A Hybrid Tableau Algorithm for $\mathcal{ALCQ}$

Jocelyne Faddoul[1], Nasim Farsinia[1], Volker Haarslev[1], and Ralf Möller[2]

[1] Concordia University, Montreal, Canada
{j_faddou, n_farsin, haarslev}@encs.concordia.ca
[2] Hamburg University of Technology, Hamburg, Germany
r.f.moeller@tuhh.de

Abstract. We propose an approach for extending a tableau-based satisfiability algorithm by an arithmetic component. The result is a hybrid satisfiability algorithm for the Description Logic (DL) $\mathcal{ALCQ}$ which extends $\mathcal{ALC}$ with *qualified number restrictions*. The hybrid approach ensures a more informed calculus which, on the one hand, adequately handles the interaction between *numerical* and *logical* restrictions of descriptions, and on the other hand, when applied is a very promising framework for average case optimizations.

1 Introduction

Using the DL $\mathcal{ALCQ}$ one can express *numerical restrictions* on concepts with $(\exists R.C)$, $(\geq nR.C)$, and $(\leq mR.C)$ as well as *logical* ones, or both.[3] Such expressiveness means that a satisfiability algorithm for $\mathcal{ALCQ}$ not only needs to satisfy logical restrictions, but also numerical ones. For example, deciding the satisfiability of an $\mathcal{ALCQ}$ concept $(\geq 5S.C \sqcap \geq 5S.\neg D \sqcap \leq 2S.(C \sqcap \neg D))$ consists of finding a model with fillers for the role S such that at least 5 fillers are instances of C, at least 5 fillers are instances of $\neg D$, and at most 2 fillers are instances of $(C \sqcap \neg D)$. Most DL tableau-like algorithms [4, 1, 5] test the satisfiability of such a concept by blindly creating 10 fillers for S, 5 of which are instances of C and 5 are instances of $\neg D$. Then a concept choose rule non-deterministically assigns $(C \sqcap \neg D)$ or $(\neg C \sqcup D)$ to these fillers. Afterwards the at-most restriction $(\leq 2R.(C \sqcap \neg D))$ is checked for satisfiability by non-deterministically merging some pairs of fillers. Searching for a model in such an arithmetically un-informed way could become very inefficient. Even by means of optimization techniques such as the signature algorithm proposed in [2], they can easily fail with many at-most restrictions or with large numbers used in at-least restrictions $(\geq nR.C)$.

Our calculus, strongly inspired by [3, 6, 7], consists of a standard tableau for $\mathcal{ALC}$ [1] modified and extended to work with a constraint solver such as a linear inequation solver. The tableau rules encode numerical restrictions into a set of inequations using the *atomic decomposition* technique [7]. The set of inequations is processed by an inequation solver which finds a minimal integer solution (distribution of role fillers) satisfying the numerical restrictions, if one exists. The

[3] A universal restriction could be considered as both numerical and logical restriction $(\forall R.C \equiv \leq 0R.(\neg C))$.

tableau rules then take care of making sure that such distribution of role fillers also satisfies the logical restrictions. Considering a minimal distribution of fillers ensures that a corresponding model is of minimum size.

Since this hybrid algorithm collects all the information about arithmetic expressions before creating any role filler, it will not satisfy an at-least restriction by violating an at-most restriction and there is no need for a mechanism of merging role fillers. Moreover, it reasons about numerical restrictions by means of an inequation solver, thus its performance is not affected by the values of numbers in *qualified number restrictions*. Considering all these features the proposed hybrid algorithm is well suited to improve average case performance.

2 Preliminaries

We introduce the syntax and semantics of the description logic $\mathcal{ALCQ}$ along with some definitions and pre-processing of input expressions.

Let N_{C}, N_{R}, be non-empty and pair-wise disjoint sets of concept names and role names respectively. The set of $\mathcal{ALCQ}$ concepts is the smallest set such that: (i) every concept name $A \in N_{\mathrm{C}}$ is a concept, and (ii) if C, D are concepts and R is a role in N_{R} then $\neg C$, $(C \sqcup D)$, $(C \sqcap D)$, $(\exists R.C)$, $(\forall R.C)$, $(\geq nR.C)$, $(\leq nR.C)$ with n a non-negative integer are also concepts.

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of $\Delta^{\mathcal{I}}$, a non-empty set of individuals, called the domain of the interpretation and $\cdot^{\mathcal{I}}$ an interpretation function which maps each atomic concept $A \in N_{\mathrm{C}}$ to a subset of $\Delta^{\mathcal{I}}$, and each atomic role $R, S \in N_{\mathrm{R}}$ to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ such that, for all $\mathcal{ALCQ}$ concepts, the following must hold:

$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$,
$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}, (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
$(\forall R.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \text{for all } t \in \Delta^{\mathcal{I}} \text{ such that } \langle s,t\rangle \in R^{\mathcal{I}} \text{ then } t \in C^{\mathcal{I}}\}$,
$(\exists R.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \text{there exists } t \text{ such that } \langle s,t\rangle \in R^{\mathcal{I}} \text{ and } t \in C^{\mathcal{I}}\}$,
$(\geq nR.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \#(FIL(R,s) \cap C^{\mathcal{I}}) \geq n\}$,
$(\leq nR.C)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \#(FIL(R,s) \cap C^{\mathcal{I}}) \leq n\}$,

Where $\#$ denotes the cardinality of a set, and $FIL(R,s)$ is the set of *R-fillers* of an individual $s \in \Delta^{\mathcal{I}}$ for some role $R \in N_{\mathrm{R}}$ and is defined as: *FIL(R,s)* $= \{t \in \Delta^{\mathcal{I}} \mid \langle s,t\rangle \in R^{\mathcal{I}}\}$. The set of all *R-fillers* for R is then defined as *FIL(R)* $= \bigcup_{s \in \Delta^{\mathcal{I}}} FIL(R,s)$.

For an interpretation $\mathcal{I}$, a concept C is satisfiable iff $C^{\mathcal{I}} \neq \emptyset$ which means there exists an individual $s \in C^{\mathcal{I}}$ as an instance of C. We abbreviate $(\geq nR.\top)$ by $(\geq nR)$ and $(\leq mR.\top)$ by $(\leq mR)$ where $\top$ is equivalent to $C \sqcup \neg C$.

Arithmetic and Logical Expressions Given an $\mathcal{ALCQ}$ concept expression with C, D concepts and R a role name, we distinguish between arithmetic and logical expressions.

Expressions of the form $(\exists R.C)$, $(\geq nR.C)$, and $(\leq mR.C)$ hold arithmetic restrictions; they specify a lower (upper) bound on the cardinality of the set of *R-fillers*. For example, $s \in (\geq 2R.C)^{\mathcal{I}}$ imposes that at least 2 individuals s_1 and

s_2 must be *R-filler*s of s, and therefore $\#FIL(R,s) \geq 2$. An $(\exists R.C)$ expression can be converted to a $(\geq 1R.C)$ expression, therefore we restrict our attention to expressions of the form $(\geq nR.C)$, and $(\leq mR.C)$.

Expressions of the form $(C \sqcap D)$, $(C \sqcup D)$, and $\neg C$ hold logical restrictions using logical operators on concepts. We refer to these expressions as logical expressions. We give special attention to expressions of the form $(\forall R.C)$ since they hold both arithmetic and logical restrictions due to the following equivalences: $s \in (\forall R.C)^{\mathcal{I}} \Leftrightarrow s \in (\leq 0R.\neg C)^{\mathcal{I}}$, and $s \in (\forall R.C)^{\mathcal{I}} \Rightarrow FIL(R,s) \subseteq C^{\mathcal{I}}$

In the following, we assume all $\mathcal{ALCQ}$ concept expressions to be in their *negation normal form* (NNF) [5]. We use $\dot{\neg}C$ to denote the NNF of $\neg C$. We also define *clos(C)* to be the smallest set of concepts such that: (a) $C \in$ *clos(C)*, (b) if $D \in$ *clos(C)* then $\dot{\neg}D \in$ *clos(C)*, (c) if $(E \sqcap D)$ or $(E \sqcup D) \in$ *clos(C)* then $E, D \in$ *clos(C)*, and (d) if $(\geq nR.E)$ or $(\leq mR.E) \in$ *clos(C)* then $E \in$ *clos(C)*. The size of *clos(C)* is bounded by the size of C.

Re-writing $\mathcal{ALCQ}$ Arithmetic Expressions We define a concept operator $(\forall(R \backslash S).D)$ and a role implication operator $(R \sqsubseteq S)$ needed to preprocess the input descriptions before applying the calculus. These operators are based on set semantics such that given an interpretation $\mathcal{I}$, then $(\forall(R \backslash S).D)^{\mathcal{I}} = \{s \in \Delta^{\mathcal{I}} \mid \langle s,t \rangle \in R^{\mathcal{I}} \text{ and } \langle s,t \rangle \notin S^{\mathcal{I}} \Rightarrow t \in D^{\mathcal{I}}\}$ is satisfied and $(R^{\mathcal{I}} \subseteq S^{\mathcal{I}})$ is satisfied for each role implication $R \sqsubseteq S \in \varphi_{\mathrm{r}}$, where φ_{r} is a set of role implications. The definition of *FIL(R,s)* is extended to support the role implications in φ_{r} as follows: $FIL(R,s) = \{t \in \Delta^{\mathcal{I}} \mid \langle s,t \rangle \in S^{\mathcal{I}}, \text{ with } S \sqsubseteq R \in \varphi_{\mathrm{r}}\}$

Given an $\mathcal{ALCQ}$ concept E and an empty set φ_{r} of role implications, we re-write the arithmetic expressions in E such that:

- each $\geq nR.C$ is replaced with $\geq nR' \sqcap \forall R'.C$, with R' new in N_{R} and $R' \sqsubseteq R$ new in φ_{r}
- each $\leq mR.D$ is replaced with $\leq mR' \sqcap \forall R'.D \sqcap \forall(R \backslash R').\neg D$, with R' new in N_{R} and $R' \sqsubseteq R$ new in φ_{r}

Let φ denote E after re-writing arithmetic expressions, φ_{r} is treated as a special case *role hierarchy* where roles can have only one super role. It is easy to see that such re-writing preserves satisfiability (see [7]); E is satisfiable iff φ is satisfiable w.r.t φ_{r}.

Satisfiability of Arithmetic Expressions w.r.t φ_{r} Using Linear Inequation Solving. The atomic decomposition technique was used in [6] to reduce reasoning about cardinalities of role fillers to inequation solving. We use the same technique and reduction to decide the satisfiability of arithmetic expressions w.r.t φ_{r} using an inequation solver, and refer the reader to [6] for a proof about the correctness of this reduction.

For each role $R \in N_{\mathrm{R}}$ that is involved in an arithmetic expression the introduced sub-role R' enables some hierarchy of roles. We define $H(R) = \{R\} \cup \{R' \mid (R' \sqsubseteq R) \in \varphi_{\mathrm{r}}\}$ as the role hierarchy of R.

For every role $R' \in H(R)$, the set of *R'-fillers* forms a subset of the set of *R-fillers* ($FIL(R') \subseteq FIL(R)$). Using the atomic decomposition of $H(R)$ we can

define all possible intersections between *R-fillers* as disjoint partitions. Each $L \subseteq H(R)$ is associated with a unique partition $P(L) = \bigcap_{\mathrm{R}' \in \mathrm{L}} FIL(R')$. Furthermore $\mathcal{P}$ is the set of partitions defined for the decompositions of all hierarchies in φ_{r},

$$\mathcal{P} = \bigcup_{\mathrm{R} \in \mathrm{N_R}} \left(\begin{array}{l} \{L \,|\, L \subseteq H(R)\} \setminus \\ \{L \,|\, L \subseteq H(R), \exists R' \in L, R \notin L \text{ and } R' \sqsubseteq R \in \varphi_{\mathrm{r}}\} \end{array} \right)$$

We do not consider L such that $R' \in L$, $R \notin L$ for some $(R' \sqsubseteq R) \in \varphi_r$ since the corresponding partition $P(L)$ does not satisfy $FIL(R') \subseteq FIL(R)$ and therefore must be empty (please refer to Sect. 3.1 for an example).

We assign a variable name v for each partition L such that v is mapped to a non-negative integer value n which denotes the cardinality of $P(L)$. Let $\mathcal{V}$ be the set of all variable names, we maintain a mapping between variable names and their corresponding partitions using a function α: $\mathcal{V} \rightarrow \mathcal{P}$ such that for some non-negative integer n assigned to a variable v we have $n = \#P(\alpha(v))$.

Since the partitions are mutually disjoint and the cardinality function is additive, a lower (upper) bound n (m) on the cardinality of the set of role fillers $FIL(S)$ for some role $S \in H(R)$ can be reduced to an inequation of the form $\sum_{v \in V_S} v \geq n$ ($\sum_{v \in V_S} v \leq m$). V_S denotes the set of variable names mapped to partitions for a role S and is defined as $V_S = \{v \in \mathcal{V} \,|\, S \in \alpha(v)\}$. Thus, we can easily convert an expression of the form $(\geq nS)$ or $(\leq mS)$ into an inequation using ξ such that $\xi(S, \geq, n) = \sum_{v \in V_S} v \geq n$, and $\xi(S, \leq, m) = \sum_{v \in V_S} v \leq m$. Each variable v occurring in an inequation can be mapped to a non-negative integer p such that assuming $\alpha(v) = \{R', R''\}$, this means that $\#(FIL(R') \cap FIL(R'')) = p$ and the corresponding partition $P(\alpha(v))$ must have p fillers.

Satisfiability of $\mathcal{ALCQ}$ Expressions In general logical and arithmetic expressions in φ share symbols in common, therefore, their satisfiability cannot be decided independently. For this purpose we propose a hybrid algorithm, we define a tableau for $\mathcal{ALCQ}$ satisfiability and describe the algorithm in the following sections.

Definition 1 (Tableau). We define a tableau as an abstraction of a model for a concept expression φ. Let $\mathrm{T} = (\mathbf{S}, \mathcal{L}, \mathcal{E})$ be such a tableau for an $\mathcal{ALCQ}$ concept expression φ with $\mathbf{S}$ a non-empty set of individuals, $\mathcal{L}$: $\mathbf{S} \rightarrow 2^{\mathrm{cl}(\varphi)}$ a mapping between each individual and a set of concepts, and $\mathcal{E}$: $N_{\mathrm{R}} \rightarrow 2^{\mathrm{S} \times \mathrm{S}}$ a mapping between each role and a set of pairs of individuals in $\mathbf{S}$, then we have some $s \in \mathbf{S}$ with $\varphi \in \mathcal{L}(s)$. For all $s, t \in \mathbf{S}$, C,$D \in$ *clos(φ)*, and R, $S \in N_{\mathrm{R}}$, and $R^T(s) = \{t \in \mathbf{S} | \langle s, t \rangle \in \mathcal{E}(R)\}$, the following properties must hold:

1. If $C \in \mathcal{L}(\mathrm{s})$ then $\doteq C \notin \mathcal{L}(\mathrm{s})$.
2. If $C \sqcap D \in \mathcal{L}(\mathrm{s})$ then $C \in \mathcal{L}(s)$ and $D \in \mathcal{L}(s)$.
3. If $C \sqcup D \in \mathcal{L}(\mathrm{s})$ then $C \in \mathcal{L}(s)$ or $D \in \mathcal{L}(s)$.
4. If $\forall S.C \in \mathcal{L}(\mathrm{s})$ and $\langle s, t \rangle \in \mathcal{E}(S)$ then $C \in \mathcal{L}(t)$.
5. If $\forall (R \backslash S).C \in \mathcal{L}(\mathrm{s})$ and $\langle s, t \rangle \in \mathcal{E}(R)$, and $\langle s, t \rangle \notin \mathcal{E}(S)$ then $C \in \mathcal{L}(t)$.
6. If $(\geq nR) \in \mathcal{L}(\mathrm{s})$ then $\#R^T(s) \geq n$.
7. If $(\leq mR) \in \mathcal{L}(\mathrm{s})$ then $\#R^T(s) \leq m$.
8. If $\langle s, t \rangle \in \mathcal{E}(R)$ and $R \in H(S)$, then $\langle s, t \rangle \in \mathcal{E}(S)$.

Lemma 1. *An $\mathcal{ALCQ}$ concept φ is satisfiable iff there exists a tableau for φ.*

The proof is similar to the one found in [5] and property 5 of this tableau ensures that the semantics of the $\forall(R\backslash S).C$ operator is preserved.

3 A Hybrid Tableau Algorithm for $\mathcal{ALCQ}$

In this section, we describe a hybrid algorithm which decides the existence of a tableau for an $\mathcal{ALCQ}$ concept expression φ. Our algorithm is hybrid because it relies on tableau expansion rules working together with an inequation solver, the corresponding model is represented using a so-called *completion graph*.

Definition 2 (Completion Graph). A *completion graph* is a directed graph $G = (V, E, \mathcal{L}, \mathcal{L}_E)$ where each node $x \in V$ is labeled with $\mathcal{L}$ and $\mathcal{L}_E$ such that $\mathcal{L}(x)$ denotes a set of concept expressions, $\mathcal{L}(x) \subseteq clos(\varphi)$, and $\mathcal{L}_E(x)$ denotes a set of inequations that must have a non-negative integer solution. Each edge $\langle x, y\rangle \in E$ is labeled with a set, $\mathcal{L}(\langle x, y\rangle) \subseteq \mathcal{P}$, of role names.

We denote by ξ_x the set of inequations in $\mathcal{L}_E(x)$ obtained by converting the at-least and at-most restrictions in $\mathcal{L}(x)$. An integer solution σ for ξ_x maps each variable v occurring in ξ_x to a non-negative integer p such that σ is a distribution of role fillers of x. The distribution is consistent with the lower and upper bounds expressed in arithmetic restrictions and the hierarchy in φ_{r}.

Lemma 2 (Satisfiability of arithmetic expressions). *The at-least and at-most restrictions in $\mathcal{L}(x)$ are satisfiable w.r.t φ_{r} iff the encoded system ξ_x of inequations in $\mathcal{L}_E(x)$ admits a non-negative integer solution.*[4]

A node x in V is said to contain a clash if either (i) $\{C, \dot{\neg}C\} \subseteq L(x)$, or (ii) the set of inequations $\xi_x \subseteq \mathcal{L}_E(x)$ does not admit a non-negative integer solution. Case (ii) is decided by the inequation solver. When no rules are applicable or there is a clash, a *completion graph* is said to be *complete*.

To decide the satisfiability of a concept expression φ, the algorithm starts with the completion graph $G = (\{x\}, \emptyset, \{\varphi\}, \emptyset)$. G is then expanded by applying the expansion rules given in Fig. 1 until no more rules are applicable or a clash occurs. When G is complete and there is no clash, this means that the arithmetic expressions are satisfied as well as the logical ones: we have a model and the algorithm returns that φ is satisfiable.

Strategy of Rule Application We assign the *fil-Rule* the lowest priority; All other rules can be fired in arbitrary order. This strategy ensures that all arithmetic expressions for a node x are encoded and satisfied by the inequation solver before creating any role fillers. Which means that role fillers created are never merged nor removed from the graph; a distribution of role fillers either survives into a complete graph model or fails due to a clash. Note that this strategy can also help in early clash detection in the case when a clash is triggered by the inequation solver even before any role fillers are created.

[4] This lemma is an adapted form of Theorem 1 in [6].

$\sqcap$**-Rule**	**If** $C \sqcap D \in \mathcal{L}(x)$, and $\{C, D\} \not\subseteq \mathcal{L}(x)$
	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 $\mathcal{L}(x) = \mathcal{L}(x) \cup \{C\}$ **or** set $\mathcal{L}(x) = \mathcal{L}(x) \cup \{D\}$.
$\forall$**-Rule**	**If** $\forall R.C \in \mathcal{L}(x)$ and $R \in \mathcal{L}(\langle x, y\rangle)$ with $C \notin \mathcal{L}(y)$
	then set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$.
$\forall_{(\setminus)}$**-Rule**	**If** $\forall(R \setminus S).C \in \mathcal{L}(x)$, and there exists y such that
	$R \in \mathcal{L}(\langle x, y\rangle)$ and $S \notin \mathcal{L}(\langle x, y\rangle)$
	then set $\mathcal{L}(y) = \mathcal{L}(y) \cup \{C\}$.
$\leq$**-Rule**	**If** $(\leq nR) \in \mathcal{L}(x)$ and $\xi(R, \leq, n) \notin \mathcal{L}_E(x)$
	then set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \xi(R, \leq, n)$.
***ch*-Rule**	**If** there exists v occurring in $\mathcal{L}_E(x)$ with $\{v \geq 1, v \leq 0\} \cap \mathcal{L}_E(x) = \emptyset$
	then set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \{v \geq 1\}$ **or** set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \{v \leq 0\}$.
$\geq$**-Rule**	**If** $(\geq nR) \in \mathcal{L}(x)$ and $\xi(R, \geq, n) \notin \mathcal{L}_E(x)$
	then set $\mathcal{L}_E(x) = \mathcal{L}_E(x) \cup \xi(R, \geq, n)$.
***fil*-Rule**	**If** there exists v occurring in $\mathcal{L}_E(x)$ such that
	(i) $\sigma(v) = m$ with $m > 0$, and
	(ii) there are no m nodes $y_1 \ldots y_m$ with
	$\mathcal{L}(\langle x, y_i\rangle) = \alpha(v)$ for $1 \leq i \leq m$
	then create m new nodes $y_1 \ldots y_m$ and
	set $\mathcal{L}(\langle x, y_i\rangle) = \alpha(v)$ for $1 \leq i \leq m$.

Fig. 1. Expansion rules for $\mathcal{ALCQ}$

3.1 Explaining the Rules

The $\sqcap$-Rule, $\sqcup$-Rule and the $\forall$-Rule rules are similar to the ones in the standard tableaux rules for $\mathcal{ALC}$ [1, 5].

$\forall_{(\setminus)}$-Rule. This rule is used to ensure the semantics of the new operator $\forall(R \setminus S).D$ which is based on set difference between roles and is defined in Sect. 2. This rule ensures that all *R-fillers* are labelled, and together with the *ch*-Rule ensure the same effect of the *choose*-rule in [1] needed to detect the unsatisfiability of concepts like $((\geq 3R.C) \sqcap (\leq 1R.D) \sqcap (\leq 1R.\neg D))$.

$\leq$-Rule and $\geq$-Rule. These rules are responsible for encoding the arithmetic expressions in the label $\mathcal{L}(x)$ of a node x into a set (ξ_x) of inequations maintained in $\mathcal{L}_{\mathrm{E}}(x)$. An inequation solver is always active and is responsible for finding an integer solution σ for ξ_x or triggering a clash if no solution is possible (see Definition 2). As long as each inequation added by one of these rules does not trigger a clash, it means that the arithmetic expressions can be satisfied and there exists a possible distribution of role fillers.

ch-Rule. This rule is used to check for empty partitions. Given a set of inequations in the label $(\mathcal{L}_{\mathrm{E}})$ of a node x and a variable v corresponding to a partition $\alpha(v)$ in $\mathcal{P}$, we distinguish between two cases:

– (i) The case when a partition must be empty; this can happen when restrictions of individuals assigned to this partition trigger a clash. For instance, if $\{\forall R_1.C, \forall R_2.\neg C\} \subseteq \mathcal{L}(x)$ then an individual y assigned to a partition $P(\alpha(v))$ such that $\{R_1, R_2\} \subseteq \alpha(v)$ and $v \in \mathcal{L}_{\mathrm{E}}(x)$ triggers a clash $\{C, \neg C\} \subseteq \mathcal{L}(y)$ and therefore $P(\alpha(v))$ must be empty.
– (ii) The case when a partition can have at least one individual; if a partition can have one individual without causing any logical clash, this means that we can have m $(m \geq 1)$[5] individuals also in this partition without a clash.

Since the inequation solver is unaware of logical restrictions of filler domains we allow an explicit distinction between cases (i) and (ii). We do this by non-deterministically assigning ≤ 0 or ≥ 1 for each variable v occurring in $\mathcal{L}_{\mathrm{E}}(x)$.

fil-Rule. This rule is used to generate role fillers for a node x depending on the distribution (solution) returned by the inequation solver. The rule is fired for every partition $\alpha(v)$ and it generates the role fillers assigned to this partition by the inequation solver if they are not yet created, it creates m role fillers such that $\sigma(v) = m$. Since every partition is an intersection of role fillers, individuals assigned to a partition can be fillers of more than one role, $\alpha(v)$ returns the set of the corresponding role names for the partition $P(\alpha(v))$ and is therefore used to set the edge label between x and each role filler y.

Detailed Example To better illustrate the calculus, we consider the simple example of testing the satisfiability of $(\geq 5S.C \sqcap \geq 5S.\neg D \sqcap \leq 2S.(C \sqcap \neg D))$. After re-writing the arithmetic expressions we need to test the satisfiability of arithmetic and universal restrictions w.r.t φ_{r} with $(\geq 5S' \sqcap \geq 5S'' \sqcap \leq 2S''')$ as arithmetic restrictions, $\varphi_r = \{S' \sqsubseteq S, S'' \sqsubseteq S, S''' \sqsubseteq S\}$, and $(\forall S'.C \sqcap \forall S''.\neg D \sqcap \forall S'''.(C \sqcap \neg D) \sqcap \forall (S \backslash S''').(\neg C \sqcup D))$ as universal restrictions. To get all possible partitions between *S-fillers*, we can compute the power set of $\{S, S', S'', S'''\}$ and drop the partitions that do not satisfy the sub-role relationships in φ_r. Since S is in each partition, we can drop it from the set and assume it is implied. Then $\mathcal{P}$ = {{S'},{S''},{S'''},{$S'S''$} ,{$S'S'''$}, {$S''S'''$}, {$S'S''S'''$}} and $\mathcal{V} = \{v_{\mathrm{S}'}, v_{\mathrm{S}''}, v_{\mathrm{S}'''}, v_{\mathrm{S}'\mathrm{S}''}, v_{\mathrm{S}'\mathrm{S}'''}, v_{\mathrm{S}''\mathrm{S}'''}, v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''}\}$. After applying the $\geq$-Rule twice and the $\leq$-Rule once we get $\mathcal{L}_{\mathrm{E}}(x) = \xi_x$ such that

$$\xi_x = \begin{pmatrix} v_{\mathrm{S}'} + v_{\mathrm{S}'\mathrm{S}''} + v_{\mathrm{S}'\mathrm{S}'''} + v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''} \geq 5 \\ v_{\mathrm{S}''} + v_{\mathrm{S}'\mathrm{S}''} + v_{\mathrm{S}''\mathrm{S}'''} + v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''} \geq 5 \\ v_{\mathrm{S}'''} + v_{\mathrm{S}'\mathrm{S}'''} + v_{\mathrm{S}''\mathrm{S}'''} + v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''} \leq 2 \end{pmatrix} \tag{1}$$

After applying the *ch*-Rule until it is not applicable anymore we might come up with the case where $v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''} \geq 1$ and all other variables are ≤ 0. A clash is detected since no solution is possible for ξ_x and the model does not survive.

Considering another model with different choices for the ch-Rule rule such as $v_{\mathrm{S}'\mathrm{S}''} \geq 1$ while other variables are zero, the inequation solver returns a solution σ where $\sigma(v_{\mathrm{S}'\mathrm{S}''}) = 5$. The *fil*-Rule becomes applicable since there are

[5] The value of m is decided by the inequation solver.

no fillers, and creates 5 new filler nodes $y_1, \ldots, y_5$ such that $\mathcal{L}(\langle x, y_i \rangle) = \alpha(v_{\mathrm{S}'\mathrm{S}''})$ $= \{S', S''\}$ for $1 \leq i \leq 5$. The $\forall$-Rule is applicable twice and $\mathcal{L}(y_i) = \{C, \neg D\}$ for $1 \leq i \leq 5$. The $\forall_{(\setminus)}$-Rule becomes applicable and $\mathcal{L}(y_i) = \{C, \neg D, (\neg C \sqcup D)\}$. After applying the $\sqcup$-Rule each of the possible choices ends up in a clash.

A model with $v_{\mathrm{S}'} \geq 1, v_{\mathrm{S}''} \geq 1, v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''} \geq 1$ and all other variables ≤ 0, has a solution σ with $\sigma(v_{\mathrm{S}'\mathrm{S}''\mathrm{S}'''}) = 2$, $\sigma(v_{\mathrm{S}'}) = 3$, and $\sigma(v_{\mathrm{S}''}) = 3$. This model survives into a completion graph G with the nodes x, x_1, x_2, $y_1, \ldots, y_3$, $z_1, \ldots, z_3$ such that for $1 \leq i \leq 2$, $1 \leq j \leq 3$ we have:

$\mathcal{L}(\langle x, x_i \rangle) = \{S', S'', S'''\}$, $\mathcal{L}(\langle x, y_j \rangle) = \{S'\}$, $\mathcal{L}(\langle x, z_j \rangle) = \{S''\}$,

$\mathcal{L}(x_i) = \{C, \neg D\}$, $\mathcal{L}(y_i) = \{C, D\}$, $\mathcal{L}(z_i) = \{\neg C, \neg D\}$

Considering the worst case analysis for this example, there are 2^7 cases for the *ch*-Rule. Whereas, with a tableau calculus [5] there are at least 2^{10} cases for the *Choose*-rule given that 10 fillers are created by the $\geq$-Rule, not to forget the cases to add for the $\leq$-Rule when pairs of individuals are considered to be merged among the 10 fillers in order to satisfy the at-most restriction.

3.2 Proofs

Lemma 3. *Given an $\mathcal{ALCQ}$ concept φ and a complete and clash free completion graph G for φ. Let x be a node in V_{G}, $C, D \in N_{\mathrm{C}}$, $R \in N_{\mathrm{R}}$, we define $\varphi_{\mathrm{a}} = \{E \in \mathcal{L}(x) \,|\, E$ is of the form $\geq nR, \leq mR\}$ as the set of at-least and at-most restrictions to be satisfied for x. A solution σ for the encoding ξ_x of φ_{a} (i) does not violate the hierarchy φ_{r} introduced during preprocessing, and (ii) does not violate a restriction implied by any other operator used in φ.*

Proof. Our hybrid algorithm depends on an inequation solver to decide the satisfiability of arithmetic expressions (φ_{a}) encoded into ξ_x. Due to Lemma 2 and assuming the inequation solver is decidable, given a solution σ for ξ_x the tableau expansion rules take care of constructing a completion graph for role fillers based on the distribution reflected by σ. The algorithm needs to make sure that this distribution is consistent with φ_{r} and all restrictions implied by $\mathcal{ALCQ}$ operators. We distinguish the following two cases:

- (i) If the distribution is not consistent with φ_{r}, then for some $R' \sqsubseteq R \in \varphi_{\mathrm{r}}$, there exists an *$R'$-filler* y assigned to a partition L such that $P(L) = (FIL(R', x) \setminus FIL(R, x))$ which means that $R' \in L$ and $R \notin L$. This case is not possible, partitions such as L are eliminated from $\mathcal{P}$ and cannot be assigned any fillers.
- (ii) If the distribution is not consistent with other restrictions such as a universal restriction, then for some $\forall S.D \in \mathcal{L}(x)$, we have a node y such that $S \in \mathcal{L}(\langle x, y \rangle)$ and $\{\neg D\} \in \mathcal{L}(y)$. The $\forall$-Rule becomes applicable to x and D is added to $\mathcal{L}(y)$. Having $\{D, \neg D\} \subseteq \mathcal{L}(y)$ is not possible since G is assumed to be clash free. Similarly, we can prove that restrictions implied by the $\forall_{\setminus}$, and ($\sqcap$, $\sqcup$, $\neg$) operators cannot be violated.

Lemma 4 (Termination). *Given an $\mathcal{ALCQ}$ concept expression φ, the proposed hybrid algorithm terminates.*

Proof. Let $l = \#clos(\varphi)$, $r = \#N_{\mathrm{R}}$ and max be the maximum number used in an arithmetic expression. Termination is guaranteed due to the following:

- The expansion rules do not remove any nodes from the graph, nor remove concepts from node labels nor change edge labels.
- For each node, the number of times that the *fil*-Rule or the *ch*-Rule can be applied is bounded by the size of $\mathcal{P}$. All other rules are applied at most once.
- The number of fillers created for each node is bounded by $max \times |N_{\mathrm{R}}|$.
- Nodes are labeled with non-empty subsets of $clos(\varphi)$ and edges with subsets of N_{R}. Since we do not admit any TBox, this means that there can be at most 2^{2lr} different possible labeling for a pair of nodes and an edge. No cyclic descriptions are allowed which means that we do not need to implement any blocking strategies and the completion graph is a finite tree.
- Getting a solution for the inequations will not affect termination of the expansion rules since we assume that the inequation solver always terminates.

Lemma 5 (Soundness). *If the expansion rules can be applied to φ such that they yield a complete and clash-free completion graph, then φ has a tableau.*

Proof. A tableau T can be obtained from a clash-free completion graph G by mapping nodes in G to individuals in T which can be defined from G as T $= (S, \mathcal{L}', \mathcal{E})$ such that: $S = V$; $\mathcal{L}'(x) = \mathcal{L}(x)$; and $\mathcal{E}(R) = \{\langle x, y\rangle \in E \,|\, R \in \mathcal{L}(\langle x, y\rangle)\}$. T is then a tableau for φ due to the following:

Properties 1,2, and 3 of a tableau are satisfied because G is clash-free and complete. For property 4, assume $\forall S.C \in \mathcal{L}'(x)$ and $\langle x, y\rangle \in \mathcal{E}(S)$ then $C \in \mathcal{L}'(y)$, otherwise the $\forall$-Rule would be applicable. Property 5 is similarly satisfied.

Property 6 and property 7 of a tableau cannot be violated due to Lemma 2 and due to the lowest priority of the *fil-Rule*. Role fillers are created only after making sure that all arithmetic expressions for a node x are satisfied by having a distribution of these fillers (a non-negative integer solution for the inequations in $\mathcal{L}_E(x)$) otherwise G would not be complete. Therefore, for each $(\geq nS) \in \mathcal{L}'(x)$ and $(\leq mS) \in \mathcal{L}'(x)$, $\#S^T(x) \geq n$ and $\#S^T(x) \leq m$ are valid.

The definition of *R-filler* takes into account the role hierarchies introduced at pre-processing and ensures that property 8 is always satisfied. □

Lemma 6 (Completeness). *If φ has a tableau, then the expansion rules can be applied to φ such that they yield a complete and clash-free completion graph.*

Proof. The proof is inspired by the one found in [5]. Let T $= (\mathbf{S}, \mathcal{L}', \mathcal{E})$ be a tableau for φ, T can be used to guide the application of the expansion rules. So that our proof is self contained, we redefine the mapping function π from [5] and extend it so that it can handle the properties defined in our tableau. π is defined as a mapping from nodes in the graph G to individuals in $\mathbf{S}$, inductively with the generation of new nodes, such that for each $x, y \in V_{\mathrm{G}}$ and a role $R \in N_{\mathrm{R}}$ we have:

1. $\mathcal{L}(x) \subseteq \mathcal{L}'(\pi(x))$

2. if $\langle x, y\rangle \in E_{\mathrm{G}}$ and $\mathrm{S} \in \mathcal{L}(\langle x, y\rangle)$, then $\langle \pi(x), \pi(y)\rangle \in \mathcal{E}(S)$
3. $\xi(R, \geq, n) \in \mathcal{L}_E(x)$ implies $\#R^T(\pi(x)) \geq n$
4. $\xi(R, \leq, n) \in \mathcal{L}_E(x)$ implies $\#R^T(\pi(x)) \leq n$

The claim is that having a completion graph G that satisfies the properties of π we can apply the expansion rules defined in Fig. 1 to G without violating the properties of π. Initially G consists of a single node x_0 such that $\varphi \in \mathcal{L}(x_0)$. Since we can have a tableau T for G, we can set $s_0 = \pi(x_0)$ for some $s_0 \in \mathbf{S}$ with $\varphi \in \mathcal{L}'(s_0)$. Whenever we can apply the expansion rules to G, the properties of π are not violated: applying the $\sqcap$-Rule, the $\sqcup$-Rule, the $\forall$-Rule or the $\forall_{(\backslash)}$-Rule strictly extends the label of a node x without violating the above properties due to properties 1-5 of a tableau. Let us consider applying the other rules:

- The ***ch*-Rule**: This rule strictly extends the system of inequations that is in the label $\mathcal{L}_E$ of a node x. It non-deterministically assigns a cardinality (0 or ≥ 1) for role fillers partitions that are not assigned any value, and therefore no properties of π can be violated.
- The **$\geq$-Rule** and **$\leq$-Rule**: If $(\geq nR), (\leq mR) \in \mathcal{L}(x)$, then $(\geq nR), (\leq mR) \in \mathcal{L}'(\pi(x))$, this implies that $\#R^T(\pi(x)) \geq n$, $\#R^T(\pi(x)) \leq m$, (properties 6 and 7 of a tableau). Applying the $\geq$-Rule, $\leq$-Rule extends $\mathcal{L}_E(x)$ with $\xi(R, \geq, n)$, $\xi(R, \leq, m)$ which is still conform with the properties of π and those of a tableau.
- The ***fil*-Rule**: When no other rule is applicable to x, this means that every $(\geq nR), (\leq mR) \in \mathcal{L}(x)$ is converted to $\xi(R, \geq, n)$, $\xi(R, \leq, m) \in \mathcal{L}_E(x)$. For each $\xi(R, \geq, n) \in \mathcal{L}_E(x)$ we have $(\geq nR) \in \mathcal{L}'(\pi(x))$ and $\#R^T(\pi(x)) \geq n$ which implies that there are at least $v_1 \ldots v_n$ distinct individuals in $\mathbf{S}$ such that $\langle \pi(x), v_i\rangle \in \mathcal{E}(R)$ for $1 \leq i \leq n$. A distribution of *R-fillers* is encoded in a solution σ for $\mathcal{L}_E(x)$. With each application j of this rule, p_j new nodes $y_{1_{\mathrm{j}}} \ldots y_{p_j}$ are created such that $1 \leq p_j \leq n$, $\langle x, y_{p_j}\rangle \in E$, $R \in \mathcal{L}(\langle x, y_{p_j}\rangle)$. After all variable solutions are exhausted and the rule is not applicable anymore, the number of all p_j nodes created satisfies $(p_1 + p_2 + \ldots p_{\mathrm{j}} \geq n)$, by setting $\pi = \pi[y_{1_1} \rightarrow v_1 \ldots y_{p_1} \rightarrow v_p, y_{2_1} \rightarrow v_{p_1+1} \ldots y_{p_2} \rightarrow v_{(p_1+p_2)}, \ldots y_{p_j} \rightarrow v_{(p_1+p_2 \ldots + p_j)}]$. One can easily see that the properties of π are satisfied.

The resulting graph G is clash free and complete due to the following properties:

1. G cannot contain a node x such that $\{C, \doteq\neg C\} \subseteq \mathcal{L}(x)$ since $\mathcal{L}(x) \subseteq \mathcal{L}'(\pi(x))$ and property 1 of the definition of a tableau would be violated.
2. G cannot contain a node x such that $\mathcal{L}_E(x)$ is unsolvable. If $\mathcal{L}_E(x)$ is unsolvable, this means that for some role $R \in N_{\mathrm{R}}$ we have $\{\xi(R, \leq, n), \xi(R, \geq, m)\} \subseteq \mathcal{L}_E(x)$, and we can have no possible distribution of *R-fillers* to satisfy $\{\geq mR, \leq nR\} \subseteq \mathcal{L}(x)$, hence property 6 and/or 7 of a tableau would be violated due to the equivalence properties between $\xi(R, \leq, n)$, $\xi(R, \geq, m) \in \mathcal{L}_E(x)$ and ($\#R^T(\pi(x)) \leq n$), and ($\#R^T(\pi(x)) \geq m$) respectively.

The completeness of our hybrid algorithm is thus proved.

4 Discussion

In this paper we have presented a hybrid tableau algorithm that efficiently handles numerical restrictions of $\mathcal{ALCQ}$ concept descriptions. The reasoning algorithm is improved by means of a more informed search where the size of the completion graph is reduced to the minimum. The hybrid algorithm fills the gaps between tableau algorithms [1, 5] which do not adequately handle numerical reasoning, and the arithmetic reasoning for description logic in [7] where no calculus was proposed and all the input is reduced to equation solving. This paper proves correctness of the hybrid algorithm as a decision procedure for testing satisfiability of $\mathcal{ALCQ}$ concepts, whereas [3] provides no proof for the *recursive* algorithm it proposes. In contrast with [3], this hybrid algorithm is potential to be extended to handle more expressive languages. This algorithm enjoys the benefits of using an arithmetic reasoner such as relatively faster reasoning and insensitivity to the values of numbers used in arithmetic expressions. On the other hand by collecting all number restrictions and then trying to satisfy them with minimum number of fillers, we reduce the size of the model and avoid creating role fillers when an arithmetic clash is triggered.

For reasons of simplicity we introduced this hybrid approach with a simple logic such as $\mathcal{ALCQ}$ with empty TBox and empty role hierarchy. However, it is not hard to see that this calculus can be easily adapted to work with role hierarchies and general TBoxes. Such an extension is part of ongoing work where we study an optimal way of creating role fillers using one proxy individual (one representative individual) as proposed in the signature calculus [2]. Finally, a prototype implemention based on this calculus for the DL $\mathcal{ALCHQ}$ is completed and preliminary results demonstrate a promising improvement in performance for satisfiability tests.

References

1. Baader, F., and Sattler, U. An overview of tableau algorithms for description logics. *Studia Logica 69* (2001), 5–40.
2. Haarslev, V., and Möller, R. Optimizing reasoning in description logics with qualified number restrictions. In *Description Logics* (2001).
3. Haarslev, V., Timmann, M., and Möller, R. Combining tableaux and algebraic methods for reasoning with qualified number restrictions. In *Description Logics* (2001), pp. 152–161.
4. Hollunder, B., and Baader, F. Qualifying number restrictions in concept languages. In *Proc. of KR-91*, pp. 335–346.
5. Horrocks, I., Sattler, U., and Tobies, S. Practical reasoning for expressive description logics. In *Proc. of LPAR-99* (1999), pp. 161–180.
6. Ohlbach, H. J., and Koehler, J. Role hierarchies and number restrictions. In *Description Logics* (1997).
7. Ohlbach, H. J., and Koehler, J. Modal logics description logics and arithmetic reasoning. *Artificial Intelligence 109*, 1-2 (1999), 1–31.