r.t. to a given role hierarchy $\mathcal{R}$.[2]

The *concept satisfiability problem* is to decide whether a given concept term C is *satisfiable* w.r.t. to $\mathcal{T}$ and $\mathcal{R}$, i.e. whether there exists an interpretation $\mathcal{I}$ such that $\mathcal{I}$ satisfies $\mathcal{T}$ and $\mathcal{R}$ and $\mathsf{C}^\mathcal{I} \neq \emptyset$.

[2]The reference to $\mathcal{R}$ is omitted in the following.

2.2 A Tableaux Calculus for $\mathcal{ALCQH}_{R^+}$

In the following we present an ABox *tableaux algorithm* which decides the satisfiability of $\mathcal{ALCQH}_{R^+}$ concepts.

First, we introduce ABox assertions used as input for the tableaux algorithm. Let C be a concept term, R be a role name, O be the set of individual names, $\mathsf{a}, \mathsf{b} \in O$ be individual names, and $x \notin O$, then the following expression are ABox assertions: (1) $\mathsf{a}:\mathsf{C}$ (*instance assertion*), (2) $(\mathsf{a}, \mathsf{b}):\mathsf{R}$ (*role assertion*), and (3) $\forall x \,.\, (x:\mathsf{C})$ (*universal concept assertion*). An interpretation $\mathcal{I}$ satisfies an assertional axiom $\mathsf{a}:\mathsf{C}$ iff $\mathsf{a}^\mathcal{I} \in \mathsf{C}^\mathcal{I}$, $(\mathsf{a}, \mathsf{b}):\mathsf{R}$ iff $(\mathsf{a}^\mathcal{I}, \mathsf{b}^\mathcal{I}) \in \mathsf{R}^\mathcal{I}$, and $\forall x \,.\, (x:\mathsf{C})$ iff $\mathsf{C}^\mathcal{I} = \Delta^\mathcal{I}$. An ABox $\mathcal{A}$ is *consistent* iff there exists an interpretation $\mathcal{I}$ which satisfies all assertions in $\mathcal{A}$ and all axioms in $\mathcal{T}$ w.r.t. $\mathcal{R}$.

Given a TBox $\mathcal{T}$, and a role hierarchy $\mathcal{R}$, a concept D is satisfiable iff the ABox $\mathcal{A} = \{\mathsf{a}:\mathsf{D}\}$ extended in accordance to the following rules is consistent. For every GCI $\mathsf{C} \sqsubseteq \mathsf{D}$ in $\mathcal{T}$ the assertion $\forall x \,.\, (x:(\neg\mathsf{C} \sqcup \mathsf{D}))$ is added to $\mathcal{A}$. Every concept term occurring in $\mathcal{A}$ is transformed into its usual negation normal form. Every concept of the form $\exists\, \mathsf{R} \,.\, \mathsf{C}$ occurring in $\mathcal{A}$ is replaced by $(\exists_{\geq 1}\, \mathsf{R}' \sqcap \forall\, \mathsf{R}' \,.\, \mathsf{C})$ and every $\exists_{\geq n}\, \mathsf{R} \,.\, \mathsf{C}$ by $(\exists_{\geq n}\, \mathsf{R}' \sqcap \forall\, \mathsf{R}' \,.\, \mathsf{C})$, with $\mathsf{R}' \in R$ fresh in $\mathcal{A}$ and $\mathsf{R}' \sqsubseteq \mathsf{R} \in \mathcal{R}$.

$\mathcal{ALCQH}_{R^+}$ supports transitive roles and GCIs. Thus, in order to guarantee the termination of the tableaux calculus, the notion of *blocking* an individual for the applicability of tableaux rules is introduced as follows. Given an ABox $\mathcal{A}$ and an individual a occurring in $\mathcal{A}$, we define the *concept set* of a as $\sigma(\mathcal{A}, \mathsf{a}) := \{\top\} \cup \{\mathsf{C} \mid \mathsf{a}:\mathsf{C} \in \mathcal{A}\}$. We define an *individual ordering* '$\prec$' for individuals (elements of O) occurring in an ABox $\mathcal{A}$. If $\mathsf{b} \in O$ is introduced into $\mathcal{A}$, then $\mathsf{a} \prec \mathsf{b}$ for all individuals a already present in $\mathcal{A}$. Let $\mathcal{A}$ be an ABox and $\mathsf{a}, \mathsf{b} \in O$ be individuals in $\mathcal{A}$. We call a the blocking individual of b if all of the following conditions hold: (1) $\sigma(\mathcal{A}, \mathsf{a}) \supseteq \sigma(\mathcal{A}, \mathsf{b})$, (2) $\mathsf{a} \prec \mathsf{b}$. If there exists a blocking individual a for b, then b is said to be *blocked* (by a).

Given an ABox $\mathcal{A}$, $\sharp(\mathsf{a}, \mathsf{R})_\mathcal{A}$ defines the number of potential R-successors for an individual a mentioned in $\mathcal{A}$.

$$\sharp(\mathsf{a}, \mathsf{R})_\mathcal{A} = \sum_{\alpha \in \mathcal{A}} \mathrm{count}(\mathsf{a}, \mathsf{R}, \alpha)_\mathcal{A}, \;\; \mathrm{count}(\mathsf{a}, \mathsf{R}, \alpha)_\mathcal{A} = \begin{cases} n & \text{if } \alpha = \mathsf{a}:\exists_{\geq n}\, \mathsf{R}',\ \mathsf{R}' \in \mathsf{R}^\uparrow \\ 0 & \text{otherwise.} \end{cases}$$

Given an ABox $\mathcal{A}$, $\min(\mathsf{a}, \mathsf{R})_\mathcal{A}$ defines the minimal number of required and $\max(\mathsf{a}, \mathsf{R}, \mathsf{C})_\mathcal{A}$ the maximal number of allowed R-successors for an individual a mentioned in $\mathcal{A}$ (whose R-successors satisfy C).

$$\min(\mathsf{a}, \mathsf{R})_\mathcal{A} = \max(\{0\} \cup \{n \mid \mathsf{a}:\exists_{\geq n}\, \mathsf{S} \in \mathcal{A},\ \mathsf{S} \in \mathsf{R}^\downarrow\}$$
$$\max(\mathsf{a}, \mathsf{R}, \mathsf{C})_\mathcal{A} = \min(\{\infty\} \cup \{n \mid \mathsf{a}:\exists_{\leq n}\, \mathsf{S} \,.\, \mathsf{C} \in \mathcal{A},\ \mathsf{S} \in \mathsf{R}^\uparrow\})$$

We are now ready to define the *completion rules* that are intended to generate a so-called completion (see also below) of an ABox $\mathcal{A}$ w.r.t. a TBox $\mathcal{T}$.

R⊓ The conjunction rule.
  if $\mathsf{a}:\mathsf{C}\sqcap\mathsf{D}\in\mathcal{A}$, and $\{\mathsf{a}:\mathsf{C},\ \mathsf{a}:\mathsf{D}\}\not\subseteq\mathcal{A}$
  then $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{a}:\mathsf{C},\ \mathsf{a}:\mathsf{D}\}$

R⊔ The disjunction rule.
  if $\mathsf{a}:\mathsf{C}\sqcup\mathsf{D}\in\mathcal{A}$, and $\{\mathsf{a}:\mathsf{C},\ \mathsf{a}:\mathsf{D}\}\cap\mathcal{A}=\emptyset$
  then $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{a}:\mathsf{C}\}$ **or** $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{a}:\mathsf{D}\}$

R∀C The role value restriction rule.
  if $\mathsf{a}:\forall\,\mathsf{R}\,.\,\mathsf{C}\in\mathcal{A}$, and $\exists\,\mathsf{b}\in O,\mathsf{S}\in\mathsf{R}^{\downarrow}:(\mathsf{a},\mathsf{b}):\mathsf{S}\in\mathcal{A}$, and $\mathsf{b}:\mathsf{C}\notin\mathcal{A}$
  then $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{b}:\mathsf{C}\}$

R$\forall_+$C The transitive role value restriction rule.
  if 1. $\mathsf{a}:\forall\,\mathsf{R}\,.\,\mathsf{C}\in\mathcal{A}$, and $\exists\,\mathsf{b}\in O$, $\mathsf{T}\in\mathsf{R}^{\downarrow}$, $\mathsf{T}\in T$, $\mathsf{S}\in\mathsf{T}^{\downarrow}:(\mathsf{a},\mathsf{b}):\mathsf{S}\in\mathcal{A}$, and
  2. $\mathsf{b}:\forall\,\mathsf{T}\,.\,\mathsf{C}\notin\mathcal{A}$
  then $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{b}:\forall\,\mathsf{T}\,.\,\mathsf{C}\}$

R$\forall_x$ The universal concept restriction rule.
  if $\forall\,x\,.\,(x:\mathsf{C})\in\mathcal{A}$, and $\exists\,\mathsf{a}\in O$: a mentioned in $\mathcal{A}$, and $\mathsf{a}:\mathsf{C}\notin\mathcal{A}$
  then $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{a}:\mathsf{C}\}$

R$\exists_{\geq n}$ The number restriction exists rule.
  if 1. $\mathsf{a}:\exists_{\geq n}\,\mathsf{R}\in\mathcal{A}$, and a is not blocked, and
  2. $\neg\exists\,\mathsf{b}_1,\ldots,\mathsf{b}_\mathsf{n}\in O$, $\mathsf{S}_1,\ldots,\mathsf{S}_\mathsf{n}\in\mathsf{R}^{\downarrow}:\{(\mathsf{a},\mathsf{b}_\mathsf{k}):\mathsf{S}_\mathsf{k}\mid\mathsf{k}\in 1..\mathsf{n}\}\subseteq\mathcal{A}$
  then $\mathcal{A}'=\mathcal{A}\cup\{(\mathsf{a},\mathsf{b}_\mathsf{k}):\mathsf{R}\mid\mathsf{k}\in 1..\mathsf{n}\}$ where $\mathsf{b}_1,\ldots,\mathsf{b}_\mathsf{n}\in O$ are not used in $\mathcal{A}$

Merge The qualified number restriction merge rule.
  if 1. $\exists\,\mathsf{a}$, C mentioned in $\mathcal{A}:\sharp(\mathsf{a},\mathsf{R})_{\mathcal{A}}>\max(\mathsf{a},\mathsf{R},\mathsf{C})_{\mathcal{A}}$, and
  2. $\hat{R}=\ \{\mathsf{R}'\in P\mid\mathsf{a}:\exists_{\leq m}\,\mathsf{R}'\,.\,\mathsf{D}\in\mathcal{A}\}$, and
  3. $\mathcal{M}^{\mathsf{R}}_{\geq}=\{\mathsf{a}:\exists_{\geq n}\,\mathsf{S}\in\mathcal{A}\mid\mathsf{S}\in\mathsf{R}'^{\downarrow},\ \mathsf{R}'\in\hat{R}\}$, $\mathcal{M}^{\mathsf{R}}_{\leq}=\{\mathsf{a}:\exists_{\leq m}\,\mathsf{S}\,.\,\mathsf{D}\in\mathcal{A}\mid\mathsf{S}\in\hat{R}\}$
  then $\langle\text{SAT},M\rangle\leftarrow\text{inequations\_satisfiable}(\mathcal{M}^{\mathsf{R}}_{\geq},\mathcal{M}^{\mathsf{R}}_{\leq},\mathcal{A})$
  if SAT
  then $\mathcal{A}'=(\mathcal{A}\setminus\mathcal{M}^{\mathsf{R}}_{\geq})\cup M$ (*add transformed assertions*)
  else $\mathcal{A}'=\mathcal{A}\cup\{\mathsf{a}:\bot\}$

Observe the completion strategy defined below. The qualified number restriction merge rule needs some explanation. It is invoked whenever there exists an individual with potential successors for a role R such that the number of these successors violates an at-most restriction for an ancestor role of R. If this is the case, the rule calls the algebraic reasoner with the set $\mathcal{M}^{\mathsf{R}}_{\geq}$ of at-least and the set $\mathcal{M}^{\mathsf{R}}_{\leq}$ of at-most assertions. If the inequations derived from both sets are satisfiable, the algebraic reasoner returns a set M of new assertions such that these assertions satisfy all restrictions from $\mathcal{M}^{\mathsf{R}}_{\geq}$ and $\mathcal{M}^{\mathsf{R}}_{\leq}$ and, thus, the assertions from M may replace the ones from $\mathcal{M}^{\mathsf{R}}_{\geq}$. The replacement is required in order to guarantee the termination of the calculus. If the inequations are unsatisfiable, the ABox $\mathcal{A}'$ is marked as contradictory (see below).

Given an ABox $\mathcal{A}$, more than one rule might be applicable to $\mathcal{A}$. The order is determined by the *completion strategy* which is defined as follows. A *meta rule* controls the priority between individuals: Apply a tableaux rule to an individual $\mathsf{b}\in O$ only if no rule is applicable to another individual $\mathsf{c}\in O$ such that $\mathsf{c}\prec\mathsf{b}$.

The completion rules are always applied in the following order: (1) All non-generating rules (R$\sqcap$, R$\sqcup$, R$\forall$C, R$\forall_+$C, R$\forall_x$); (2) Qualified number restriction merge rule; (3) Number restriction exists rule (R$\exists_{\geq n}$). In the following we always assume that the completion strategy is observed. It ensures that rules are applied to individuals w.r.t. the ordering '$\prec$' and the number restriction exists rule is only applied to individuals if the qualified number restriction merge rule is not applicable.

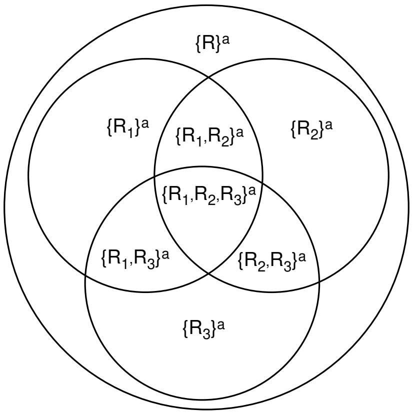
We assume the same naming conventions as used above. An ABox $\mathcal{A}$ is called *contradictory* if the following *clash trigger* is applicable. If the clash trigger is not applicable to $\mathcal{A}$, then $\mathcal{A}$ is called *clash-free*. The clash trigger has to deal with so-called primitive clashes: $\mathsf{a}:\bot \in \mathcal{A}$ or $\{\mathsf{a}:\mathsf{A}, \mathsf{a}:\neg\mathsf{A}\} \subseteq \mathcal{A}$, where A is a concept name. Any ABox containing a clash is obviously unsatisfiable. A clash-free ABox $\mathcal{A}$ is called *complete* if no completion rule is applicable to $\mathcal{A}$. A complete ABox $\mathcal{A}'$ derived from an ABox $\mathcal{A}$ is called a *completion* of $\mathcal{A}$. The purpose of the calculus is to generate a completion for an ABox $\mathcal{A}$ in order to prove the consistency of $\mathcal{A}$. For a given ABox $\mathcal{A}$, the calculus applies the completion rules. It stops the application of rules, if a clash occurs. The calculus answers "*yes*" if a completion can be derived, and "*no*" otherwise.

2.3 The Algebraic Reasoner

The algebraic reasoner has to determine whether the assertions contained in $\mathcal{M}^{\mathsf{R}}_{\leq} \cup \mathcal{M}^{\mathsf{R}}_{\geq}$ are satisfiable. This is achieved by a derivation process which is inspired by the approach presented in [7]. A set of (in)equations over set cardinalities is derived from the set $\mathcal{M}^{\mathsf{R}}_{\leq} \cup \mathcal{M}^{\mathsf{R}}_{\geq}$. These (in)equations can be mapped to a set of linear inequations where set cardinalities are represented as "variables" of the inequations. The satisfiability of such a set of linear inequations is decided with the help of a Simplex procedure which allows only solutions in $\mathbb{N}$. This Simplex procedure has been implemented in accordance with [1] using sparse arrays as basic data structures.

In the following we illustrate the reasoning with a simple example. Let R_1, R_2, R_3, and R be role names with $\mathsf{R}_1 \sqsubseteq \mathsf{R}$, $\mathsf{R}_2 \sqsubseteq \mathsf{R}$, $\mathsf{R}_3 \sqsubseteq \mathsf{R}$, and C be an atomic concept. As an example, we assume that the satisfiability of the following concept $\exists_{\leq 3}\,\mathsf{R} \sqcap \exists_{\geq 2}\,\mathsf{R}_1 \sqcap \exists_{\geq 2}\,\mathsf{R}_2 \sqcap \exists_{\geq 2}\,\mathsf{R}_3 \sqcap \forall\,\mathsf{R}_2\,.\,\mathsf{C} \sqcap \forall\,\mathsf{R}_3\,.\,\neg\mathsf{C}$ has to be checked. According to the rules described in the previous sections, the algebraic reasoner will be called with the sets $\mathcal{M}^{\mathsf{R}}_{\geq} = \{\mathsf{a}:\exists_{\geq 2}\,\mathsf{R}_1, \mathsf{a}:\exists_{\geq 2}\,\mathsf{R}_2, \mathsf{a}:\exists_{\geq 2}\,\mathsf{R}_3\}$ and $\mathcal{M}^{\mathsf{R}}_{\leq} = \{\mathsf{a}:\exists_{\leq 3}\,\mathsf{R}\}$, and the ABox $\mathcal{A} = \mathcal{M}^{\mathsf{R}}_{\geq} \cup \mathcal{M}^{\mathsf{R}}_{\leq} \cup \{\mathsf{a}:\forall\,\mathsf{R}_2\,.\,\mathsf{C}, \mathsf{a}:\forall\,\mathsf{R}_3\,.\,\neg\mathsf{C}, \dots\}$.

In order to better understand the following derivation process one has to keep in mind that the sets of successors for $\mathsf{a}^{\mathcal{I}}$ w.r.t. the roles R_1, R_2, R_3, and R are split in disjoint subsets as illustrated in Figure 2a. Let us assume that $\mathsf{R}^{\mathsf{a}} = \{b \in \Delta^{\mathcal{I}} \mid (\mathsf{a}^{\mathcal{I}}, b) \in \mathsf{R}^{\mathcal{I}}\}$ denotes the R-successors of $\mathsf{a}^{\mathcal{I}}$. For a non-empty subset $RS \subseteq R$ we define $RS^{\mathsf{a}} = \{\mathsf{b} \in \Delta^{\mathcal{I}} \mid (\mathsf{a}^{\mathcal{I}}, \mathsf{b}) \in \mathsf{R}'^{\mathcal{I}},\ \mathsf{R}' \in RS,\ (\mathsf{a}^{\mathcal{I}}, \mathsf{b}) \notin \mathsf{R}''^{\mathcal{I}},$ $\mathsf{R}'' \in (R \setminus RS)\}$. Now we can represent the partitioning of the sets R^{a}, $\mathsf{R}^{\mathsf{a}}_1$, $\mathsf{R}^{\mathsf{a}}_2$,



(a) Venn diagram illustrating the partitioning of $\mathsf{R^a}$.

Partitioning:

$\mathsf{R_1^a} = \{\mathsf{R_1}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a}$
$\mathsf{R_2^a} = \{\mathsf{R_2}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2}\}^\mathsf{a} \cup \{\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a}$
$\mathsf{R_3^a} = \{\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a}$
$\mathsf{R^a} = \{\mathsf{R}\}^\mathsf{a} \cup \{\mathsf{R_1}\}^\mathsf{a} \cup \{\mathsf{R_2}\}^\mathsf{a} \cup \{\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a} \cup \{\mathsf{R_1},\mathsf{R_2},\mathsf{R_3}\}^\mathsf{a}$

Contents of $\mathcal{S}$:

$\{\|\mathsf{R^a}\| \geq 0, \|\mathsf{R_1^a}\| \geq 2, \|\mathsf{R_2^a}\| \geq 2, \|\mathsf{R_3^a}\| \geq 2, \|\mathsf{R^a}\| \leq 3,$
$\|\mathsf{R_1^a}\| = \mathsf{r_1} + \mathsf{r_1r_2} + \mathsf{r_1r_3} + \mathsf{r_1r_2r_3},$
$\|\mathsf{R_2^a}\| = \mathsf{r_2} + \mathsf{r_1r_2} + \mathsf{r_2r_3} + \mathsf{r_1r_2r_3},$
$\|\mathsf{R_3^a}\| = \mathsf{r_3} + \mathsf{r_1r_3} + \mathsf{r_2r_3} + \mathsf{r_1r_2r_3},$
$\|\mathsf{R^a}\| = \mathsf{r} + \mathsf{r_1} + \mathsf{r_2} + \mathsf{r_3} + \mathsf{r_1r_2} + \mathsf{r_1r_3} + \mathsf{r_2r_3} + \mathsf{r_1r_2r_3}\}$

(b) Partitions according to left figure and (in)equations derived from partitions and number restrictions.

Figure 2: Partitioning of $\mathsf{R^a}$ and derivation of initial (in)equations.

and $\mathsf{R_3^a}$ as shown in Figure 2b. For better readability we denote $\|RS^\mathsf{a}\|$ by $\mathsf{r_{i_1}} \ldots \mathsf{r_{i_k}}$ if $RS = \{\mathsf{R_{i_1}}, \ldots, \mathsf{R_{i_k}}\}$. The set $\mathcal{S}$ of linear (in)equations initially contains $\|\mathsf{S^a}\| \geq 0$ for every role S mentioned in $\mathcal{M}_\leq^\mathsf{R} \cup \mathcal{M}_\geq^\mathsf{R}$, $\|\mathsf{S^a}\| \geq \mathsf{n}$ for every $\mathsf{a}\!:\!\exists_{\geq n}\,\mathsf{S} \in \mathcal{M}_\geq^\mathsf{R}$, and $\|\mathsf{S^a}\| \leq \mathsf{m}$ for every $\mathsf{a}\!:\!\exists_{\leq m}\,\mathsf{S} \in \mathcal{M}_\leq^\mathsf{R}$.[3] For our example the initial contents of $\mathcal{S}$ is displayed in Figure 2b.

The algebraic reasoner has to verify whether a set RS^a has to be empty, i.e. $\|RS^\mathsf{a}\| = 0$. It uses a role successor satisfiability test $\mathrm{RSAT}(\mathcal{A}, \mathsf{a}, \mathsf{C}, RS)$ where $\mathcal{A}$ is an ABox, a an individual, C a concept term, and RS a role set. The test $\mathrm{RSAT}(\mathcal{A}, \mathsf{a}, \mathsf{C}, RS)$ is successful, i.e. $RS^\mathsf{a} \neq \emptyset$, iff the ABox $\mathcal{A}' = \mathcal{A} \cup \{\mathsf{a}\!:\!\mathsf{C}\} \cup \{(\mathsf{a},\mathsf{b})\!:\!\mathsf{S} \mid \mathsf{S} \in RS\}$ with b new in $\mathcal{A}$ is consistent.

For our example, the reasoner iteratively has to test all non-empty subsets RS of $\{\mathsf{R}, \mathsf{R_1}, \mathsf{R_2}, \mathsf{R_3}\}$: If $\neg\mathrm{RSAT}(\mathcal{A} \setminus \mathcal{M}_\geq^\mathsf{R}, \mathsf{a}, \top, RS)$ holds, the equations $\|RS'^\mathsf{a}\| = 0$ are added to $\mathcal{S}$ for every set RS' with $\{\mathsf{R}, \mathsf{R_1}, \mathsf{R_2}, \mathsf{R_3}\} \supseteq RS' \supseteq RS$. Since $\mathcal{A}$ contains $\mathsf{a}\!:\!\forall\,\mathsf{R_2}\,.\,\mathsf{C}$ and $\mathsf{a}\!:\!\forall\,\mathsf{R_3}\,.\,\neg\mathsf{C}$ there cannot exist a $\{\mathsf{R_2}, \mathsf{R_3}\}$-successor of a. Thus, the equations $\|\{\mathsf{R_2}, \mathsf{R_3}\}^\mathsf{a}\| = 0$ and $\|\{\mathsf{R_1}, \mathsf{R_2}, \mathsf{R_3}\}^\mathsf{a}\| = 0$ have to be added to $\mathcal{S}$. This leads to an unsatisfiable set $\mathcal{S}$ of (in)equations:

$$\|\mathsf{R^a}\| \leq 3$$
$$\|\mathsf{R^a}\| = \mathsf{r} + \mathsf{r_1} + \mathsf{r_2} + \mathsf{r_3} + \mathsf{r_1r_2} + \mathsf{r_1r_3} \geq (\mathsf{r_2} + \mathsf{r_1r_2}) + (\mathsf{r_3} + \mathsf{r_1r_3}) = \|\mathsf{R_2^a}\| + \|\mathsf{R_3^a}\| \geq 4$$

If the set $\mathcal{M}_\leq^\mathsf{R}$ contains assertions of the form $\mathsf{a}\!:\!\exists_{\leq m}\,\mathsf{R}\,.\,\mathsf{D}$, these are transformed by the algebraic reasoner into $\mathsf{a}\!:\!(\exists_{\leq m}\,\mathsf{R}' \sqcap \forall\,\mathsf{R}'\,.\,\mathsf{D} \sqcap \forall\,\mathsf{R}\backslash\mathsf{R}'\,.\,\neg\mathsf{D})$ where R' is fresh in $\mathcal{A}$ and $\mathsf{R}' \sqsubseteq \mathsf{R} \in \mathcal{R}$. The new operator $(\forall\,\mathsf{R}\backslash\mathsf{R}'\,.\,\mathsf{E})$ is based on set difference for roles (see [7] for details). Its semantics is defined as follows.

[3] Elements of the form $\mathsf{a}\!:\!\exists_{\leq m}\,\mathsf{S}\,.\,\mathsf{C}$ with $\mathsf{C}^\mathcal{I} \neq \Delta^\mathcal{I}$ are discussed below.

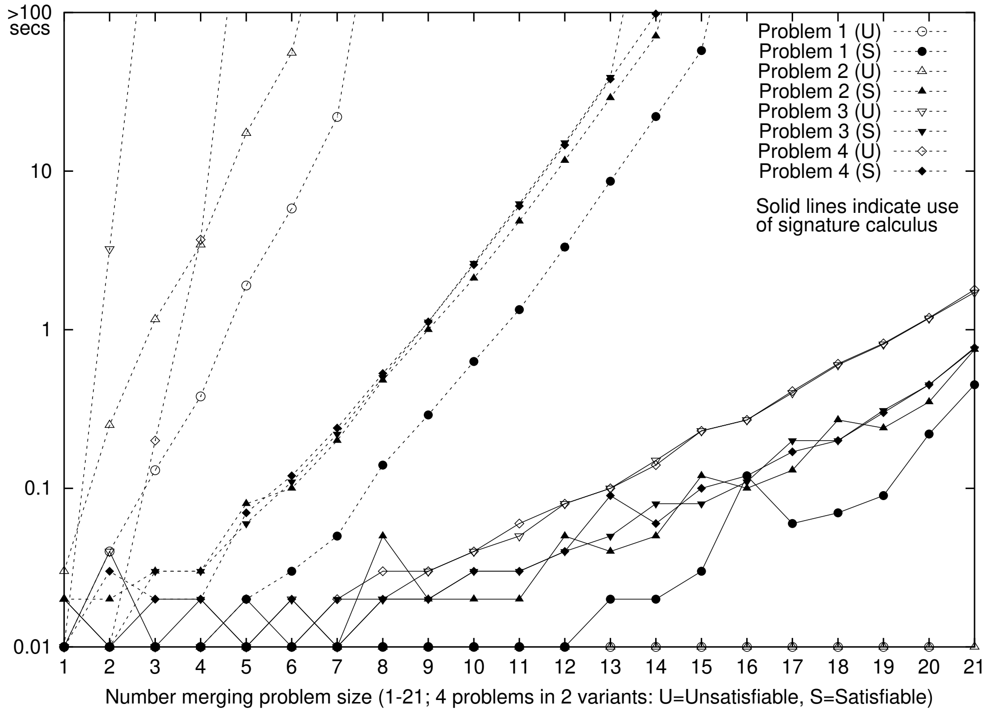


Figure 3: RACER: benchmark problems w/out signature calculus.

$(\forall\, \mathsf{R} \backslash \mathsf{R}' . \mathsf{E})^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \,|\, \forall\, b : (a, b) \in (\mathsf{R}^{\mathcal{I}} \backslash \mathsf{R}'^{\mathcal{I}}) \Rightarrow b \in \mathsf{E}^{\mathcal{I}}\}$

The algebraic reasoner implements this semantics as follows. Let us assume that assertions of the form $\mathsf{a} : \exists_{\leq m_1} \mathsf{S}_1 . \mathsf{D}_1, \ldots, \mathsf{a} : \exists_{\leq m_k} \mathsf{S}_\mathsf{k} . \mathsf{D}_\mathsf{k}$ with $\mathsf{k} \geq 1$ have to be handled. Then, the role successor satisfiability tests have the form $\mathrm{RSAT}(\mathcal{A}, \mathsf{a}, \mathsf{E}_1 \sqcap \ldots \sqcap \mathsf{E}_\mathsf{k}, \{\mathsf{S}_1, \ldots, \mathsf{S}_\mathsf{k}\})$ where E_i is nondeterministically chosen from $\{\mathsf{D}_\mathsf{i}, \sim\mathsf{D}_\mathsf{i}\}$, $\mathsf{i} \in 1..\mathsf{k}$ ($\sim\mathsf{D}$ denotes the negation normal form of $\neg\mathsf{D}$).

3 Evaluation

In order to indicate the advancement of this new architecture, we compare the performance of the hybrid architecture against settings where a standard tableaux calculus and the signature calculus is used. A set of four benchmark problems were generated. The increased difficulty of the problems is caused by exponentially increasing the size of numbers used in at-least and at-most concepts. Each of the four problems exists in two variants (a 'test concept' is consistent vs. inconsistent). A problem basically employs concept terms of the form $\exists_{\leq n} \mathsf{R} \sqcap \exists_{\geq m_1} \mathsf{R}_1 \sqcap \exists_{\geq m_2} \mathsf{R}_2 \sqcap \exists_{\geq m_3} \mathsf{R}_3 \sqcap \forall\, \mathsf{R}_2 . \mathsf{C} \sqcap \forall\, \mathsf{R}_3 . \neg\mathsf{C}$ with $\mathsf{R}_\mathsf{i} \sqsubseteq \mathsf{R}, \mathsf{i} \in 1..3$. The (in)consistency of these terms has to be proven. A term is made consistent by choosing values for n, m_i such that $\max(m_1, m_2 + m_3) \leq n$ or inconsistent if $\max(m_1, m_2 + m_3) > n$.

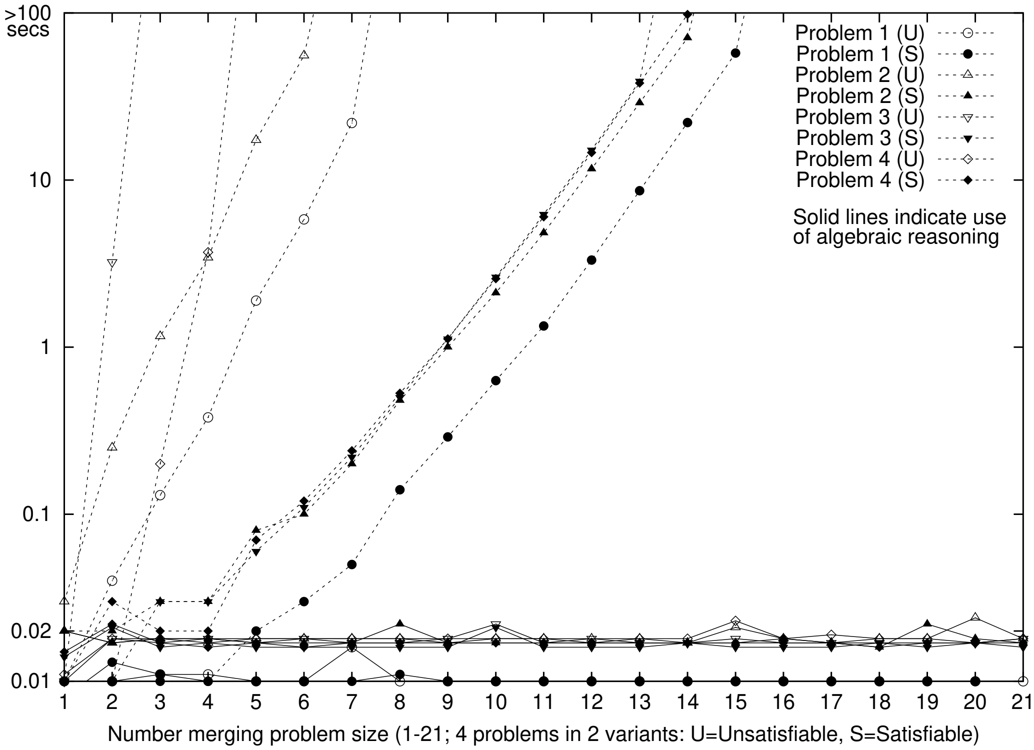


Figure 4: RACER: benchmark problems w/out algebraic reasoning.

Figure 3 demonstrates the result of this benchmark w/out the signature calculus and Figure 4 the result w/out algebraic reasoning. Although the performance gain in Figure 3 (signature calculus) is dramatic, the result in Figure 4 (algebraic reasoning) is even more dramatic since these problems can now be solved in constant time (usually below 0.02 seconds). The speed enhancement also scales up for problems with *qualified* number restrictions. The "intractable" problem from the introduction can now be handled as well. Using the algebraic reasoner it can be solved well below 0.1 seconds even if the values occurring in number restrictions are increased up to $\sim$1000.

However, there exist problems such that the number of required role successor satisfiability tests, and, in turn, the number of variables required for the Simplex procedure, might increase exponentially in the worst case. This can be illustrated with concepts of the form $\exists\, \mathsf{R} \,.\, \mathsf{C_1} \sqcap \ldots \sqcap \exists\, \mathsf{R} \,.\, \mathsf{C_n} \sqcap \exists_{\leq m}\, \mathsf{R}$, $m < n$. If such a concept is satisfiable, the algebraic reasoner has to consider $\mathcal{O}(2^n)$ variables for the Simplex procedure and $\mathcal{O}(2^n)$ role successor satisfiability tests in the worst case. Our experiments indicate that these concepts can often be dealt with by the signature calculus [3] quite efficiently. The is due to the fact that the signature calculus tries to construct a model "on the fly" and it might succeed rather quickly if the right strategy is chosen, e.g. to merge as many role successors as possible. However, we conjecture that a similar strategy can also be employed for the algebraic reasoner, e.g. by trying to intersect as many role successor sets as possible.

4 Conclusion and Outlook

In this paper we have presented a hybrid architecture for efficiently dealing with qualified number restrictions in the DL $\mathcal{ALCQH}_{R^+}$. The architecture has been implemented for concept satisfiability tests and evaluated in the ABox description logic system RACER. In contrast to [7] our approach is integrated into a tableaux calculus and can deal with GCIs, transitive roles, and cyclic terminologies. We are currently extending this approach to arbitrary ABoxes for the DL $\mathcal{ALCQHI}_{R^+}$ which extends $\mathcal{ALCQH}_{R^+}$ by inverse roles.

References

[1] R.E. Gomory. An algorithm for integer solutions to linear programs. In R.L. Graves and P. Wolfe, editors, *Recent Advances in Mathematical Programming*, pages 269–302. McGraw-Hill, Ney York, 1963.

[2] V. Haarslev and R. Möller. Expressive ABox reasoning with number restrictions, role hierarchies, and transitively closed roles. In A.G. Cohn, F. Giunchiglia, and B. Selman, editors, *Proceedings of Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'2000), Breckenridge, Colorado, USA, April 11-15, 2000*, pages 273–284, April 2000.

[3] V. Haarslev and R. Möller. Optimizing reasoning in description logics with qualified number restriction: Extended abstract. In D.L. McGuinness and P. Patel-Schneider, editors, *Proceedings of the International Workshop on Description Logics (DL'2001), Aug. 1-3, 2001, Stanford, CA, USA*, August 2001.

[4] V. Haarslev and R. Möller. RACER system description. In *Proceedings of the International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, 2001, Siena, Italy*, LNCS. Springer-Verlag, Berlin, June 2001.

[5] B. Hollunder and F. Baader. Qualifying number restrictions in concept languages. In J. Allen, R. Fikes, and E. Sandewall, editors, *Second International Conference on Principles of Knowledge Representation, Cambridge, Mass., April 22-25, 1991*, pages 335–346, April 1991.

[6] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In H. Ganzinger, D. McAllester, and A. Voronkov, editors, *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, number 1705 in LNAI, pages 161–180. Springer-Verlag, September 1999.

[7] H.J. Ohlbach and J. Köhler. Modal logics, description logics and arithmetic reasoning. *Artificial Intelligence*, 109(1-2):1–31, 1999.