Optimizing Algebraic Tableau Reasoning for $\mathcal{SHOQ}$: First Experimental Results

Jocelyne Faddoul and Volker Haarslev

Concordia University, Montreal, Canada
{j_faddou,haarslev}@cse.concordia.ca

Abstract. In this paper we outline an algebraic tableau algorithm for the DL $\mathcal{SHOQ}$, which supports more informed reasoning due to the use of semantic partitioning and integer programming. We introduce novel and adapt known optimization techniques and show their effectiveness on the basis of a prototype reasoner implementing the optimization techniques for the algebraic approach. Our first set of benchmarks clearly indicates the effectiveness of our approach and a comparison with the DL reasoners Pellet and HermiT demonstrates a runtime improvement of several orders of magnitude.

1 Motivation

Nominals play an important role in Description Logics (DLs) as they allow one to express the notion of identity and enumeration; nominals must be interpreted as singleton sets. An example for the use of nominals in $\mathcal{SHOQ}$ would be $\mathit{Eye\_Color} \equiv \mathit{Green} \sqcup \mathit{Blue} \sqcup \mathit{Brown} \sqcup \mathit{Black} \sqcup \mathit{Hazel}$ where each color is represented as a nominal. The cardinality of *Eye_Color* is restricted to have at most 5 instances, i.e., the above-mentioned nominals. Qualified cardinality restrictions (QCRs) allow one to specify lower ($\geq n R.C$) and upper ($\leq n R.C$) bounds on the number of elements related via a certain role with additionally specifying qualities on the related elements. Due to the interaction between nominals and QCRs the $\mathcal{SHOQ}$ concept $\geq 6\, \mathit{has\_color}.\mathit{Eye\_Color}$ is unsatisfiable. Each *nominal* must be interpreted as a set with the cardinality 1 (and thus can be used to enumerate domain elements), whereas an atomic concept is interpreted as a set with an unbounded cardinality. Moreover, the quasi-tree model property, which has always been advantageous for DL tableau methods, does not hold for $\mathcal{SHOQ}$.

Resolution-based reasoning procedures were proposed in [8] and were proven to be weak in dealing with QCRs containing large numbers. Hypertableaux [9] were recently studied to minimize non-determinism in DL reasoning with no special treatment for QCRs. These approaches and standard tableau techniques suffer from the low level of information about the cardinalities of concepts and the number of role successors implied by nominals and QCRs (e.g., see the example above) because these algorithms treat these cardinalities in a blind and uninformed way.

Our early work on performance improvements for reasoning with QCRs for the DL $\mathcal{SHQ}$ was based on a so-called signature calculus [5] and, alternatively, on algebraic reasoning [6] (not applicable to Aboxes). Our algebraic approach represents the knowledge about implied cardinalities as linear inequations. The advantages of such an approach have been demonstrated in [4] where an Abox calculus combining tableau and algebraic reasoning for $\mathcal{SHQ}$ is presented that dramatically improves the runtime performance for reasoning with QCRs. This paper extends this line of research [1, 3, 4] to $\mathcal{SHOQ}$. This calculus [2] is by no means a simple extension because (i) the quasi-tree model property is lost, (ii) QCRs cannot be dealt with locally anymore, and (iii) possible interactions between QCRs and nominals need to be considered globally.

2 The Description Logic $\mathcal{SHOQ}$

Let N_C, N_R be non-empty and disjoint sets of concept and role names respectively. Let $N_o \subseteq N_C$ be the set of nominals, and $N_{R^+} \subseteq N_R$ the set of transitive role names. An RBox $\mathcal{R}$ is a finite set of *role inclusion axioms* (RIAs) of the form $R \sqsubseteq S$, where R, S are role names in N_R. With $\sqsubseteq_*$ we denote the reflexive transitive closure of $\sqsubseteq$ on $\mathcal{R}$. A role name R is called *simple* if it is neither transitive nor has a transitive subrole. A TBox $\mathcal{T}$ is a finite set of *general concept inclusion axioms* (GCIs) of the form $C \sqsubseteq D$, where C, D are concepts, and $C \equiv D$ abbreviates $\{C \sqsubseteq D,\ D \sqsubseteq C\}$. The set of $\mathcal{SHOQ}$ concepts is the smallest set such that: (i) $A \in N_C$ is a concept, and (ii) if C, D are concepts, $R \in N_R$, and $S \in N_R$ is a simple role then $\neg C$, $(C \sqcup D)$, $(C \sqcap D)$, $(\exists R.C)$, $(\forall R.C)$, $(\geq nS.C)$, $(\leq nS.C)$ with $n \in \mathbb{N}$ are also concepts. We use $\top$ $(\bot)$ as an abbreviation for $A \sqcup \neg A$ $(A \sqcap \neg A)$ and $\geq nS$ $(\leq nS)$ for $\geq nS.\top$ $(\leq nS.\top)$. We do not consider descriptions of the form $\exists R.C$ as they can be converted to $\geq 1\, R.C$, without imposing the simple role restriction.

We assume a standard Tarski-style interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ such that $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ for $A \in N_C$, $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ for $R \in N_R$. Using # to denote the cardinality of a set, we define the set of *R-fillers* for a given role name R and an individual s as $FIL(R, s) = \{t \in \Delta^{\mathcal{I}} \mid \langle s, t\rangle \in R^{\mathcal{I}}\}$ and the set of all *R-fillers* as: $FIL(R) = \bigcup_{s \in \Delta^{\mathcal{I}}} FIL(R, s)$. The semantics of $\mathcal{SHOQ}$ concept descriptions is such that $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$, $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$, $(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$, $\#o^{\mathcal{I}} = 1$ for all $o \in N_o$, $(\forall R.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \langle s, t\rangle \in R^{\mathcal{I}} \Rightarrow t \in C^{\mathcal{I}}\}$, $(\exists R.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \exists t : \langle s, t\rangle \in R^{\mathcal{I}} \wedge t \in C^{\mathcal{I}}\}$, $(\geq nS.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \#(FIL(S, s) \cap C^{\mathcal{I}}) \geq n\}$, $(\leq nS.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \#(FIL(S, s) \cap C^{\mathcal{I}}) \leq n\}$.

Let KB($\mathcal{T}, \mathcal{R}$) denote a $\mathcal{SHOQ}$ knowledge base consisting of a TBox $\mathcal{T}$ and an RBox $\mathcal{R}$. The KB($\mathcal{T}, \mathcal{R}$) is said to be consistent iff there exists an interpretation $\mathcal{I}$ satisfying $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for each $C \sqsubseteq D \in \mathcal{T}$ and $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ for each $R \sqsubseteq S \in \mathcal{R}$. In this case, $\mathcal{I}$ is said to be a model of KB($\mathcal{T}, \mathcal{R}$). A concept C is said to be satisfiable w.r.t. KB($\mathcal{T}, \mathcal{R}$) iff $C^{\mathcal{I}} \neq \emptyset$. $\mathcal{I}$ is called a model of C w.r.t. $\mathcal{R}$ and $\mathcal{T}$. A $\mathcal{SHOQ}$ ABox $\mathcal{A}$ is a finite set of concept membership assertions of the form $a : C$ or role membership assertions of the form $(a, b) : R$ with a, b two individual names. An Abox $\mathcal{A}$ is said to be consistent w.r.t. KB($\mathcal{T}$, $\mathcal{R}$) if there exists a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{R}$ such that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ is satisfied for each $a : C$ in $\mathcal{A}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ for each $(a, b) : R$ in $\mathcal{A}$. Using nominals, concept satisfiability and ABox consistency can be reduced to KB consistency. Hence, without loss of generality we restrict our attention to KB consistency in the following.

We assume all concepts to be in their *negation normal form* (NNF). We use $\dot{\neg}C$ to denote the NNF of $\neg C$ and $nnf(C)$ to denote the NNF of C. When checking KB($\mathcal{T}, \mathcal{R}$) consistency, the concept axioms in $\mathcal{T}$ can be reduced to a single axiom $\top \sqsubseteq C_{\mathcal{T}}$ such that $C_{\mathcal{T}}$ abbreviates $\bigsqcap_{C \sqsubseteq D \in \mathcal{T}} nnf(\neg C \sqcup D)$. A TBox consistency test can be checked by testing the consistency of $o \sqsubseteq C_\mathcal{T}$ with $o \in N_\mathrm{o}$ new in $\mathcal{T}$, which means that at least $o^\mathcal{I} \in {C_\mathcal{T}}^\mathcal{I}$ and ${C_\mathcal{T}}^\mathcal{I} \neq \emptyset$. Moreover, since $\top^\mathcal{I} = \Delta^\mathcal{I}$ then every domain element must also satisfy $C_\mathcal{T}$ (every domain element is a member of $C_\mathcal{T}$).

3 Algebraic Tableau for $\mathcal{SHOQ}$

Given KB $(\mathcal{T}, \mathcal{R})$, such that we have $\top \sqsubseteq C_\mathcal{T}$, we apply a rewriting algorithm (see [2] for details) to $C_\mathcal{T}$ which returns $C'_\mathcal{T}$ and extends $\mathcal{R}$ with role inclusion axioms. This rewriting transforms all QCRs of the form $\geq nR.C$ or $\leq nR.C$, where C can be also equal to $\top$, into unqualified cardinality restrictions of the form $\geq nR'$ ($\leq nR'$) by using a new role-set difference operator ($\forall_\backslash$) and adding universal restrictions using newly introduced subroles ($R' \sqsubseteq R$). Roughly speaking, $\geq n\,R.C$ is transformed into $\geq n\,R' \sqcap \forall R'.C$ with adding $R' \sqsubseteq R$ to $\mathcal{R}$, and $\leq n\,R.C$ into $\leq n\,R' \sqcap \forall R'.C \sqcap \forall(R \backslash R').\dot{\neg}C$ with adding $R' \sqsubseteq R$ to $\mathcal{R}$. In both cases R' is always fresh in $\mathcal{R}$, and the transformation is satisfiability-preserving (see [2] for a proof and more details). The semantics of the role-set operator is defined such that $(\forall(R \backslash S).D)^\mathcal{I} = \{s \in \Delta^\mathcal{I} \mid \langle s,t\rangle \in R^\mathcal{I} \wedge \langle s,t\rangle \notin S^\mathcal{I} \Rightarrow t \in D^\mathcal{I}\}$.

3.1 Partitioning domain elements

The key technique and major difference between algebraic and standard tableau reasoning for $\mathcal{SHOQ}$ is the *atomic decomposition technique* [10] which is used to compute a partitioning of domain elements into disjoint subsets allowing numerical restrictions implied by QCRs and nominals to be encoded into sets of inequations.

Let $H(R)$ denote the set of role names for all subroles of $R \in N_\mathrm{R}$: $H(R) = \{R' \mid R' \sqsubseteq_* R\}$. For technical reasons we do not add R to $H(R)$ since R is a superrole for elements in $H(R)$ and R does not occur in number restrictions anymore after preprocessing. For every role $R' \in H(R)$, the set of R'*-fillers* forms a subset of the set of R*-fillers* ($FIL(R') \subseteq FIL(R)$). We define $\overline{R'}$ to be the complement of R' w.r.t. $H(R)$, the set of $\overline{R'}$*-fillers* is then defined as $\overline{R'}$*-fillers* $=(FIL(R) \setminus FIL(R'))$. Since we do not have $\geq nR$ or $\leq nR$ concept expressions using role complements, no role complement will be explicitly used. For ease of presentation, we do not list role complements.

Qualifications on Role fillers: The atomic decomposition must also consider when $FIL(R)$ intersects with the interpretation of a *qualifying concept*. A qualifying concept D is a concept used to impose a qualification, D, on the set of R-fillers for some role $R \in N_R$. Let $Q_\mathrm{C}(R) = \{D \mid \forall S.D$ occurs in $C_\mathcal{T}$ with $R \sqsubseteq_* S \in \mathcal{R}\}$ be the set of qualifying concepts for $R \in N_\mathrm{R}$. Since $D \in Q_\mathrm{C}(R)$ could be a complex expression or a nominal, and for ease of presentation, we assign a unique qualification name q for each $D \in Q_\mathrm{C}(R)$. Let $\mathcal{Q}_\mathrm{N}$ be the set of all qualification names assigned, and $\mathcal{Q}_\mathrm{C} = \bigcup_{R \in N_\mathrm{R}} Q_\mathrm{C}(R)$ be the set of qualifying concepts in $C_\mathcal{T}$. We maintain a mapping between qualification names and their corresponding concept expressions using a bijection $\theta : \mathcal{Q}_\mathrm{N} \rightarrow \mathcal{Q}_\mathrm{C}$; in case a nominal $o \in N_\mathrm{o}$ has been used as a qualifying concept expression then o is also used as the qualification name and $\theta(o) = o$. Let $Q_\mathrm{N}(R)$ denote the set of qualification names for a role ($R \in N_R$) then $Q_\mathrm{N}(R)$ is defined as $Q_\mathrm{N}(R) = \{q \in \mathcal{Q}_\mathrm{N} \mid \theta(q) \in Q_\mathrm{C}(R)\}$.

We define $\mathcal{Q}_{\mathrm{C}}^{\dot\neg} = \{\dot\neg D \,|\, D \in \mathcal{Q}_{\mathrm{C}}\}$ as the set of negated qualifying concepts in their NNF. A mapping $\dot\neg_{Q}$ is maintained between $\mathcal{Q}_{\mathrm{C}}$ and $\mathcal{Q}_{\mathrm{C}}^{\dot\neg}$ such that given a qualifying concept $D \in \mathcal{Q}_{\mathrm{C}}$, $\dot\neg_{Q}(D) = \dot\neg D$ with $\dot\neg D \in \mathcal{Q}_{\mathrm{C}}^{\dot\neg}$.

Interaction with Nominals: For each nominal $o \in N_{\mathrm{o}}$, $o^{\mathcal{I}}$ can interact with *R-fillers* for some R in N_{R} such that $(o^{\mathcal{I}} \subseteq FIL(R))$. Also the same nominal o can interact with *R-fillers* and *S-fillers* for $R, S \in N_{\mathrm{R}}$ such that R, S do not necessarily share subroles or superroles in $\mathcal{R}$. This means that *R-fillers* and *S-fillers* could interact with each other due to their common interaction with the same nominal o. These interactions lead to the following definitions.

Definition 1 (Decomposition Set). Given a role R we define the decomposition set for R-fillers as $\mathcal{D}_R = H(R) \cup Q_N(R) \cup N_{\mathrm{o}}$. $\mathcal{D}_R$ is a decomposition set since each subset P of $\mathcal{D}_R$ defines a unique set of nominals, roles, and/or qualification names that admits an interpretation $P^{\mathcal{I}} = \bigcap_{o \in P \cap N_{\mathrm{o}}} o^{\mathcal{I}} \cap \bigcap_{i \in N_{\mathrm{o}} \setminus P} \neg i^{\mathcal{I}} \cap \bigcap_{\mathrm{R}' \in P \cap H(R)} FIL(R') \cap \bigcap_{\mathrm{R}'' \in (H(R) \setminus P)} FIL(\overline{R''}) \cap \bigcap_{p \in P \cap Q_N(\mathcal{R})} \theta(p)^{\mathcal{I}} \cap \bigcap_{q \in (Q_N(R) \setminus P)} (\dot\neg\theta(q))^{\mathcal{I}}$. For all sets $P, Q \subseteq \mathcal{D}_R$ with $P \neq Q$, it holds by definition that $P^{\mathcal{I}} \neq Q^{\mathcal{I}}$. This makes all $P^{\mathcal{I}}$ with $P \subseteq \mathcal{D}_R$ disjoint with one another and the set of all P with $P \subseteq \mathcal{D}_R$ defines a partitioning of $\mathcal{D}_R$.

Definition 2 (Global Partitioning). Let $\mathcal{DS} = (\bigcup_{R \in N_{\mathrm{R}}} \mathcal{D}_R \cup N_{\mathrm{o}}) \setminus \{\dot\neg C \,|\, \{C, \dot\neg C\} \subseteq Q_{\mathrm{C}}\}$[1]. The set $\mathcal{P} = \{P \,|\, P \subseteq \mathcal{DS}\}$ defines a *global partitioning* of $\mathcal{DS}$ and $\mathcal{P}^{\mathcal{I}} = \Delta^{\mathcal{I}}$ because it includes all possible domain elements which correspond to a nominal and/or a role filler: $\mathcal{P}^{\mathcal{I}} = \bigcup_{P \subseteq \mathcal{DS}} P^{\mathcal{I}}$.

3.2 Encoding Numerical Restrictions into Inequations

Given $\mathcal{T}$ and a partitioning $\mathcal{P}$ for $\mathcal{DS}$, one can reduce the satisfiability of expressions of the form $(\geq nR)$ and $(\leq mR)$ and the satisfiability of the nominals semantics into inequation solving based on the following principles.

Mapping Cardinalities to Variables A variable name v is assigned for each partition name P such that v can be mapped to a non-negative integer value n using $\sigma : \mathcal{V} \to \mathbb{N}$ with $\sigma(v)$ denoting the cardinality of $P^{\mathcal{I}}$. Let $\mathcal{V}$ be the set of all variable names and $\alpha : \mathcal{V} \to \mathcal{P}$ be a one-to-one mapping between each partition name $P \in \mathcal{P}$ and a variable $v \in \mathcal{V}$ such that $\alpha(v) = P$, and if a non-negative integer n is assigned to v using σ then $\sigma(v) = n = \#P^{\mathcal{I}}$. Given $L \subseteq \mathcal{DS}$, let V_L denote the set of variable names mapped to partitions satisfying $L^{\mathcal{I}}$, V_L is defined as

$$V_L = \begin{pmatrix} \{v \in \mathcal{V} \,|\, p \in \alpha(v) \, \textit{for each} \; p \in (L \cap N_{\mathrm{R}})\} \cap \\ \{v \in \mathcal{V} \,|\, oq \in \alpha(v) \, \textit{for each} \; oq \in (L \cap (N_o \cup \mathcal{Q}_{\mathrm{N}}))\} \cap \\ \{v \in \mathcal{V} \,|\, oq \notin \alpha(v) \, \textit{for each} \; \neg oq \in L, \; oq \in (N_o \cup \mathcal{Q}_{\mathrm{N}}))\} \end{pmatrix}$$

Encoding Inequations Since the partitions in $\mathcal{P}$ are mutually disjoint the cardinality of a union of partitions is equal to the sum of the cardinalities of the partitions (e.g., if $P_1, P_2 \in \mathcal{P}$, then $\#(P_1 \cup P_1) = \#P_1 + \#P_2$) and one can encode a cardinality restriction on a partition's elements into an inequation using ξ such that $\xi(L, \geq, n) = \sum_{v \in V_L} \sigma(v) \geq n$, and $\xi(L, \leq, m) = \sum_{v \in V_L} \sigma(v) \leq m$ where $L \subseteq \mathcal{DS}$. With $\mathcal{SHOQ}$ we distinguish and encode the following cardinalities: (i) Concepts of the form $(\geq nR)$ and $(\leq mR)$ in the label of a node x express cardinality bounds n and m, respectively, on the set $FIL(R, x)$

[1] When C and $\dot\neg C$ are both used as qualifying concepts, we only include C in $\mathcal{DS}$.

for some $R \in N_{\mathrm{R}}$. These bounds can be reduced into inequations using $\xi(L, \geq, n)$ and $\xi(L, \leq, m)$ for $L = \{R\}$ or $L = \{R, q\}$, if additionally, we have $\forall S.C$ such that $(R \sqsubseteq_* S)$ with $C \in DS$ and $\theta(q) = C$. (ii) Nominals represent singleton sets. This cardinality bound can be encoded into inequations using $\xi(\{o\}, \geq, 1)$ and $\xi(\{o\}, \leq, 1)$ for each nominal $o \in N_{\mathrm{o}}$. When cardinalities (i) and (ii) are both encoded into inequations, the interaction between nominals and role fillers is handled while preserving the semantics of nominals.

Getting a Solution Given a set ξ of inequations, an integer solution defines the mapping σ for each variable v occurring in ξ to a non-negative integer n denoting the cardinality of the corresponding partition. For example, assuming $\sigma(v_a) = 1$ and $\alpha(v_a) = \{R_1, R_2\}$, this means that the corresponding partition $(\alpha(v_a))^{\mathcal{I}}$ must have 1 element; $\#(FIL(R_1) \cap FIL(R_2)) = 1$. Additionally, by setting the objective function to minimize the sum of all variables, a minimum number of role fillers is ensured at each level. A solution σ then defines a distribution of individuals that is consistent with the numerical restrictions encoded in ξ.

3.3 Tableau Algorithm

The tableau algorithm described in this section relies on an inequation solver working together with tableau expansion rules to construct a representation of a tableau model using a *compressed completion graph*.

Definition 3. [Compressed Completion Graph] A (CCG) is a directed graph $G = (V, E, \mathcal{L}, \mathcal{L}_{\mathrm{E}}, \mathcal{L}_{\mathrm{P}})$, where nodes represent domain elements and the edges between the nodes represent role relations. Each node $x \in V$ is labeled with three labels: $\mathcal{L}(x)$, $\mathcal{L}_E(x)$ and $\mathcal{L}_P(x)$, and each edge $\langle x, y\rangle \in E$ is labeled with a set, $\mathcal{L}(\langle x, y\rangle) \subseteq N_{\mathrm{R}}$, of role names. $\mathcal{L}(x)$ denotes a set of concept expressions, $\mathcal{L}(x) \subseteq clos(\mathcal{T})$, that the domain element, i_x, represented by x must satisfy. $\mathcal{L}_P(x)$ denotes a non-atomic partition name (i.e., we consider the set $\mathcal{L}_P(x)$ as a name) and is used as a tag for x based on the partition that i_x belongs to. A partition name $\mathcal{L}_P(x) \subseteq \mathcal{DS}$ can include roles, nominals, or qualification names.

When a role $R \in N_{\mathrm{R}}$ appears in $\mathcal{L}_P(x)$ this means that i_x belongs to the partition for *R-fillers* and can therefore be used as an *R-filler*. When a nominal $o \in N_{\mathrm{o}}$ appears in $\mathcal{L}_P(x)$ this means that $i_x \in o^{\mathcal{I}}$, and o is added to $\mathcal{L}(x)$ when x is created. On the other hand if a nominal $i \in N_{\mathrm{o}}$ does not appear in $\mathcal{L}_P(x)$ this means that i_x satisfies the complement of i, $i_x \in (\neg i)^{\mathcal{I}}$ and $(\neg i)$ is added to $\mathcal{L}(x)$ when x is created (see *fil*-Rule). When a qualification name $q \in Q_N$ appears in $\mathcal{L}_P(x)$ this means that i_x satisfies the qualifying concept mapped to q, $i_x \in \theta(q)^{\mathcal{I}}$ and $\theta(q)$ is added to $\mathcal{L}(x)$ when x is created. As with the nominals case, if a qualification name $p \in Q_N$ does not appear in $\mathcal{L}_P(x)$ this means that i_x satisfies the complement of the qualifying concept mapped to p, $i_x \in \dot\neg(\theta(p))^{\mathcal{I}}$ and $\dot\neg\theta(p)$ is added to $\mathcal{L}(x)$ when x is created (see *fil*-Rule). Using $\mathcal{L}_P(x)$ as a tagging allows for the re-use of nodes instead of creating new ones.

$\mathcal{L}_{\mathrm{E}}(x)$ denotes a set ξ_x of inequations that must have a non-negative integer solution. The set ξ_x is the encoding of number restrictions and qualifications that must be satisfied for x. In order to make sure that numerical restrictions local for a node x are satisfied while the global restrictions carried with nominals are not violated, the inequation solver collects all inequations and variable assignment in $\mathcal{L}_{\mathrm{E}}$ before returning a distribution. This makes sure that an initial distribution of nominals and/or role fillers is globally preserved while still satisfying the numerical restrictions (a distribution of role fillers) at each level.

Definition 4. [Proxy node] A proxy node is a representative for the elements of each partition. Proxy nodes can be used since partitions are disjoint and all elements within a partition P satisfy common restrictions (see [2] for proofs).

Let us assume that KB($\mathcal{T}$,$\mathcal{R}$) such that $\mathcal{T}$ has been preprocessed and rewritten into $C'_{\mathcal{T}}$. To check KB consistency, the algorithm starts with the completion graph $G = (\{r_0\}, \emptyset, \mathcal{L}, \mathcal{L}_{\mathrm{E}})$. With $\mathcal{L}_{\mathrm{E}}(r_o) = \bigcup_{o \in N_o}\{\xi(o, \leq, 1), \xi(o, \geq, 1)\}$ which is an encoding of the nominal semantics into inequations. The node r_0 is artificial and is not considered as part of the tableau model, it is only used to process the numerical restrictions on nominals using the inequation solver which returns a distribution for them.

The distribution of nominals is processed by the ***fil*-Rule** which is used to generate individual nodes depending on the solution (σ) returned by the inequation solver. The *fil*-Rule rule is fired for every non-empty partition P using $\sigma(v)$. It generates one proxy node y as the representative for the m elements assigned to $P^{\mathcal{I}}$ by the inequation solver. In the case of nominals, m is always equal to 1. The node y is tagged with its partition name using $\alpha(v)$ in $\mathcal{L}_P(y)$. The set of inequations is accumulated in $\mathcal{L}_E(y)$. Nominals and qualifications satisfied by the partition elements are extracted from the partition name and added to $\mathcal{L}(y)$. $C'_{\mathcal{T}}$ is also added to $\mathcal{L}(y)$ to make sure that every node created by the *fil*-Rule also satisfies $C'_{\mathcal{T}}$.

After at least one nominal is created, G is expanded by applying the expansion rules given in Fig. 1 until no rules are applicable or a clash occurs. The $\sqcap$-Rule, $\sqcup$-Rule, $\forall$-Rule and the $\forall_+$-Rule are similar to the ones in [1, 7]. The **$\forall_\backslash$-Rule** is used to enforce the semantics of the role set difference operator $\forall_\backslash$ introduced at preprocessing by making sure that all *R-fillers* are labelled. The **$\bowtie$-Rule** encodes the numerical restrictions in the label $\mathcal{L}$ of a node x, for some role $R \in N_R$, into a set of inequations maintained in $\mathcal{L}_{\mathrm{E}}(x)$. The inequation solver is always active and responsible for finding a non-negative integer solution σ or triggering a clash if no solution is possible. If the inequations added by this rule do not trigger a clash, then the encoded at-least/at-most restriction can be satisfied by a possible distribution of role fillers. We distinguish two cases.

Case (i): R-fillers of x must also satisfy a qualifying concept C due to a $\forall S.C$ restriction on a role S such that $R \sqsubseteq_* S$ and C is either a nominal or a qualifying concept such that $\theta^-(C)$ in $\mathcal{DS}$. Then the numerical restriction is encoded on partitions $P \in \mathcal{P}$ with $P^{\mathcal{I}} \subseteq (C^{\mathcal{I}} \cap FIL(R))$ which means $\{R, \theta^-(C)\} \subseteq P$.

Case (ii): There exist no qualified restrictions on R-fillers of x due to a $\forall$ restriction on a role S such that $R \sqsubseteq_* S$. In this case the numerical restriction is encoded on partitions $P \in \mathcal{P}$ with $P^{\mathcal{I}} \subseteq FIL(R)$ which means $\{R\} \subseteq P$.

***ch*-Rule.** This rule checks for empty partitions while ensuring completeness of the algorithm. Given a set of inequations in the label $\mathcal{L}_{\mathrm{E}}(x)$ of a node x and a variable v such that $\alpha(v) = P$ and $P \in \mathcal{P}$ we distinguish between two cases.

(i) $P^{\mathcal{I}}$ must be empty ($v \leq 0$); this happens when restrictions on elements of this partition trigger a clash because the signature of P cannot be satisfied. For instance,

$\sqcap$-Rule	**If** $C \sqcap D \in \mathcal{L}(x)$, and $\{C, D\} \nsubseteq \mathcal{L}(x)$ **Then** set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{C, D\}$.
$\sqcup$-Rule	**If** $C \sqcup D \in \mathcal{L}(x)$, and $\{C, D\} \cap \mathcal{L}(x) = \emptyset$ **Then** set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{E\}$ with $E \in \{C, D\}$.
$\forall$-Rule	**If** $\forall R.C \in \mathcal{L}(x)$ and there exists y such that $\mathcal{L}(\langle x, y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$, and $C \notin \mathcal{L}(y)$ **Then** set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$.
$\forall_+$-Rule	**If** $\forall R.C \in \mathcal{L}(x)$ and there exists y such that $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) \neq \emptyset$, $S \in N_{R^+}$ with $S \sqsubseteq_* R$, and $\forall S.C \notin \mathcal{L}(y)$ **Then** set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{\forall S.C\}$.
$\bowtie$-Rule	**If** $(\bowtie nR) \in \mathcal{L}(x)$ for $\bowtie \in \{\leq, \geq\}$, **Then If** $\forall S.C \in \mathcal{L}(x)$ with $R \sqsubseteq_* S$ and $\xi(\{R, \theta^-(C)\}, \bowtie, n) \notin \mathcal{L}_E(x)$ **Then** set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \{\xi(\{R, \theta^-(C)\}, \bowtie, n)\}$. **Else If** $\xi(\{R\}, \bowtie, n) \notin \mathcal{L}_E(x)$ **Then** set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \{\xi(\{R\}, \bowtie, n)\}$.
ch-Rule	**If** there exists v occurring in $\mathcal{L}_E(x)$ such that $\{v \geq 1, v \leq 0\} \cap \mathcal{L}_E(x) = \emptyset$ **Then** set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \{V\}$, $V \in \{v \geq 1, v \leq 0\}$.
e-Rule	**If** $(\bowtie nR) \in \mathcal{L}(x)$, and there exists y such that $R \in \mathcal{L}_P(y)$ and $R \notin \mathcal{L}(\langle x, y\rangle)$ **Then If** $\forall S.C \in \mathcal{L}(x)$ with $R \sqsubseteq_* S$ and $\theta^-(C) \in \mathcal{L}_P(y)$, OR $\forall S.C \notin \mathcal{L}(x)$ with $R \sqsubseteq_* S$ **Then** set $\mathcal{L}(\langle x, y\rangle) = \mathcal{L}(\langle x, y\rangle) \cup \{R\}$, and **If** $\mathcal{L}_E(x) \nsubseteq \mathcal{L}_E(y)$ **Then** set $\mathcal{L}_E(y) = \mathcal{L}_E(y) \cup \mathcal{L}_E(x)$.
fil-Rule	**If** there exists v occurring in $\mathcal{L}_E(x)$ with $\sigma(v) = m$ and $m > 0$, and there exists no y with $\mathcal{L}_P(y) = \alpha(v)$ **Then** 1. create a new node y, 2. set $\mathcal{L}_P(y) = \alpha(v)$, 3. set $\mathcal{L}_E(y) = \mathcal{L}_E(x)$, 4. set $\mathcal{L}(y) = \bigcup_{o \in (\alpha(v) \cap N_o)} o \cup \bigcup_{i \in (N_o \setminus \alpha(v))} \neg i \cup \bigcup_{q \in (Q_N \cap \alpha(v))} \theta(q) \cup \bigcup_{p \in (Q_N \setminus (Q_N \cap \alpha(v)))} \dot{\neg}_Q \theta(p) \cup \{C'_{\mathcal{T}}\}$
$\forall_\setminus$-Rule	**If** $\forall(R \setminus S).C \in \mathcal{L}(x)$, and there exists y such that $\mathcal{L}(\langle x, y\rangle) \cap (H(R) \cup \{R\}) \neq \emptyset$, $\mathcal{L}(\langle x, y\rangle) \cap (H(S) \cup \{S\}) = \emptyset$, and $C \notin \mathcal{L}(y)$ **Then** set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$.

Fig. 1. Completion rules for $\mathcal{SHOQ}$ (in groups of decreasing priority from top to bottom)

if $\{\forall R_1.A, \forall R_2.\neg A\} \subseteq \mathcal{L}(x)$, $v_{R_1R_2} \geq 1 \in \mathcal{L}_E(x)$ and there exists a node y with $\mathcal{L}_P(y) = \{R_1, R_2\}$ and $\{R_1, R_2\} \subseteq \mathcal{L}(\langle x, y\rangle)$, the qualifications on R_1 and R_2-fillers trigger a clash $\{A, \neg A\} \subseteq \mathcal{L}(y)$ and $v_{R_1R_2} \leq 0$ is enforced.

(ii) $P^{\mathcal{I}}$ must have at least one element ($1 \leq m \leq \sigma(v)$); if $P^{\mathcal{I}}$ can have at least one element without causing any logical clash, this means that the signature of P is satisfiable and we can also have m elements in $P^{\mathcal{I}}$ without a clash.

***e*-Rule.** This rule creates the edges between the proxy nodes created by the *fil*-Rule. If $\geq nR \in \mathcal{L}(x)$ for some R, this means that x must be connected to a number r of R-fillers such that $n \leq r$. If $\leq mR \in \mathcal{L}(x)$ then x could be connected to a maximum number r' of R-fillers such that $r' \leq m$. If there exists a node y such that $R \in \mathcal{L}_P(y)$, this means that a distribution of R-fillers has been assigned by the inequation solver such that the numbers n and m are satisfied and y is a representative for a number p of R-fillers such that $r \leq p \leq r'$. We distinguish between two cases.

(i): R-fillers of x must also satisfy a qualifying concept C due to a $\forall S.C$ restriction on a role S such that $R \sqsubseteq_* S$. In this case, if $\theta^-(C)$ is also in $\mathcal{L}_P(y)$ then the partition represented by y intersects with $C^{\mathcal{I}}$ and y is a member of C.

(ii): There exists no qualified restrictions on R-fillers. In this case there is no restriction on the partitions intersecting with R-fillers.

In both cases, an edge can safely be created between x and y such that $R \in \mathcal{L}(\langle x, y\rangle)$ and this edge is also a representative for the number p of edges between x and the p elements represented by y. If S is also in $\mathcal{L}_P(y)$ this means that the p R-fillers represented by y are also S-fillers and y is a representative for a partition $p \in \mathcal{P}$ such that $p^{\mathcal{I}} \subseteq FIL(R) \cap FIL(S)$. Therefore y can be re-used to connect x or another node y having $\geq n'S$ or $\leq m'S$, $n' \leq n$ and $m' \geq m$, in their label.

Definition 5. [Strategy of Rule Application] Given a node x in the completion graph, the rules are triggered when applicable based on the following order (listed with decreasing priority) in order to ensure completeness of the algorithm (see [2] for details): 1. $\sqcap$-*Rule*, $\sqcup$-*Rule*, $\forall$-*Rule*, $\forall_+$-Rule, *ch-Rule*, $\bowtie$-*Rule*, *e-Rule*. These rules can be fired in arbitrary order. 2. *fil-Rule*. 3. $\forall_{\backslash}$-*Rule*.

Definition 6. [Clash] A node x in $(V \setminus \{r_0\})$ is said to contain a *clash* if: (i) $\{C, \neg C\} \subseteq \mathcal{L}(x)$, or (ii) a subset of inequations $\xi_x \subseteq \mathcal{L}_{\mathrm{E}}(x)$ does not admit a non-negative integer solution, this case is decided by the inequation solver.

When no rules are applicable or there is a clash, a *completion graph* is said to be *complete*. When G is complete and clash free it means that a model exists for KB($\mathcal{T}, \mathcal{R}$) satisfying the numerical and the logical restrictions; the algorithm returns that KB($\mathcal{T}, \mathcal{R}$) is consistent, otherwise it returns that KB($\mathcal{T}, \mathcal{R}$) is inconsistent.

4 Optimizing Algebraic Tableau Reasoning

The main goal for introducing algebraic reasoning to DL is to efficiently handle reasoning with QCRs and/or nominals. Although global partitioning of domain elements gives a worst-case double exponential algorithm (see [2] for proofs), one can exploit its high level of information to adapt well known and devise new optimization techniques for improving reasoning with nominals and QCRs. The atomic decomposition technique allows a more semantically structured model construction algorithm which exhibits a high level of information on cardinalities implied by QCRs and nominals.

The next two optimization techniques exploit simple interactions between so-called “told nominals” and QCRs to discard unnecessary partitions and impose some ordering on applying the ch-Rule for nominal variables.

Discarding Partitions This optimization aims at reducing the number of partitions and their variables. It does this at the preprocessing level by collecting and analyzing the following interactions between nominals and QCRs.

(i) We have $\geq nR.C$ with $C \equiv o_1 \sqcup \cdots \sqcup o_n$ or $C \sqsubseteq o_1 \sqcup \cdots \sqcup o_n$. For example, $\geq 1R.o$ is rewritten into $\geq 1R' \sqcap \forall R'.o$ and this means that the partition for R'-fillers must intersect with the partition for the nominal o and, therefore, the partitions for R'-fillers that do not intersect with o can be safely discarded when computing the global partitioning.

(ii) We have $\leq nR.C$ with $C \equiv o_1 \sqcup \cdots \sqcup o_n$ or $C \sqsubseteq o_1 \sqcup \cdots \sqcup o_n$. For example, $\leq 1R.o$ is rewritten into $\leq 1R' \sqcap \forall R'.o \sqcap \forall R \backslash R'.\neg o$ and similar to the case with $\geq nR.C$ the partitions for R'-fillers that do not intersect with o can be discarded. Additionally, the partitions for R-fillers that do not intersect with R'-fillers and intersect with o can also be discarded.

Variable Preference For each nominal o, only one variable $v \in V_o$ can be assigned ≥ 1 by the ch-Rule. This heuristic aims at selecting nominal variables that are more likely to succeed. It does this similarly to the case of discarding partitions and allows the ch-Rule to branch on a partition, where nominals intersect with their interacting roles, before branching on a variable, where these nominals do not intersect with the role fillers. For example, we have two variables v_1 and v_2 for a nominal o with $\alpha(v_1) = \{o\}$ and $\alpha(v_2) = \{o, R'\}$ and R' is mapped to $\{o\}$. The variable-preference heuristic then directs the ch-Rule to branch on $v_2 \geq 1$ before branching on $v_1 \geq 1$.

Skip UnSat ch-Rule This optimization affects the ch-Rule and aims at bypassing choice points that are known to lead to a clash. For example, if the ch-Rule is applied to a variable v_a with $o \in \alpha(v_a)$ for $o \in N_\text{o}$ and $v \leq 0$ for all $v \in V_o$, this means that branching on $v_a \leq 0$ will result in a clash because the encoded inequation $\xi(o, \geq, 1)$ for o becomes infeasible. The branch for $v_a \leq 0$ can be safely bypassed. If $R \in \alpha(v_a)$ for some $R \in N_R$ and we have $v \leq 0$ for all $v \in V_R$, then the branch for $v_a \leq 0$ can therefore be safely bypassed if v_a occurs in an inequation encoding an at-least restriction. Similarly, the branch for $v_a \geq 1$ is discarded if assigning v_a a value ≥ 1 renders the inequation where v_a occurs obviously infeasible.

Using noGood Variables A variable v is assigned to be a noGood if v must have the value zero. This can happen for a partition P where $\alpha(v) = P$ must be empty because no domain element can be distributed over P without causing a clash. Using the *ch*-Rule a semantic split is performed over each partition's elements; $v \geq 1$ is the case when the restrictions on the partition's elements can be satisfied, and $v \leq 0$ means the restrictions on the partition's elements cannot be satisfied.

Skip UnSat OR-Rule This optimization affects the $\sqcup$-Rule and aims at bypassing choice points that are known to lead to a clash. When the $\sqcup$-Rule is applied to a node y, the branch adding C to $\mathcal{L}(y)$ can be discarded for the following cases: (i) C is a restriction $\geq nR$ and all variables mapped to R are noGood variables, then choosing this disjunct will result in an arithmetic clash. (ii) C is a nominal o and y is assigned to a partition P intersecting with $\neg o$ ($\{o\} \notin P$). (iii) C is the complement of a nominal, $\neg o$, and y is assigned to a partition P intersecting with o ($\{o\} \in P$).

Dependency Directed Backtracking This is a well known optimization technique which allows a search algorithm to bypass choice points. We identify three types of clashes: the logical, OR, and arithmetic clash, and for each type a clash handler is responsible for setting the next choice point to explore.

Logical Clash Handler If a node y has $\{C, \neg C\} \subseteq \mathcal{L}(y)$, y is said to contain a logical clash. The logical clash handler analyzes the clash sources looking for alternative choice points where the algorithm can backjump to. If no such alternative choice is found, then y cannot survive without causing a clash. One can safely assume that the corresponding partition represented in $\mathcal{L}_p(y)$ must be empty and the variable v with $\mathcal{L}_P(y) = \alpha(v)$ must be zero. The algorithm can backjump to the ch-Rule choice point where $v \leq 0$ and safely bypass the choice points with $v \geq 1$. Additionally, if the noGood variable optimization is turned on, then v is also assigned to be a noGood variable.

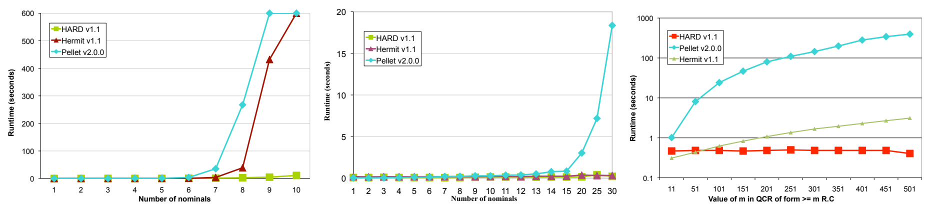
OR Clash Handler If we have $\neg o \sqcup \neg C \in \mathcal{L}(y)$ and we have $o, C \in \mathcal{L}_P(y)$ then the node y will not survive because all choice points generated by the $\sqcup$-Rule will result in a clash ($\{o, \neg o\} \subseteq \mathcal{L}(y)$ or $\{C, \neg C\} \subseteq \mathcal{L}(y)$). The node y is said to contain an OR-clash and the variable v with $\mathcal{L}_P(y) = \alpha(v)$ must be zero. The algorithm can backjump to the ch-Rule choice point where $v \leq 0$ and safely bypass the choice points with $v \geq 1$. An OR-clash can only be detected if the "Skip UnSat" optimization is turned on and the OR clash handler cannot find alternative choice points because the applicability of the OR-Rule returns an empty list of choice points, i.e., all choices would clash.

Arithmetic Clash Handler An arithmetic clash is detected when the system of inequations cannot have a solution. The following arithmetic clashes can be detected and handled even before running the Simplex procedure (i.e., as soon as inequations are added by the $\bowtie$-Rule). *Clash A*: If there exists a node $y \in G$ such that $\xi(L, \geq, m) \in \mathcal{L}_E(y)$ and $v \leq 0 \in \mathcal{L}_E(y)$ for all $v \in V_L$ (due to the *ch*-Rule), then $\xi(L, \geq, m)$ is infeasible and renders ξ_y infeasible. *Clash B*: If there exists a node $y \in G$ such that $\xi(L, \bowtie, m) \in \mathcal{L}_E(y)$ and for all $v \in V_L$, v has been assigned a value $\sigma(v)$ (due to a previous distribution σ) such that $\sum_{v \in V_L} \sigma(v)$ does not satisfy m. *Clash C*: If there exists a node $y \in G$ such that $\xi(L, \bowtie, m) \in \mathcal{L}_E(y)$ and for some $v_n \in V_L$, the ch-Rule must skip the branch where $v_n \geq 1$ because v_n is a noGood and branching on $v_n \leq 0$ triggers a clash of type A. In all three cases the algorithm can backjump to a branching point for some $v \in V_L$ where $v \geq 1$ and v has not been assigned to be a noGood.

5 Evaluation: First Experimental Results

Our prototype reasoner HARD (Hybrid Algebraic Reasoner for DL) is implemented in Java and uses the OWL-API. We integrated the reasoner interfaces of Pellet v.2.0.0 [11] and HermiT v.1.1 [9] into our implementation and run KB consistency tests using HARD, HermiT, or Pellet. This first evaluation was targeted to test how the algebraic tableau in combination with the proposed optimizations scales for KBs exhibiting interactions between nominals and QCRs. Unfortunately, there are not many suitable ontologies available because QCRs were only recently added to OWL 2 and the ones that are available do not serve well as benchmarks for HARD because their potential difficulty is not caused by interactions between nominals and QCRs. Furthermore, HARD was designed as a research prototype to demonstrate the effectiveness of our algebraic tableau approach and intentionally does not implement most of the optimization techniques implemented by other DL reasoners. It is therefore not the focus of this paper to evaluate HARD's performance for real world ontologies due to the overhead necessary to implement other optimization techniques not related to this line of research.

A typical nominal-QCR interaction occurs when a KB includes axioms of the form $C \equiv o_1 \sqcup \cdots \sqcup o_n$ with $o_1, \ldots, o_n$ nominals, and $D \sqsubseteq\ \geq mR.C$ or $D \sqsubseteq\ \leq mR.C$ with $n, m \geq 0$ in $\mathcal{T}$. Our claim is that these patterns are more likely to occur in real world ontologies. For example, in a KB used to classify countries based on their spoken languages one could find axioms of the form $SSC \equiv Argentina \sqcup Belize \sqcup Bolivia \sqcup \cdots \sqcup Venezuela$ (SSC stands for *Spanish_Speaking_Countries*) and $South\_America \sqsubseteq\ \geq 11\, Includes.SSC \sqcap \leq 11\, Includes.SSC$, and $Caribbean \sqsubseteq\ \geq 3\, Includes.SSC$ where $Argentina, \ldots, Venezuela$ are all distinct nominals representing unique countries.



(a) Increasing n in $\mathcal{T}_A$ ($n = m$) (b) Increasing n in $\mathcal{T}_B$ ($n = m$) (c) Increasing m in $\mathcal{T}_B$ (log-linear scale)

Fig. 2. Evaluation of HARD with HermiT and Pellet (all runtimes in seconds)

We developed two sets of benchmarks consisting of the simple TBoxes $\mathcal{T}_A$ and $\mathcal{T}_B$ defined below where $o_1, \ldots, o_n$ are all disjoint nominals and n and m positive numbers: $\mathcal{T}_A = \{C \equiv o_1 \sqcup \ldots \sqcup o_n,\ D \sqsubseteq\ \geq (m+1)\, R.C\}, \mathcal{T}_B = \{C \equiv o_1 \sqcup \ldots \sqcup o_n,\ D \sqsubseteq\ \geq m\, R.C\}$

In a first set of benchmarks we set $n = m$ and increment n by 1. Notice that due to the nominals semantics and their interaction with $FIL(R)$, $\mathcal{T}_A$ is inconsistent because the cardinality of $FIL(R)$ can be at most n while $\mathcal{T}_B$ is consistent. The results of the tests are shown in Fig. 2 (the runtimes were computed as the average of 10 independent runs). For HARD all optimization techniques described above were switched on. In the case of inconsistent KBs (Fig. 2(a)) one can easily see that HARD outperforms the other reasoners whose performance quickly degrades even with small values of n. In the case of consistent KBs (Fig. 2(b)) HARD performs similar to HermiT while Pellet's performance degrades. In a second set of tests for consistent KBs the size of m in $\mathcal{T}_B$ increases but the number of nominals remains constant; we set $n = 5$ and increment m by 50. Fig. 2(c) clearly demonstrates that HARD's performance remains constant while the performance of the other reasoners severely degrades as m grows (observe the logarithmic scale for the runtime).

6 Conclusion and Future Work

We exploited the high level of information of the algebraic method and presented optimization techniques related to nominals, QCRs and their interactions. Our first experimental results show that algebraic reasoning outperforms existing DL reasoning methods by several orders of magnitude, although we used small examples. One might argue that these results are based on special case patterns, however, it is clear that such patterns are inevitable for designing some real world ontologies. It is part of ongoing work to report on performance improvements in more general cases. We are also working on extending our calculus to $\mathcal{SHOIQ}$ by additionally allowing inverse roles. Our conjecture is that the worst-case complexity of our calculus might remain unchanged and, thus, would become worst-case optimal for $\mathcal{SHOIQ}$.

References

1. J. Faddoul, N. Farsinia, V. Haarslev, and R. Möller. A hybrid tableau algorithm for $\mathcal{ALCQ}$. In *Proc. of the 2008 Int. Workshop on Description Logics, also in 18th European Conference on Artificial Intelligence (ECAI 2008)*, pages 725–726, 2008.
2. J. Faddoul and V. Haarslev. Algebraic tableau reasoning for the description logic $\mathcal{SHOQ}$. *Logic Journal of the IGPL, Special Issue on Hybrid Logics*, 2010. To appear.
3. J. Faddoul, V. Haarslev, and R. Möller. Algebraic tableau algorithm for $\mathcal{ALCOQ}$. In *Proc. of the 2009 Int. Workshop on Description Logics (DL 2009)*, 2009.
4. N. Farsiniamarj and V. Haarslev. Practical reasoning with qualified number restrictions: A hybrid Abox calculus for the description logic $\mathcal{SHQ}$. *AI Communications*, Special Issue on Practical Aspects of Automated Reasoning, 2009. To appear.
5. V. Haarslev and R. Möller. Optimizing reasoning in description logics with qualified number restrictions. In *Description Logics*, pages 142–151, 2001.
6. V. Haarslev, M. Timmann, and R. Möller. Combining tableaux and algebraic methods for reasoning with qualified number restrictions. In *Description Logics*, pages 152–161, 2001.
7. I. Horrocks and U. Sattler. Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)*, pages 199–204. Morgan Kaufmann, Los Altos, 2001.
8. Y. Kazakov and B. Motik. A resolution-based decision procedure for $\mathcal{SHOIQ}$. *Journal of Automated Reasoning*, 40(2-3):89–116, 2008.
9. B. Motik, R. Shearer, and I. Horrocks. Optimized reasoning in description logics using hypertableaux. In *(CADE-21)*, volume 4603 of *Lecture Notes in Artificial Intelligence*, pages 67–83. Springer, 2007.
10. H. J. Ohlbach and J. Koehler. Modal logics, description logics and arithmetic reasoning. *Artificial Intelligence*, 109(1-2):1–31, 1999.
11. B. Parsia, B. Cuenca Grau, and E. Sirin. From wine to water: Optimizing description logic reasoning for nominals. In *Proc. of the 10th Int. Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 90–99, 2006.